ENCYCLOPEDIA OF LIBRARY AND INFORMATION SCIENCE

Executive Editor

ALLEN KENT

SCHOOL OF INFORMATION SCIENCES
UNIVERSITY OF PITTSBURGH
PITTSBURGH, PENNSYLVANIA

Administrative Editor

CAROLYN M. HALL

ARLINGTON, TEXAS

VOLUME 70

SUPPLEMENT 33

MARCEL DEKKER

MARCEL DEKKER, INC. NEW YORK • BASEL

BIBLIOGRAPHY

CIO/WebMaster Magazine's Intranet Research Center; http://www.cio.com/forums/intranet/.
Complete Intranet Resource—Intranet Reference Site; http://www.intrack.com/intranet/.
Computerworld Intranets, monthly June 1996 to Sept. 1998.
Earthweb's *Intranet Journal,* http://www.intranetjournal.com
I-net Organization, http://www.iorg.com.
Intranet Design Magazine. http://idm.internet.com/.
Intranet Resources, http://www.strom.com/ics/intra resources.htm.
Scott, J. E., "Organizational Knowledge and the Intranet." *Decis. Sup. Syst.,* **23**, 3–17 (1998).

JUDY E. SCOTT

PERSONAL BIBLIOGRAPHIC SYSTEMS

Introduction

This discussion of Personal Bibliographic Systems (PBS) features deals with microcomputer programs that manage the input, storage, retrieval, and output of bibliographic references. Although various partially overlapping terminologies may be encountered in the literature such programs as *EndNote, ProCite,* and *Reference Manager*—three of the most popular PBS—have by now become household names in academic circles. Several dozens of similar programs exist, each with its own peculiar combination of desirable and less appropriate features. The names of over 100 PBS examples are listed in Appendix 2.

INPUT

From the early days of personal computing, the idea of replacing the traditional stacks of bibliographic reference cards by adequately manageable and searchable computer files has had its appeal. Although keying in these references was generally a tedious job, once they were entered in the database, one could easily select any combination of records and reproduce them in a wide variety of ways. The ability to eliminate manual input while still having your personal selection of references in a highly manipulable database makes this concept all the more attractive. In the last 15 years, the availability of bibliographic references in an electronic formant has increased dramatically. in the 1960s and 1970s, these generally resulted from rather expensive searches on the large online hosts, intermediated by information professionals. Whereas today this is still an option [e.g., through selective dissemination of information (SDI) services], from the 1980s onward, access has become far more democratic and widespread through end-user searching of CD-ROM databases and current awareness products, such as *Current Contents on Diskette.* In the 1990s, the Internet became an additional source of (often free) bibliographic references. When such ubiquitous electronic records are that easy to acquire, the opportunity to store them in a database and have them ready to retrieve and recycle in all sorts of ways clearly makes this "saving" of records more than worth its while. The incentive to use this kind of software is obviously booming. This evolution may become even more advantageous when full-text information is involved, but this does not really concern this survey.

STORAGE

Accommodating for bibliographic records, PBS are essentially database managers coupling both structured and textual information. As such, they belong to the continuum of the diverse software group Sieverts et al. have labeled "information storage and retrieval" (ISR) software in their classic series of articles (*1–7*). Although individual PBS may differ substantially from one another, most of them could be classified in Sieverts's typology under the "classical retrieval systems" (*2*) and "end-user software" (*3, 8*) subdivision. PBS has less affinity with other ISR categories, such as "indexing programs," "full-text retrieval programs," "personal information managers," "hypertext programs," and "relevance ranking programs," although some packages may combine characteristics of different classes.

Managing bibliographic records, PBS are also related to the catalog modules of automated library systems. Yet there are essential differences. First, PBS are generally used to accommodate personal collections consisting of thousands of records, not hundreds of thousands or millions. Many PBS can accommodate only a limited number of records (typically, 32,00 or 64,000) or may not have adequate techniques to guarantee fast retrieval in more extensive databases. They are better suited for personal applications, such as managing reprint collections. However, PBS are not just downsized library catalogs. The bibliographic records that manage tend to be of a more variable nature than the more or less uniformly structured book items one typically finds in library catalogs. PBS must allow for a number of different document types to accommodate such diverse publication types as journal articles, books, book chapters, dissertations, reports, unpublished papers, electronic documents, and Internet-derived media, preferably mixing these types within the same database. This has consequences on all the levels involved; the PBS must be able to import these highly divergent record types, whether through laborious keyboard typing or by uploading and converting electronic records from external sources, store them adequately, make them easily retrievable, and export them in a flexible way. So, on the one hand, the need for malleability is far higher than in electronic library catalogs. On the other hand, they have no need for the various administrative functions that are so essential for full-sized integrated library systems, such as dynamic linking of book records with patron or bookseller files. As such, these "personal" programs can also be valuable tools for research workgroups, smaller libraries, and documentation centers, where versatility and structural flexibility matter more than sheer bulk.

As database managers, PBS can vary strongly in concept; some are rather basic, modeled on the early general-purpose database management systems such as *dBase* and *Paradox*, which allow for the bare necessities of storage, retrieval, and output of records, but they are not really suited to deal with the specific properties of bibliographic information. A major handicap was their structural limitation to fixed-length fields and predefined numbers of occurrences for each field, which continually forced trade-offs between artificially restricting the amount of data that can actually be stored and reserving an inordinate amount of "wasted" space for filler blanks. PBS soon overcame this obvious handicap by offering variable-field lengths. Although with today's multigigabyte hard disks, economy of disk space may no longer be that important; back when 10-megabyte hard disks were respectable, offering variable-field lengths was a true asset. This illustrates how the constantly evolving hardware has its influence on the evolution and utilization of PBS; with today's powerful microcomputers, a number of previously critical features are no longer problematic and new perspectives surface, such as the standard ability to open several databases or programs simultaneously and fast sequential retrieval in modest sized databases. Conversely, this implies that fairly new microcomputers may be needed to run the more recent systems.

RETRIEVAL

When searching a PBS, the typical aim is to quickly find specific references; for instance, and article author A published in journal B during year C. It is therefore not surprising that in many PBS, retrieval options are fairly straightforward and basically limiting, narrowing subsequent options until the correct (set of) reference(s) is found. The first search formulation obviously examines the full database (possibly limited to a specific field). Except for the odd widening (OR) operation, all subsequent searches tend to be relative to the set created immediately before. Generally, these combinations are made using the traditional Boolean operators AND, OR, (AND) NOT, or field indicators. They may also offer more subtle relations, such as "containing," "starting with," the more exotic "ending with," and all kinds of numerical (and alphabetical) ranges. More recently developed ISR systems (or program versions) may use alternative procedures, often based on inventive artificial intelligence algorithms. Although these may be more appropriate for finding information within less-structured full-text collections, some of these techniques can also be helpful for PBS (e.g., synonym activation, sound-alikes, or look-alikes). Another example of increasing sophistication is the inclusion (or simulation) of relational (like) features, offering pleasant hypertext functionality or (field-) index-based navigation.

OUTPUT

Whereas adequate structure, comfortable input, and easy retrieval are essential features, for many users the true eye-catcher and valediction for starting with a PBS is the versatility of its output module. Whereas fielded screen display or printouts and downloads are common in most catalogs or online or CD-ROM databases, PBS distinguish themselves by their ability to reconstruct individual records in a virtually unlimited number of ways. As research papers are now generally written using word-processing programs, the idea of letting the microcomputer also handle the in-text citations and the list of bibliographic references has become quite popular. Although word processors can do an amazing amount (think, e.g., of elaborate endnote and footnote possibilities), their global editing capabilities and spelling and vocabulary checkers are just not up to the automated reformatting of existing database records into bibliographies tailored to any of the hundreds of different bibliographic styles used in the scientific literature, each one featuring its own peculiar intricacies. This is where PBS are supposed to excel. While writing your manuscript with a word processor, you only need to insert an adequate pointer (e.g., record number or a combination of author and publication year) and the PBS takes complete care of the production of the reference list on the spot, thus eliminating the tedious manual job of typing and correctly formatting bibliographies. If you later need to send the manuscript to another target journal featuring a different style, you only need to select another style option from the menu and the system automatically creates a new reference list, equally correct for its changed purposes. Good PBS are not limited to correctly structured individual records, but they also take care of all aspects of generating reference lists, including record arrangement and page formatting. Obviously, these qualities are equally effective for making stand-alone bibliographies. It is not surprising that such qualities are increasingly being foregrounded in the literature and the ample advertisements. The obvious added value in this case is the number of bibliographic styles included with the program (or sold separately), their accuracy, and the ability to do something about incomplete or inaccurate results. Equally important, if full records can be exported in a format recognizable by other PBS, or at least a more or less standardized exchange format, the risk of these valuable data being locked up within an aging system is mitigated.

As is clear, automated bibliography formatting for manuscripts involves at least two programs. For DOS users, this used to be a rather cumbersome process, consecutively dealing with each program separately, meanwhile necessarily closing the other. With the Windows platform, this interaction became far more comfortable, with the ability to switch between two programs without the need to close either of them. In the luckier cases, the Windows clipboard could be used to copy and paste (sets of) formatted records. For many modern PBS, the cooperation with word processors has now evolved into a kind of seamless integration, in which one can deal with the other's contents (e.g., the PBS can read the word processor's document source text, or vice versa) or even a symbiosis in which one program comfortably nestles itself within the other by plugging in a series of options in the other one's menu structure.

DATABASE MANAGEMENT VERSUS BIBLIOGRAPHY FORMATTING

Although present-day PBS excel in bibliography formatting, it should always be borne in mind that they can do these often amazing feats only if the pertinent records are already available in the database, can be retrieved unambiguously, and have finely enough tuned field structures to allow for the necessary manipulations. Whatever the potential versatility of the output module, the records must first have been stored in a correct and sufficiently detailed fashion, as "garbage in, garbage out" is undeniably a relevant motto in these matters. Although, it is possible to upload thousands of records in your database in a matter of minutes, if specific items such as "author initials" or "cited volume" are not available in the correct (sub)field(s), it may not be possible later to automatically truncate or invert these initials or print the volume number in bold, underlined, or italic type. So before, PBS can become elegant bibliography formatters, they must first be capable database managers.

Whereas several PBS combine database management and bibliography formatting admirably, other specialize in one of these two functions [and on another level, there is the eternal trade-off between versatility and (immediate) ease of use]. Thus, the usefulness of a PBS may not so much depend on "objective" superiority, but rather on the user's functional priorities; whereas proficient academic researchers writing lots of papers will especially enjoy the automatic bibliography formatting features, those that read a lot will find the database aspects helpful to manage their often extensive reprint collections (including articles, books, unpublished reports, etc.)—and what scientist today does not or is not compelled to combine both gullible reading with proficient writing?

WINDOWS

Personal Bibliographic Systems software originated long before the Windows age. Excepting a number of early Macintosh programs, most microcomputer-based interfaces were not graphic, but character-based. Basically, they presented themselves on screens consisting of 25 lines with 80 positions each with invariable nonproportional fonts. The major navigational instruments were commands, which, in the worst cases, needed to be memorized and typed in full at the DOS prompt. In more merciful cases, there were menus in which options also needed to be typed or could be selected by moving the cursor, or by typing a specific (highlighted) character. The menus generally allowed no more than a single option per line; therefore, only a limited amount of information could be displayed on each screen. Function keys could be used as shortcuts, but if not displayed on the screen, they, too, needed to be memorized.

By now, the leading PBS have been upgraded to Windows version (9). As with all Windows software, the way of interacting with the system is quite different from DOS applications. First, a lot more information can be displayed on the screen, starting with the fonts, which can be modified. This allows for the WYSIWYG ("what you see is what you get") approach: different fonts, variable sizes, and presentation modes, such as underlined, bold, italic, subscript, and superscript, are displayed as such. The screen can be separated in different easily resizable or relocatable windows, either dividing the screen between them ("tile") or overlapping one another ("cascade"). Each window can contain far more information than what is immediately visible on the screen (partition). The invisible part can easily be uncovered using horizontal and vertical scroll bars and elevator cages or by enlarging the active window. In Windows applications, the general layout and many of the functions tend to be fairly consistent (e.g., similar <Alt> menus; extensive help and standardized exit procedures). Windows features fast manipulation and activation of functions using the mouse and icons, buttons (several per line), and selection boxes (often masking many options on one line). With the arrival of full 32-bit Windows95 versions, long file names, self-designed toolbars for quick execution, context-sensitive menus (right mouse support), and tailored database start-up shortcuts became standard features in state-of-the-art PBS versions.

Whether the appreciation of this graphic approach is a matter of personal taste or a true revolution welcomed by all is open for discussion, but one genuine advantage of Windows is certainly the increasing feel of familiarity resulting in a kind of de facto standardization. Modern PBS have not only come to resemble each other more and more, but as fast as the interface is concerned, they also resemble other Windows-based applications. For instance, *ProCite* and *Reference Manager* now superficially show a striking resemblance to each other, offering similar screen display features and program modules. Yet upon closer inspection, it remains clear that both are still based on their highly divergent early versions, each featuring quite different data structures. Also, although one uses predominantly sequential retrieval, with the addition of a few field indexes the other is strongly index-based and does not allow fully sequential searches. One reason may be that both these top PBS originally were close competitors but have in the meantime become artificial siblings, now distributed by the same company [which is also related to the Institute of Scientific Information (ISI), a major provider of electronic bibliographic information].

As mentioned earlier, another major advantage of Windows over DOS is the standard capacity to temporarily leave a Windows application and execute other jobs without the need to close the former one, and to transport information from one application to the other using the clipboard-based copy/cut/paste techniques or the drag and drop possibilities. This flexibility allows for a modest form of multitasking, of which even DOS programs run from within Windows can partially profit.

Next to DOS and Windows programs, Macintosh versions or hybrids are quite common (although still a minority). Some were derived from existing programs from nonmicrocomputer environments, such as UNIX. When using different platforms, it is a great advantage when all PBS versions can use the same database file without the need for conversion. The same is true for subsequent software releases (i.e., different versions). If not, easy convertibility of the database to an upgraded version is a must, but this tends to be irreversible. As Windows98 does not appear to constitute a dramatic departure from Windows95, it may not immediately inspire major PBS interface changes (except maybe for Internet-related information).

CONNECTIVITY

The growing convergence is not limited to superficial cosmetic touches, but it appreciably increase functionality. Cooperation and compatibility now appear to be far more fashionable than they used to be. Over the last few years there has been a trend toward the convertibility of PBS to one another (i.e., the facility of importing one system's records into another's databases.) Not surprisingly, the capacities for the importation of records from foreign systems are better advertised than the reverse functionality. Another important development in the evolution of text-based databases was the increased connectivity to all kinds of links to external programs, allowing graphics (images and other objects) to be linked using object linking and embedding (OLE) and dynamic data exchange (DDE) technology. This may be more interesting for other types of ISR than for those dealing with bibliographic records, but it could still be useful for illuminating records or including images of (non-OCRed) full-text versions, or simply to let a PBS double as a multimedia collection catalog. Apart from intrasectorial cooperation, there is also a trend toward increased cooperation with other complementary services from a different niche (e.g., current awareness services, such as ISI's *Current Contents* or *Research Alert,* popular online databases, such as ISI's *Web of Science* or *Medline* (especially through the free WWW-based *PubMed* gateway), or word processors, such as *Microsoft Word* and *Corell WordPerfect*), as mentioned earlier. Another recent value addition is the integration of Internet-based functionality; for instance, allowing for HTML documents or Websites to be described adequately in a bibliographic database, or the reverse—to produce ready-made bibliographies in HTML format, but also active linking from within the database records to the actual (full-text) documents. Also, the standard opportunity to offer searchable PBS databases on a Webserver has recently emerged. So, where the PBS used to be one link in a chain of complementary but separate modules, they are becoming more and more "all-in-one" packages, integrating such functionalities as online communication, record conversion, database storage and retrieval, bibliography generator, and word processor. Thus, PBS have reached a third level of overall functionality. In the early days of PBS, bibliographic references were typed from the keyboard and eventually printed on paper. This straightforward pattern was revolutionized when external records could be uploaded en masse and bibliographies generated from within the word processor. This often implied some extra software and a little tinkering, but generally the PBS was connected to the world and a lot of time-consuming routine work could be eliminated. Now, PBS are integrated packages offering high levels of connectivity and compatibility, importing from and exporting to all kinds of sources and targets, including the World Wide Web.

A Survey of PBS Features

In the following sections, a number of characteristics that individual PBS do or do not feature are discussed in a generic way. Most of these items are available in at least one of the mainstream systems. Some additional features can be encountered in packages that are not strictly PBS (e.g., some CD-ROM interfaces) but that would do credit to any PBS. As both the advantages and the disadvantages of individual features are not always immediately obvious, this discussion may prove helpful when shopping for a PBS [or selecting CD-ROM interfaces (*10*)]. First, the very heart of these systems will be studied; that is, the structural possibilities and limits of the database file itself, as these define, arguably more than any other issue, whether or not a PBS is adequate for the user's basic needs. Further on, the actual database functions such as input, retrieval, and output will be discussed. Finally, a number of interface- and management-related issues will be discussed. A product-independent table of desirable features is included as an appendix. For a discussion of the major PBS themselves, I refer the reader to a number of excellent review articles (*1–7, 9, 11–21*). Although several PBS have become obsolete by now, many others are flowering, releasing new major upgrades every few years. Stigleman (*9*) lists about 50 different PBS names is included as an appendix. An earlier version of the main text was published as a journal article (*22*).

Database Limitations and Structural Flexibility

CAPACITY

Personal Bibliographical Systems are generally not designed to accommodate an unlimited amount of information. There can be limitations to the number of databases that can be maintained, as well as to the number of records, fields per record, characters per field, or the total amount of bytes per field, per record, or per database. To what extent such limitations constitute a problem depends on what you want to achieve with a PBS. This goal determines the capacity items you must look for. If you want to keep a number of autonomous databases, it is not much use buying a PBS that only allows for one database, whatever its otherwise breathtaking features. If you intend to catalog over 40,000 books, a software package that limits its database(s) to 32,000 records just will not do. Conversely, there is no need to look for a PBS with virtually unlimited capacity when you only want to catalog your personal reprint collection, which may never surpass 5000 items. Although this observation is essentially still true, the growing tendency to remove or mitigate such limitations is also a good example of how some practicalities of earlier days have now been largely removed by improved software and enhanced hardware capacities.

DOCUMENT TYPES

When dealing with bibliographic references, it is essential that the system can handle highly structured data. This does not imply that it is not possible to find relevant information in nonstructured texts. This is often done admirably by highly sophisticated full-text storage and retrieval software. However, in order to retrieve or (re)produce specific bibliographic elements, it is necessary to keep these different pieces of information in separate (sub)fields, grouped in adequate document types.

Whereas some software packages may present themselves as "naked" tool boxes at the disposal of creative enthusiasts willing and able to spend some time to develop the bibliographic system of their dreams from scratch, most PBS offer a number of standard document types, such as journal article, book, book chapter, and dissertation. Some include quite a variety of predefined formats, including videotapes, electronic files (nowadays also including such Internet-based formats as electronic mail messages or hypertext markup language (HTML) documents and uniform resource locators (URLs), and other nonbook materials. If you believe the variety of document types offered lives up to your expectations, there is not much point in looking for further structural flexibility. However, you may feel the need to modify these formats or add extra ones. This may not be that important for personal reprint databases, where you can put divergent materials within the constraints of existing format. The field labels or output formats may not be fully representative of the data included, but for personal purposes, they will suffice. In a library situation, however, it may be necessary to alter these formats. If you want to incorporate a great variety of materials in your database(s) (e.g., including unpublished documents or annual reports), it is no good if your system only allows for standard book or journal article formats. Even if an abundant array of formats are available, it may be necessary to create new fields, adjust some field tags, and so forth. Therefore, it is certainly a boon if you are allowed to change field characteristics and the structure of document types. Another important issue is that of a posteriori flexibility: Can fields (or document types) be added, moved, renamed, or deleted without limitation once the database structure is defined, and what effect does this have on existing records (e.g., are field contents moved to other fields or just removed from the record)?

In the most flexible PBS, practically all fields and record types can be defined freely, which allows you to accommodate totally different kinds of information (even including nonbibliographic data). Practical examples are, for instance, video catalogs, musical LP/CD and track databases, interlibrary loan administration, transaction logging information for database usage, and even "expert system"-like databases. Of course, this is not what you buy a PBS for in the first place, but if it fulfills your bibliographic needs, it is certainly an asset if the same system can also deal with such other (media) materials or managerial information. Of course, flexibility of structure can only be fully enjoyed if it is matched by an equal flexibility of display and output formats. It does not make much sense to rename or create fields when they cannot be displayed, printed, or downloaded in an adequate fashion. Tinkering with predefined structures also implies that some system-supplied formats (e.g., journal output styles) may no longer give correct results unless they are equally doctored. Therefore, there generally is a price attached to flexibility. On the other hand, using only the default predefined modules offers a sort of comfortable plug-and-play ease of use.

FIELDS

Unlike general-purpose databases using fixed length fields to deal with numerical data, postal addresses, and the like, bibliographic databases should be able to provide for information with highly variable sizes. Fields can be just a few bytes long, in which case it would be a waste to reserve a fixed number of bytes for each potential occurrence of each field for each record in the database. On the other hand, field contents such as corporate sources or abstracts can occasionally get quite long. In many systems, there is a limit to field lengths, but this limit can be several thousand bytes, so it generally poses no practical problems. Even though megabytes are getting less expensive every day, it is obvious that variable-field length is still a desirable feature for bibliographic data, both as a prevention of space waste (economy) and as a virtual absence of limits, allowing high accuracy [e.g., full corporate source names and (serial) titles]. Some PBS offer a compromise with hybrid types featuring a number of indexed fields of limited length, coupled with one or a few of long length that are not indexed.

Not all systems are equally user-friendly in exploiting their field structures. Many use short alphabetic tags of one or a few characters. There are often mnemonic, so it is easy to remember that the title field is coded with "T" or "TI". This makes it easy to do field-specific searches. This is far less evident when fields are identified with numerical tags (compare MARC codes), which are not so easy to remember or guess for experienced users, let alone for novice users. When working with fields and field tags, there should be an easy way to get a survey of these fields and their major characteristics (e.g., via pop-up or drag-down lists). Furthermore, this basic field indication may be complemented by the interface [e.g., by automatically providing full field names for display purposes (e.g., "T" is transformed into "TITLE:")].

Some systems allow for individual subfields. This can be useful to give initials more autonomy or to work with more than one publication date (e.g., one for retrieval purposes, another for actual output), or conversely, to put related information (e.g., imprint) within one and the same field. In most instances, however, this can equally be resolved by using separate fields for each individual type of information.

With the advent of the World Wide Web, a new type of field has been adopted by the leading PBS; URLs or HTML text can be inserted in just any field (e.g., "notes"), but new functionalities have evolved, so these Internet-based formats are not only searchable in their own right, but merely clicking on them activates the Web browser and loads the page to which the record refers (as is also common with electronic mail).

CHARACTERS

For bibliographic purposes, it is essential that all necessary characters be adequately supported by the system. Next to the standard alphanumeric ASCII characters, there is also a need for the extended ASCII set, including for instance, the French accented characters and other diacritics. In the early days, this used to be a problem with many Anglo-Saxon systems, but nowadays most PBS support the extended ASCII set (256 characters), so the most prevalent foreign texts can be accommodated. In the future, the more "exotic" characters (e.g., Japanese) may be included if the current 8-bit character sets are ever replaced by 16-bit sets. The same is true for scientific and graphic symbols. In this respect, there is not only the question of input and storage to be addressed, but there are also repercussions on other levels, such as retrieval, display, printing, downloading, and uploading in word processors—and with each of them, there is the issue of (self-definable) sorting values.

Another advantageous feature is the capacity to foreground or highlight certain parts of the text, so that these strings can be displayed or printed in bold, italic, underlined, superscript, and subscript, independent of the actual display or output formats or the type of field to which they belong.

RECORD NUMBERING AND LINKING

It may be helpful to dispose of an explicit permanent number for each record, so it can still be uniquely identified when intermediate records are deleted or the database is rearranged. Useful options include the automatic generation of such numbers and the possibility to modify single record numbers or renumber a complete database. Setting the automatic number increments to a higher value than one allows you to later reclaim the unused intermediate numbers to place newer records in the vicinity of specific older ones, thus manipulating, for instance, the browsing order. This may be less important with the newer Windows-based systems, which, if presented in columns, can change the sort order by simply clicking on the key field's label. Still, one sort key may give far faster results than others, as it is already index-based while the others need to perform ad hoc sorting. Automatic generation of the date of record creation and last modification may also be useful.

Some systems allow the linking of separate records. In this way, pointers to hierarchically related records can be included, so a lot of bibliographic information of the parent record need not be duplicated within each individual child record, thus guaranteeing economy and consistency. This is generally possible in the relational database category, but it can also be found in some other PBS types.

In the last few years, multimedia capabilities have gained in popularity. Ever more, PBS can now handle images or graphics, either through limited links to external files and viewing software or by full integration, including internal viewing software. This may not only be useful for displaying pictures but also for more text-related items such as diagrams, mathematical formulas, and chemical structures. Managing sound files appears to be less common in PBS. The capability to interpret and produce bar codes can be useful in library environments. Another type of linking is that to full-text sources, either on the local hard disk or CD-ROM, or Internet-based (e.g., Web pages, library catalogs).

Database Selection

Database selection generally the first option you are presented with when starting up the system. A good PBS makes it easy to access the right database. Therefore, it is important that all available databases are listed in an adequate way. Most systems display this choice via a menu; the database of choice can then be selected by either typing a number of a highlighted character from the name, or by manipulating the cursor keys (DOS) or (double-) clicking on a menu item with the mouse (Windows). The length allowed for database names may also influence the ease of selection; again Windows95 systems will have advantages. Having access to other database directories is also helpful. In this way, a hierarchical structure can be maintained, in which each type of database has its specific subdirectory; thus, related databases are displayed within the same menu excluding other irrelevant ones. This may also be feasible within one default database directory, but then the system should offer ways to customize the presentation of databases (i.e., in an order other than the alphabetical or chronological default). Another advantage is the ability to define a default database that is automatically opened at system start-up. These database-selection issues may sound rather far-fetched, but some early popular systems did not list the available databases, so you could not open a database when you did not know its exact name and location (path). This can only be helpful if you want to avoid others peeking into your databases.

Entering New Records (Input)

MANUAL INPUT

Keying bibliographic references in a database is a tedious chore. Any help to enlighten this job, avoid unnecessary duplication of effort, and minimize spelling mistakes is more than welcome. Generally, manual input is achieved using an electronic input sheet offering a number of fields to be filled in. Preferably, only those fields that are relevant for the specific document type are presented. Having different input sheet for different document types (one of which should be the system default, or the de facto default until changed explicitly) avoids being confronted with a lot of irrelevant fields during manual input. Also, it should be easy to customize the information you have just entered using this default input sheet, so that when necessary, less common fields can also be added (provided they are defined in the field tables).

Input modules generally make use of some kind of text editor. These may differ greatly in versatility and power. When using a simple line editor, a line can no longer be modified once the cursor has reached the following line. Full-screen editors are far more flexible and allow jumping from one line to other lines above or below. Good editors can make life much easier. A common example is the capability to automatically replace or copy strings or full-field contents to other fields within the same record (copy, cut and paste). A comparable issue is the creation of new records by duplicating existing records [e.g., for articles in the same journal (issue), chapters in the same book and books in the same series]. If this function is not explicitly available, Windows users may be able to use the standard clipboard and copy/cut/paste or grad and drop techniques. DOS users may be able to mimic this by exporting the record in question and subsequently importing it, but this is obviously a less comfortable and speedy surrogate. Another useful feature is the option to specify a default content for specific fields (e.g., journal or book title) or automatically add field occurrences (e.g., specific keywords) to all new records created during an input session. Some PBS offer automatic generation of keywords from the title or abstract field (optionally coupled to a stop list, or a "go" list, for that matter). It is not always easy to enter foreign characters or diacritics when you do not know their ASCII value by heart (and often impossible when these do not belong to the extended ASCII set). Some programs have special modules that can remap your keyboard (e.g., using <Alt> or <Ctrl> combinations) or display pop-up lists with foreign characters.

Index-assisted input, where the strings types are compared to information that is already available within the indexes, has two major advantages. First, it guarantees alphabetic consistency because the index(es) act(s) as authority file(s). Second, if automatic truncation is incorporated, this can dramatically shorten the amount of typing. For example, you may only need to enter "t t m" to get the rather elaborate string "Transactions of the Royal Society of Tropical Medicine and Hygiene" in your journal title field. It gets even better if the index extract also displays the actual number of hit records, thus showing the comparative popularity of each suggested item. This is helpful when choosing between several alphabetically related terms (e.g., for keywords), yet it should equally be possible to overrule this feature or temporarily neutralize it, otherwise no new index terms can be added. It may even be necessary to deviate from such authority files. Although they are very useful for enhancing uniformity and thus boosting retrieval, it should not be forgotten that one of the purposes of PBS is the generation of reference lists. Many authors use different formats of their names (e.g., one versus two versus three initials). If during manual data entry you revert the author name to its "standardized" format, the resulting bibliographic references will, strictly speaking, not be correct. Non-Western names will present their own problems, which, in some systems, can be partially prevented by typing them in a special way (e.g., with additional punctuation to overrule "normal" processing). Input-validation tables, defining what type of data (alphabetic, numeric, alphanumeric, etc.) are allowed in what field or whether certain compulsory fields are not left empty may further optimize data integrity and decrease flexibility.

IMPORTING EXTERNAL RECORDS

As discussed earlier, ever more electronic records are becoming available from external databases through all kinds of channels (diskettes, CD-ROM, online, Internet, etc). When these can easily be imported into the PBS, adding new records to the system almost becomes a zero-effort action, afterward offering the advantages of fast retrieval and flexible reformatting. Of course, there is the legal aspect of copyright. Can you just download electronic records from external databases? How many (at a time)? Can you redistribute them? As with photocopying, a kind of "fair-use" policy seems to be accepted, although the exact limits may not be very clear and a lot depends on the specific contracts (*23, 24*). Anyway, many database providers are offering their data for downloading in a recyclable format, but this may be more for comparative marketing purposes than the result of sheer altruism.

On a technical level, the uploading of electronic records is only possible if the import files do conform to the internal structure of the PBS. The more complex this structure, the more difficult it is to prepare acceptable input files. In the easiest case, the source database provides an output format tailored to match the structure of your PBS, or this internal structure conforms to a standardized data interchange format. In this case, no data manipulation is necessary—but this will be the exception rather than the rule. Therefore, many PBS provide made-to-measure import profiles that can convert records downloaded from specific sources such as popular online on CD-ROM databases to their own format. *Endlink, Biblio-Links,* and *Capture* are the reformatting modules matching *Endnote, ProCite,* and *Reference Manager,* respectively. Sophisticated PBS may offer hundreds of such conversion modules. However, even then, certain sources may not be provided for, especially when a specific discipline (such as biomedical sciences) is targeted. As far as the data structure is concerned, some PBS are quite severe, whereas others are more forgiving. Evidently, the more lenient a system is, the less adequately structured the imported records may turn out to be. Some PBS vendors are willing to add new profiles, provided the source database/host combination has a sufficiently broad user base or you are willing to pay for their creation. But then again, these conversion modules may be inaccurate or incomplete or you may have tinkered with the internal structure of the records. Therefore, it is a boon if you can edit existing profiles or add new ones, tailored to your specific needs. Originally, these reformatting profiles were separate modules distributed individually or in group(s). Newer PBS tend to integrate these often abundant profiles within the major program.

When PBS-specific conversion modules are insufficiently helpful, generic conversion programs, such as *Fangorn, Headform,* or *Refwriter,* may offer a solution. Such programs can convert practically any structured data into a standard data-exchange format. Of course, even with the best conversion programs, it will not always be possible to fully translate records from one PBS or database host to any other. For one thing, the respective interpretations of certain fields or document types may not be 100% compatible (e.g., subtle functional differences among "article authors," "book authors," and "book editors"). Even using popular standards as an intermediate format may not always be the ultimate solution. The *Medline* format, for instance, is not much of a help when dealing with nonjournal article records. Field tags may be missing completely or the internal order of fields within the records may not be fixed and mix up the conversion. Carter (*25*) lists a number of possible problems, including inconsistencies and errors in the source text, omission of correct delimiters, different author name formats, insufficiently detailed source fields, and indentation of field tags. If such problems cannot be overcome by conversion software, it may be better to load these records in your word processor and reformat them manually, possibly helped by a number of adequately designed macros.

To limit the damage, it is a good thing if during import unrecognized fields or unauthorized field contents are placed in a separate field (e.g., "note") so that these contents can later be salvaged by manually putting them in more appropriate fields. The same goes for complete records; in the best case, problem records are diverted to a temporary file, together with an explanation for their rejection. In the worst case, the complete file is rejected or the system crashes. Just as with manual input, it may be helpful if default field contents (e.g., availability indicators or additional keywords) can be generated during automated import. A special case is the importation of records from Internet-based sources, which are in the HTML format or include URLs. Some PBS offer the opportunity to store full Web page contents or to retain a self-selected portion in the database.

Automatic duplication detection is a time-saving feature. This is achieved either during import, comparing each individual new record of the uploaded file to the records already present in the database, or acting on the whole database at a later time, as a batch procedure. Again, it is an asset if you can define the criteria to judge whether two records are duplicates or not, or are prompted to judge each candidate duplicate individually. In this way, you can choose between a rigid and a more forgiving approach.

Uploading records from other databases produced with the same PBS can be considered a special case of electronic problems, yet there may be mutually incompatible field- or document-type definitions between these two databases. Merging complete databases is a more complex matter, as it may be necessary to avoid duplicate records, to keep original record numbers, or to maintain a specific ranging order.

Modifying Records (Edit)

Many helpful features for creating records are equally relevant when editing them: line versus screen editor; using only relevant (i.e., nonempty) fields versus the complete data sheet; various copy, cut, and paste capabilities; and so forth. Other important features are the maximum number of records that can be modified simultaneously using global editing (i.e., when a vast number of records or even the entire database is changed in one go) and the corresponding qualitative capacities, such as full-text modifications versus only within specific fields, and case-specific versus case-independent. Having the edit functionality available at all times (e.g., while viewing the retrieval results) is more comfortable than when it constitutes an individual module to be accessed separately. Some systems allow many types of external (and often very powerful) editors to be used instead of the standard one provided with the system.

A special case of editing is the deletion of records. Logically deleted records are not always physically removed from the database, in which case they do not free the disk space they occupy. All references to deleted records should also automatically removed, but this may imply complete reindexing of the database.

Searching the Database (Retrieval)

SEQUENTIAL VERSUS INDEX-BASED RETRIEVAL

Basically, PBS feature two broad types of searching. With sequential full-text retrieval, the system searches the database file character by character for matching strings. It is obvious that for a large database, this may not be the optimal mode of searching. This was especially true in the early years, but with powerful present-day microcomputers, sequential searching again gives acceptably fast results in smaller databases. Sequential searching can even have its own advantages: As the actual database is searched instead of the (selective) index file((s), sequential searching may get more accurate results and, as it is an independent of way of indexing, literally anything can and will be found. It is, for instance, not necessary that the filed in question be indexed or individual words be indexed separately or on the field-occurrence level. Even truncation or masking may be more effective in sequential mode.

In larger databases, indexes (inverted files) generally remain essential to guarantee fast retrieval. Single-term searches should give quasi-instantaneous results. More complex searches will give different response times in different index-based PBS. It is, of course, possible to compare these with each other, but no definition exists of what exactly is meant by fast. Evidently, the type of microcomputer (processor, clock speed, RAM) used is a highly influential factor, insofar as for modestly sized databases, there may no longer be any noticeable differences when searched with fast microcomputers. A disadvantage of using indexes is that they may take up a lot of disk space (e.g., 50% or more compared to the database file itself–there are vast differences between individual systems).

Some PBS generate one general index, with terms extracted from all meaningful fields, whereas others create field-specific indexes, so you can, for example, limit the search to title words or keywords, discarding less relevant information from, for example, abstracts or author addresses. Field-specific indexes are not only an advantage when searching but also when the input module allows some kind of authority control (e.g., for index-assisted manual entry of new records). Being able to view (part of) the (field-specific) index (preferably showing the number of hits for each item) is essential for quality control of your databases. They make it feasible to systematically track down typos and alphabetically related notations. Some systems can also generate field-specific lists in (descending) order of frequency, so that you can easily spot the most popular authors, keywords, or journals in a database. Often, when field-specific indexes are supported, these are limited in number or cannot be defined by the user. Another type of field-specific index is not primarily retrieval-based, but can be used to generate authority lists to enhance standardization during import.

Not all fields are indexed in the same way. Most PBS offer more than one indexing technique, making a distinction between "text" and "nontext" fields. Text fields, such as "title" or "abstract," do not constitute a standard entity; therefore, each word is indexed individually. In nontext fields, the full contents are indexed as a coherent unit, so standard combinations, such as "journal name," "series title," or "corporate authors," are represented as one index entry. In this way, the composite search terms will be found in one step and need not be retrieved by combining their constituent parts. Some systems have a limited maximum length for index entries (even when the corresponding field contents have no such limits). For "text" fields, this is generally not a handicap, but "nontext" index entries often need several dozens of characters to uniquely identify them (e.g., long journal names or series titles, including subsections). Customizable stop lists can keep less meaningful words out of the index(es). It is even better when separate stop lists for each specific database or database type can be maintained. Special types of stop lists may be field-specific [e.g., to ignore prepositions in author names (De, La, Le, Van, Von, etc.)] during retrieval or ranging. In contrast to stop lists, some systems may opt for the opposite principle; only terms specified by a "go list" are included in the index(es). Another related mode of indexing avoids irrelevant terms by only selecting strings explicitly marked during input or editing. Other characteristics (repeatable versus nonrepeatable fields, alphanumeric versus numeric data, decimal versus integer ranging of numericals, author versus nonauthor fields, etc.) can also influence the way fields are indexed or sorted within the index.

Indexes are generally maintained using one of two basic methods: "real-time" updating brings the index(es) up to date immediately after modifying the database file, so they are fully reliable at any time. The disadvantage is that this can take some time, so that each time a record is created to changed the database cannot be accessed for a short while. This may take only a few seconds, but when you are used to millisecond microcomputer performance, this will quickly become a nuisance.

The second method consists of batch updating; the indexes are not updated until the user decides it is time to do so or has programmed the system to do it at fixed time intervals. No time is wasted while creating or editing records. The disadvantages are that the modified information is not incorporated in the index at that time, so the system should provide a way to overcome this setback. Some programs offer an additional sequential searching of that part of the database that differs from the index (e.g., the new or modified records). Updating the indexes may not always consists of just adding a small amount of information to the existing index, but may imply the generation of a completely new index. This can take quite a while, which means that the database is best indexed at times when it is not being used (e.g., overnight). Some PBS offer a choice between both immediate index updating and batch updating.

As both sequential and index-based retrievals have their advantages and disadvantages, the ideal, of course, is a combination of both: using indexes for default fast retrieval, and sequential searching for information that is normally not indexed or has not yet been (re)indexed. Some systems make trade-offs, basically searching sequentially except for a few fields for which indexes are maintained, or offer sequential searching for not indexed fields, without allowing fully sequential searches on all fields.

BASIC RETRIEVAL PATTERNS

Retrieval is basically a dialog between yourself and the PBS; you propose words or strings and the PBS tells you how often (and preferably in which fields) these words or strings are to be found within the database. Apart from the issue of sequential versus index-based retrieval, there is also a big difference in the way in which the various possibilities are presented or can be activated. Using formal commands has the advantage that you can indicate precisely what you want and often get in one step. The disadvantage, of course, is that you must first master the command language and know the available operators, field tags, and so forth. This is generally not the case with menu-based systems, which offer a choice between carefully explained alternatives at each step. The disadvantages is that they cannot always offer the same power and speed as formal commands do. Because of their step-by-step approach, they can become rather tedious for experienced users. The choice between both types is often a matter of personal preference. When more than one person uses the same PBS (e.g., both experienced staff and novice patrons in a library), the best option is a system offering a choice between both possibilities.

As indicated earlier, many PBS are fundamentally limiting; they assume that each new operation is meant to limit the current set. This is fine as long as you want to narrow your search, but it may not permit returning to previous steps to adjust the search formulation. In this case, the option to "select all records" is essential to start new searches.

Although this straightforward design will often suffice for its purposes, the setbuilding alternative can offer a more refined retrieval. In many traditional systems, each search formulation is executed on the complete database and its results constitute an autonomous set kept during the rest of the session. In this way, you can return to previously executed searches or combine them with later sets without the need to reactivate them. Backtracking is a comparable feature that allows you to go back one or several steps to previous screens, selections, commands, or menus. Also useful is the ability to retain the previous search formulation so that it can be modified without the need for retyping. Records marked during display should also be kept as autonomous sets that can later be combined with other sets or be printed or downloaded in their own right. It can also be useful to drop specific sets from the current session survey in order to get a clearer view of the search strategy used so far or to save active RAM memory. Being able to permanently save specific sets so that they can be recalled in later sessions is also helpful. This is especially so when it not only holds for sets of records but also for sets of commands or search histories so that the same combination of commands can be used in subsequent sessions. This is essential if you intend to use your PBS for SDI purposes. However, as with most aspects of retrieval and other modules as well, increased capabilities may come at the cost of ease of use.

REFINING RETRIEVAL

Retrieval options can be rather basic and limited, but a wide variety of extra possibilities exists—some rather evident, others pretty ingenious. A basic requirement, however, is the capacity to combine individual search terms (and previously defined sets). Generally, this implies the use of the Boolean operators AND, OR, and NOT. These may be freely applicable at the command prompt, selectable from (pop-up/pull-down) tables (explaining their meaning), or compulsory (which is helpful for novice users but may prevent some types of more advanced search formulations). Serious searching may involve quite intricate combinations with various sets of (nested) parentheses. A low maximum search expression length can be a basic limitation for building complex searches.

Most PBS offer the ability to limit the search to specific fields. Like the Boolean operators, this can be optional, fully command-driven, assisted by (pop-up) field tables or compulsory, being integrated in the search dialog. More refined "operators" include "begins with...," "ends with...," and the radical "field is (not) empty," Consultation of field-specific indexes showing the number of actual postings for each index entry is helpful and points out alphabetically related items, including their comparative popularity. Just seeing this information on the screen is helpful, but being able to select your search terms directly from these index extracts is still better. Some systems allow you to choose just one index term; others allow several, either activating each index term as a separate search set or combining them in an OR relation (or doing both). These facilities can be optional or the automatic system default, or a combination of both. In the best case, default retrieval fields can be customized (i.e., which fields are searched and in what order).

As explained earlier, not all fields are necessarily indexed in the same way. It certainly is an advantage if you can choose between text and nontext indexing and retrieval for some fields. Depending on your preferences, you can select records by either searching for the full field contents (e.g., "American Journal of Tropical Medicine and Hygiene" in the "journal name" field), which gives you the unambiguous result in one go, or by combining some of the constituent parts (e.g., "American" and "Tropical," both in the "journal name" field), which is more helpful when you do not know the exact name. However adequate the indexes may be, optional sequential searching (of the whole database or a specific set) can be a valuable extra option, even if just for case-specific verification.

Using lots of different fields can be interesting to guarantee highly sophisticated output formats and ultraspecific retrieval (e.g., first authors only), but sometimes it is more interesting to combine several related fields. For example, when you are looking for all publications by a specific individual, it is not important whether he or she is a first author, a secondary author, or a book editor. Some PBS offer the possibility of automatically combining similar fields under one field(group) tag "author" = "author1" + "author2" + "author3").

Proximity operators are more narrowing than the Boolean AND relation; they demand that two words must appear within the same field, the same paragraph, the same sentence, or a self-selected (exact or maximum) number of words separated, in either specified or indiscriminate order. Comparative searching allows you to select records by specifying that the contents of a field are equal to, larger than, or smaller than a certain value (e.g., publication date). Interval searching is especially valid for numerical data [values, years (e.g., "1993–1997)], but it can also be useful for alphabetical data, such as range of author names (e.g., "AU = smith-smythers" yields all authors, starting from the first Smith up to the last Smythers).

A special retrieval option is that of lateral searching; while viewing records, specific words (e.g., title words or keywords) can be marked, which are then automatically posted as new search terms. A more elegant option is when you can directly jump to the other (hyper)linked records in which this search term is featured (e.g., by simply clicking on them). In this way, you can navigate around the database. This is a very attractive feature when you want to look at, for instance, all available titles within a specific series when this series was not even a search term in the first place. This navigation is especially effective in split-screen mode. A comparable option is the immediate (short list) display of records featuring individual index terms when these (field-specific) indexes are browsed. However, if this cannot be combined with traditional retrieval techniques, it will not be easy to select, for instance, all French language journal articles authored by a specific individual and published between 1985 and 1995.

ALPHABET-RELATED DEVICES

It is obvious that while searching for specific words or strings, a number of alphabetically related items may be missed. This can be overcome by index browsing, which alerts you to the closest alternatives. An easier way is (implicit) truncation. Most systems offer a way to activate (or deactivate) right-hand truncation. For instance, "immun*" yields "immunity," "immunodeficiency," "immunology," and so forth. Left-hand truncation is less common and, if available, often only on a nontext field basis. For instance, "*diseases" yields "infectious diseases," "sexually transmitted diseases," and so forth. Full left truncation, in which "*ology" yielding "epidemiology," "immunology," and "parasitology," is rather rare in index-based systems. In sequential searches, however, this is no more difficult to execute than right-hand truncation. An extremely powerful variant of automatic truncation of nontext fields is sometimes referred to as "embedded wild cards." Only the first letter(s) of some of the constituents are needed to find (among others) the correct item. For instance, "j = j e m" (in which "j =" stands for the "journal name" field tag) finds "journal of electron microscopy," "journal of emergency medicine," and "journal or experimental medicine" (or a hypothetical "japanese electronics magazine" for that matter). A third possibility is internal truncation or masking, in which one or more characters within a word are replaced by a wild card. For instance, "h*matology" yields both "hematology" and "haematology." Truncation or masking symbols can replace either an exact number or just any number of characters, ranging from zero to a dozen or more. Stemming is a comparable feature, which includes related forms such as adverbial or conjugated forms, based on fixed or manually customizable lists or on more sophisticated artificial intelligence-based devices.

Some retrieval systems distinguish between uppercase and lowercase; others do not.

Ideally, you can choose whether to search in a case-specific way or not. This may be attractive to immediately distinguish terms that are always written in uppercase or with a peculiar mixture of uppercase and lowercase, but mostly the use of initial uppercase letters will be decided by whether this word appears at the beginning of a sentence or not. Just as it is interesting to combine several fields to one field tag, it is also helpful if you can specify a number of characters (including diacritics and foreign characters) to be searched together by default (yet be able to search them individually when necessary); for example, "a" yields "a," "à," "á," "ä," "â," and "å" (and their respective uppercase characters).

SUBJECT-RELATED RETRIEVAL DEVICES

Just as (field) indexes or other authority files (e.g., journal abbreviation lists) alert for alphabetically related terms, the thesaurus shows related subject terms (synonyms, related terms, higher-level terms, lower-level terms). These may function as optional suggestions or as an obligatory vocabulary police, rejecting all terms that are not explicitly defined in the thesaurus. Some powerful thesauri can automatically activate all lower-level terms when choosing a higher-level term ("explode"). For example, using the explicit keyword "africa" may yield only a small proportion of the relevant results. Using the explode feature, however, recall is much higher because records with keywords such as "nigeria", "rwanda" and "zimbabwe" are also selected, even though the word "africa" itself is not present in these records. However powerful a thesaurus may be, if the keywords are not selected in a consistent way, the thesaurus loses a lot of its functionality. During manual input, this may be minimalized by automatic thesaurus checking, but when importing records from different external databases, the chances that keywords are used consistently become quite low.

If an explicit thesaurus is lacking, the explode function can sometimes be emulated by defining a set of trigger terms; these automatically activate a number of self-specified search terms. These can be maintained using, for instance, a synonym list or a number of individually saved search strategies. Synonym lists need not necessarily include meaningful alternatives. They may include sound-alikes and look-alikes: words that sound like the target term but may differ considerably when written down, and vice versa. This may be especially helpful when the database contains scanned and OCRed records. Some PBS allows for more than one level of keyword; each primarily keyword can be combined with a number of secondary keywords acting like subheadings.

SOPHISTICATED RETRIEVAL TECHNIQUES

The retrieval techniques discussed so far are relatively straightforward and logical. In order to guarantee useful results, the search terms should conform to certain conditions (e.g., authors should alphabetically conform to an author index, and keywords should conform to the hierarchic thesaurus or synonym list). Certain highly sophisticated technique try to overcome these one-dimensional limitations in such a way that natural language can be adequately interpreted, so the information need not be entered with the rigid Boolean operators in mind. These techniques are often based on artificial intelligence techniques, using probabilistic relevance ranking, fuzzy set searching, and the like. This may imply special computer algorithms that automatically combine the search terms in an OR relation and count actual postings and compare relative positions. Others go one step further and find out which other words often appear in close proximity with the actual search terms and suggest or include these as extra search terms. Sound-alikes and look-alikes may also be activated on an artificial intelligence basis instead of an explicit (customizable) list.

These less common techniques can be quite useful when dealing with unstructured texts. When searching for structured bibliographic records, however, it is not unwise to keep Boolean operators and field-specific indexes as a basis. This basis may then be supplemented with these extra features, but in real like these do not always lead to fully orthodox results. Another essential drawback of natural language retrieval techniques is their language dependence; if they are successful, they may be so with one specific language only.

A special retrieval issue is that of multiple database searching. Some of the large online hosts offer the possibility of accessing several databases at the same time, thus reducing the need to search individual databases consecutively. They may also allow searching for specific terms from the database menu, in which case they show the number of hits in each database, so you know which databases resemble each other and use the same filed indicators and/or keywords systems. PBS are now also starting to offer some of these options.

Searching a PBS need not necessarily be limited to the actual databases records physically stored on the microcomputer or local network drive. Nowadays, retrieval may go way beyond these, automatically connecting with a plethora of World Wide Web resources (online databases, external library catalogs). Some offer modules based on client–server technology, which allows searching in external Z39.50-enabled databases while still using your own PBS interface. Others combine with external Z39.50-based retrieval programs (e.g., *BookWhere*). This client–server-based searching over the Internet may feature amazingly fast and diversified retrieval performances and may offer the highest degree of streamlining the importation of external records procedure by making it possible to just drag and drop records from one database to the other. As such, this also implies the incorporation of the previously separate modules for communication and online searching (e.g., *ProSearch,* the early twin of *ProCite*) and thus illustrates once more the general trend overall connectivity and integration.

Display

The immediate attraction of a PBS very much depends on the way the databases can be browsed and in what fashion search results are displayed. The least flexible PBS offer one fixed format (per record type), showing all fields included in the data definition tables, whether or not they are actually used in the record, and always in the same sequence. This type of "electronic formula" can be improved upon in several ways. First, empty fields to subsequent screens. Having the choice between different presentation formats is another advantage, and the options should be clearly indicated. Toggling between several formats is a boon, especially if the options include a (customizable) short list, showing up to 25 records per screen, each reduced to one line. Ideally, you can define which fields are displayed in what order. This default can then be overruled at any time.

While viewing, a number of scrolling possibilities should be allowed: go to the top or the bottom of a record (especially when more than one screen per record is involved); go to the next or previous record; or go to the beginning or the end of the current set. When displaying search results, hit terms should be highlighted so it is immediately clear why a record was selected. In some systems, this highlighting loses its power when previous sets are combined and the search terms themselves are not explicitly used in the final search formulation. It is also useful if you can quickly navigate toward these search terms within these records (e.g., with "next hit" or "previous hit" options or buttons). During record display, selection and deselection of specific items should be possible. Such records are best kept as autonomous sets that can later be combined with other sets or printed or downloaded in their own right.

Some PBS can only display the most recently created set; therefore, in order to view an earlier set, the corresponding search needs redoing. The option of displaying just any set ad hoc, using a mouse, cursor, or set number to select them, is far more elegant. Ideally, you can specify (a) specific (range of) record(s) to be displayed within a specific set. Another aspect is the sequence in which the records are displayed. Early systems tended to display the records in the order in which they were physically stored in the database. Although records can be sorted in a number of ways before they are printed or downloaded, this opportunity is not equally obvious when displaying them. However, this is an agreeable option and it adds to overall clarity if records can be viewed in a certain order [e.g., alphabetical (by author, or any other meaningful alphanumeric field such as "journal name") or in (reverse) chronological order (e.g., by publication date)]. Now many PBS offer several ways of arranging records before display, but these may be based on a limited sort key (e.g., only the first few characters), which only creates an approximate order. Ranging can be quite refined though, including several sort levels (e.g., year of publication, then authors, then title).

The flexibility of record display has gained a lot from the migration to the Windows95 platform with its split-screen features. Within different windows, a selection of several types of data can be displayed simultaneously [e.g., short list, full record, preview of the formatted record, index extract, command options (e.g., operators), and overview of search sets. As such, PBS now may resemble Web pages with their various frames. The presentation is quite dynamic, as you can resize these windows at any time and choose which fields to display in what order in the short lists. Within the short lists, you can range the records on a specific key just by clicking on the corresponding field label (but one key may give far faster results than others as it is already index-based, whereas the others need ad hoc sorting). The indexes displayed can double as quick search fields or authority files for data entry or searching.

The advantages of the ability to store and display images or graphics are discussed earlier. Next to the issues of linking or integrating techniques and (internal versus external) image viewers, there remains the question of flexibility and manipulability: Can the images be enlarged or reduced? Are thumbnails available? and so on.

Printing and Downloading (Output)

RECORD SELECTION

A first question to be addressed is that of record selection. Do you need to print or download a complete search set, or can you specify a subset using either (fixed) record numbers or relative positions within the set, indicate ranges (idem), or select records during display (marked records)? Compulsive limitation of the number of records that can be printed or downloaded can be a nuisance, but if they can be defined and customized by the systems manager, such limits can be advantageous for keeping public access output under control. Compared to printing, downloading has a few extra parameters to take into account. How much freedom do you get to specify output drive, path, and file name (including extensions)? When using the name of an already existing file, does the system alert you and allow the choice between appending the new records to the old file and overwriting the old file or renaming the new file?

RECORD FORMATTING

One of the essential advantages of putting bibliographic information in a number of separate fields is that you can reconstruct these records in many different ways. As discussed in the Introduction, the advantage is obvious when you consider the hundreds of divergent bibliographic styles that are being used by different scientific journals. With a good PBS, reformatting a complete bibliography is generally a matter of less than a minute. The most popular formats (e.g., *ANSI, Harvard, Science,* and *Vancouver*) should be provided with the program. If you are allowed to customize these formats or create new ones, the possibilities are virtually unlimited, and if you can freely construct new document types, this is an absolute must in order to be able to export them in a meaningful way.

For both the largely inflexible and the fully customizable systems, there exists a whole range of levels of detail. Some allow certain fields to be printed, without allowing any change to the field contents or their respective order. Others allow fields to be selected in any chosen order or allow conditional relations between fields (e.g., "if A then B, else C"). This can result in highly sophisticated nesting of instructions. Some allow formatting within individual fields (e.g., to define the number of words or characters for each field). This is especially important for adequate author formatting or short lists (e.g., using one line for each reference, restricting the length of each of the fields included to fit into self-specified columns). Some PBS go as far as to select what types of characters or how many of each should be printed; for instance, only upper case letters, only numericals, change all lowercase letters to uppercase, or remove all interpunction. Some formatting issues, such as absolute positioning use autonomous instead of relative values (e.g., to start printing field "a" at "position 25").

Some fields tend to inspire more attention of system designers than others. Special provisions for author fields include inversion of initials, removal or inclusion of blanks or periods, using "," "&," or "and" before the last author, conditional selection of the number of authors reproduced (e.g., when there are more than six authors, print only the first three plus the notion "et al."—as in the *Vancouver* style, popular in biomedical sciences), and so forth. Some PBS feature special modules that manage a number of alternatives for each journal name, such as full name, *Index Medicus* abbreviation (a de facto standard, but limited to biomedical sciences!), other (self-selected) abbreviations (e.g., regardless of which form is actually stored in the record's journal name field. Again, receiving such authoritative lists of alternatives with the system is a boon, but being able to modify or supplement it may remain essential. Certain output styles may feature their own idiosyncratic journal abbreviations that cannot always be foreseen in the PBS, and the PBS's formatting language. (Conditional) capitalization of such elements as author names or titles may be difficult to incorporate correctly. The same is true for cited page indication, where page "1134–1137" may need to be reformed to "1134–7" or simply truncated to "1134," Comparable routines to those of the journal name and journal abbreviation fields may also be offered for publication date fields, thus providing, for instance, different ranges and formats for year, month, and day values.

BIBLIOGRAPHY FORMATTING

Other issues concern the overall outlook of the formatted bibliography, such as ranging the references: sorting on several levels (freely choosing the fields involved), combining different sort directions (e.g., ascending versus descending: "A → Z" for one field, and "Z → A" for another), and page formatting (defining margins, indenting and numbering references [starting from 1, or using the (fixed) record numbers], providing customizable bibliography headings, and so forth). Again, foreign characters and diacritics can cause trouble if there is no provision for them. In some PBS, you can define a sort order integrating both standard and special characters. As with display, it can be useful for highlighting the hit terms, but this is generally easier for printing than for downloading purposes. Page formatting options may be integrated within the specific output styles or be systemwide. The first method is the most comfortable one if all the styles you need are provided for accurately, as you will not need to think about these intricacies when generating a reference list. If you have deviant document types, need extra output styles, or simply want to be able to experiment with these settings, systemwide options may be more suitable.

Of course, the more options that are available the more confusing the overall picture becomes. When all formatting instructions are fully integrated within the same output style, each variant calls for a separate combination, so the number of different styles quickly becomes overwhelming. Some PBS group the relevant parameters and output formats in a number of subsequent levels (e.g., basic record format, additional features, sorting order, page format, physical output device driver), so each level offers a limited amount of alternatives, but because each item can be combined with each item from each other level, a few possibilities on each level lead to a vast array of different output formats. Regardless of what fashion they are present in the worst-case scenario is when you do not know what output format to choose because you cannot know if any exist, as there is no listing of formats, and you can only use them when you have learned their exact names (and positions) by heart. WYSIWYG previews of printed output on the screen offer a last control and can avoid wasteful printing of incorrectly formatted reference lists. With non-Windows programs, the available specific printer support (fonts, drivers) may limit the eventual output.

Output does not always imply printing or downloading lists of bibliographic records. If index surveys can be produced easily, generating subject or author lists, featuring record numbers or full bibliographic descriptions linked with each keyword or author, becomes child's play. More essential is the possibility of exporting a complete database in a generally accepted interchange format, thus allowing the transfer of the database to another (newer and more versatile) PBS.

For the output modules, the Internet has inspired PBS producers to come up with new provisions. Not only do references to Web pages or E-mails need to be printed in an adequate fashion, but you may also need to provide bibliographies in HTML format to put them on your private or corporate Website or send search results by E-mail.

Word Processing

As discussed earlier, an output-related feature that has gained a lot of importance in recent years is the ability to combine the PBS's database functionality with word-processing programs. While writing the manuscript, some kind of pointer is inserted at the place of the in-text citation. This pointer can be a reference-specific anchor (e.g., the record number) or a retrieval-based key (typically an author–publication year combination). Generally, these pointers are inserted as the result of a genuine retrieval action that is launched from within the word processor, either while citing each record individually or in a group later. When ambiguity arises (e.g., when more than one publication in the database confirms to these search criteria), the more sophisticated systems will be able to deal with this (e.g., by offering a selection opportunity), and when necessary, they will provide multiple notations, such as 1995a and 1995b. When, finally, the bibliographic aspects of the manuscript need to be generated, the corresponding references are retrieved from the database and both the in-text citations and the reference list are formatted in a style appropriate for the specific target journal to which the manuscript is submitted.

There are difficult approaches to this kind of integration. In the best case, the PBS software, upon installation, adds extra database-related menu options to the toolbars of the word processor. The most current word processors, such as *Word* and *WordPerfect* are in this way provided for by Windows-based PBS. Older systems feature less tight integration, but even when such capabilities are rather poor, using Windows's clipboard or task-switching features may get you quite far (*9*). Even when there is no integration whatsoever, it may prove useful to generate the reference list with a PBS rather than typing it from scratch in the word processor, thus avoiding typos, formatting errors, and omissions–provided the records were entered correctly in the database and the right bibliographic output format is available and accurate.

Interface- and Management-Related Issues

Apart from the differences between the DOS and Windows platforms discussed earlier, there are still a number of other interface-related issues to be taken into account.

NETWORKING

Although PBS are essentially personal software programs, they can also be very useful for more than one user (e.g., smaller libraries, and research-unit-level-based workgroups). Obviously, not all microcomputer-based PBS are network products. In the best case, the system is fully network-compatible (preferably at no substantial extra cost), including full file and record locking. Several users can then access the same database simultaneously, but editing specific records is automatically limited to one user at a time to prevent data corruption. Other systems only work in a stand-alone situation, or provide networking facilities with non-DOS versions only (e.g., UNIX). In practice, however, many single-user systems can be used in microcomputer networks. Conflicts and data loss can be prevented by installing the DOS "Share" program, but this implies that only one user at a time can access a specific database. Also, this may constitute copyright violation (some programs allow this type of networking; many others explicitly do not) and offer no guarantee whatsoever that the system will not crash unexpectedly. Therefore, if you want to offer your database(s) to more than one (simultaneous) user, choosing a PBS awith full networking capabilities is a must. The possibility of mounting PBS databases on a Webserver and offering access to them worldwide is one of the more impressive new developments.

MODES, LANGUAGES, AND HELP

A preference for either menu-based or command-based modes is often a matter of personal taste. Having a choice between several modes is certainly an asset when dealing with a mixed public; expert users can use a no-nonsense command mode, which gives them the results quickly and accurately, whereas novice users can select a (often more limited) menu-based interface, carefully explaining the options at each step.

If workgroup users or library patrons represent different languages, a multilingual interface is quite an asset. Again, these come in various formats. The most sophisticated systems offer a choice between several dialog languages, featuring a full interface for each of these languages. Others come in separate versions one for each language. Still others are basically unilingual, but offer the possibility of including multilingual (help) messages. In this case, it is aboon if you can edit these messages yourself.

A good PBS should have clear and easily accessible help functions. These should either be continuously present on the screen (in shorthand form), or some shortcut indication should remain visible on the screen. In Windows interfaces, this tends to be standardized, whereas in DOS-based PBS, all kinds of help modules exist. Two other options can be viewed as extensions of the help functions. An indication of the progress being made during time-consuming actions (e.g., while searching or downloading) can reassure a user that the system has not blocked, or make him or her decide to cancel the operation—if this option is available. In the same fashion, error messages, if pertinent, are useful extensions of the help functions.

Being able to visit DOS without leaving the active database was also helpful feature. In this way (if the basic DOS memory allows it), you could activate functions that were not available in the PBS software itself (e.g., by using simple DOS batch files to activate multilingual help messages). In the now current Windows interfaces, this task swapping is, of course, a standard possibility. It is also an advantage if your system is based on one of the more popular programming languages, as built-in routines are often available or can be developed by the user.

From an ergonomical viewpoint, being able to change the screen colors (both foreground and background) is not a superfluous luxury, especially if you have to spend long hours accessing your databases, or when hit terms or otherwise foregrounded words or fields are highlighted and you dispose of a monochrome display unit only.

SAFETY AND SECURITY

Some systems are sensitive to specific strings or keyboard combinations. In the best case, the retrieval session is closed. In the worst case, an automatic deinstallation program is activated. Prompting for confirmation when (partially) deleting sets of records or search histories, changing databases, or leaving the system can prevent accidental loss of the results of considerable efforts. Unexpected power failure generally does not corrupt the data file. If it does, installing an uninterrupted power supply (UPS) may be the only safe solution.

As is clear from the previous paragraphs, password protection can be important if more than one user has access to the system or you want to prevent others from (ab)using your personal databases. Even with microcomputer-based PBS, password protection can be quite sophisticated, including various levels of authorization. It is feasible that although many people use the same database, certain information (e.g., specific fields) may only be accessed by a specific public, part of which is also allowed to enter new records or change existing ones. Still other functions, such as creating or deleting entire databases, should be accessible to the system administrator(s) only. Comparably, a disadvantage of the extramural options discussed above is that users can use them to circumvent a dedicated PBS installation and roam freely around your microcomputer or network, possibly tampering with parameters or deleting whole databases or systems. Where this is an issue, special "locking" software may be a solution.

FEEDBACK

The automatic generation of usage statistics is also an interesting feature in multiuser environments such as libraries. In this way, you get to know what (category of) patrons consulted what databases, when (date, hour), for how long, and how many searches were performed or records displayed, printed, and downloaded for each. This is not to suggest that a relatively simple PBS should in this respect do a better job than some full-scale integrated library systems featuring highly sophisticated transaction logging modules, but it certainly is an advantage if your low-cost PBS can do something workable in this direction. Another type of feedback is more data-related (e.g., the availability of accessing such statistical information as the total number of records available in each database, and the number of individual author or keyword postings).

Conclusion

Personal Bibliographic Systems constitute a highly prolific microcomputer software, related to, but using a basically different approach from general-purpose database management systems or traditional electronic library catalogs. Within their own niche, PBS come in all sorts and varieties and can differ strongly in basic concept. The various individual features reviewed in this article together offer an enormous potential of attractive characteristics. When looking at individual systems, however, it is obvious that none has all the aces. Each package has its own strong and weak points; you will generally be impressed by certain features, but you will equally find it a pity that other aspects are not dealt with as ingeniously as in some of its rival systems. Also, some impressive bells and whistles may hide fundamental shortcomings. Moreover, the usefulness of the separate features will be evaluated differently by individual users, either for personal or for professional (e.g., in a library) reasons. Instead of advertising one specific PBS, this survey points out the major pros andcons (or even the very existence) of the various features that may be encountered in this type of software. Potential users should define their own lists of preferences and priorities and then select the real-life PBS that offers the best compromise. Such systems are reviewed adequately in the surveys included in the Reference section. Nowadays, the more popular producers have informative Websites, although these may be strongly marketing-biased and claims of inherent superiority may need to be taken with a pinch of salt. On the other hand, these Websites may offer freely downloadable demo-versions so you can check their claims for yourself. Apart from these, the Internet features dozens of discussion lists concerning PBS. Their postings may be very informative but also biased in both positive and negative ways.

There are three final aspects that you should bear in mind when selecting a PBS.

1. Is your microcomputer powerful enough to fully exploit the possibilities of the system (and if not, do scaled-down versions exist, which are less memory and disk-space hungry)? Inexpensive read-only versions may be very useful when you want to distribute your databases, offering recipients full search and output capabilities.
2. Is the provider sufficiently reliable and supporting? How long has it and its system been in this branch of the software business (compare program version number)? Are (freely downloadable) demo-versions available? How useful are the (printed or electronic) manuals? What kind of support (nearby representatives? toll-free telephone number? informative Website?) or training can you get at what price? Obviously, the Internet has dramatically enhanced these communication-based issues.
3. Can you afford it? Is the package worth the investment, both in money and in time and effort (learning, installing, customizing, etc.), compared to others? Simpler and less expensive systems might fulfill your needs just as well—but then again, they might not.

APPENDIX 1: A GENERIC SURVEY OF PBS FEATURES*

Structural Flexibility
Maximum number of databases in the system
Maximum length of database names
Maximum number of records per database
Maximum number of bytes per record
Structured fields versus unstructured text
Maximum number of different document types
Predefined document types available
Free definition of document types (all versus some versus specific)
Maximum length of document type names
Internet-based document types (HTML pages, E-mails)
Maximum number of fields per record
Free field definition (all versus some versus specific)
Field order (fixed versus free)
Maximum field length (idem)
Variable field length (idem)
Repeatable fields (idem, limited versus unlimited occurrences)
Subfields (free versus fixed; all versus some versus specific)
Free creation of field names and tags (idem)
Mnemonic field names and tags (idem)
Maximum length of field names and tags
Full ASCII set, including diacritics and adequate symbols
Highlighting: bold, italic, underline, superscript, subscript
Active ("clickable") URL fields
Change existing structures

- Add new fields/document types
- Delete fields/document types
- Change (rename) fields/document types

Record (identification) number (free versus fixed)
Renumber records (individual versus all); change renumbering increments
Automatic generation of dates (creation)
Linking records (e.g., parent–child)
Incorporate multimedia: images, sound (links to external files versus fully integrated)
Incorporate bar codes

Database Selection
Database menu available
Length of database names
Fixed versus customizable database presentation (e.g., order of databases)
Easy database selection from menu
Gateway to alternative directories
Define default database to be opened
Input
Electronic input sheet
Systemwide versus document-type specific
Customizable sheet
Line editor versus full-text editor
Extended ASCII accepted

*The desirability and (dis)advantages of each of these features are discussed in the main text.

Accents and diacritics module
Copy, cut, and paste capabilities
Duplication of field contents (all versus some versus specific fields)
Duplication of field contents (idem)
Automatic addition of specific field contents
Duplication of existing records
Index-assisted entry (all versus some versus specific fields)
Authority control (idem; automatic versus optional)
Input validation tables
Compulsory fields
Generic data formats accepted (which)
Records from other PBS accepted (which)
Web pages accepted from browser (complete contents versus selection versus URL and title only)
Internal conversion profiles

- Availability: integrated versus separate "group" module versus individually marketed
- Quantity: how many source formats (which)?
- Representativity: Are the source formats you need included?
- Quality: How accurate is the result?
- Flexibility: Can the profiles be modified?
- Expandability: Can you create new profiles?

Automatic duplicate detection (real time versus batch)
Automatic alert for new field contents (e.g., keywords)
Merging databases (keeping original record numbers)

Editing
Integrated in display versus separate module
Line editor versus screen editor
Internal editor versus free choice of external editor
Copy, cut and paste capabilities
Global editing

- Maximum number of records per session
- Field-specific versus full text
- Case-specific versus case independent

Delete records

- Logical versus physical (free disk space)
- Real-time index modification versus batch

Indexes
Indexed versus sequential retrieval
Real-time indexing versus batch indexing
Index-size compared to database file
Individual indexes for all fields versus basic index
Individual indexes for some fields (fixed versus free)
Maximum index-term length
Individual words ("text") versus full-field contents ("nontext") versus selective indexing
Numeric versus alphanumeric indexing
Integer versus decimal ranging of numericals
Indexing of numericals: decimals versus integer ranging
Support personal name fields (author formatting)
Stoplists

- Customizable versus fixed
- Maximum length
- Systemwide versus database-specific versus field-specific

Go lists (idem)
Alphabetical index survey available (per field)
Frequency index survey available (per field)

Retrieval
Sequential versus indexed searching as default
Sequential searching as additional option
Selective indexes for "quick searches"
Response times for single searches
Response times for combined searches
Autonomous sets versus limiting retrieval
Individual set creation
Search history survey
Backtracking
Marked records as individual sets
Older sets can be reused
Older set can be dropped ("clear")
Save sets
Save search formulations
Survey of saved sets and search formulations
Boolean operators

- AND
- OR
- NOT

Combine several operators (use nested parentheses)
Maximum length of search formulation
Proximity searching

- Within the same field
- Within the same paragraph
- A maximal number of words apart
- A specific number of words apart
- Order (in)dependence

Comparative searching

- Equals
- Is larger than (or equals)
- Is smaller than (or equal)

Interval searching

- Numerical
- Alphabetical

Field specific retrieval
Limitations: begins with, ends with, field is (not) empty, etc.
Default retrieval fields (fixed versus free)
Search multiple fields
Retrieval using index extracts
Idem, including display of the number of hits (and occurrences)
Idem: number of index terms that can be selected simultaneously
Truncation right (exact versus minimal)
Truncation left (idem)
Masking (+internal truncation; idem)
Stemming
Case-independent versus exact match

Combine several fields with one retrieval code
Combine several diacritics with one retrieval character
Lateral searching
Navigation ("hypertext")
Structured keywords (several levels)
Hierarchic thesaurus
Automatic mapping to thesaurus terms
Explode capabilities
Trigger terms (fixed versus customizable)
Synonyms list (fixed versus customizable)
Natural language interpretation
Relevance ranking
Sound-alikes or look-alikes
Multiple database searching (including automatic deduplication)
External database searching (Z39.50-based)

Display
Display any set versus last set only
Display any specific (range of) record(s)
Short list versus full record display

- Separate screens
- Split screens (fixed versus self-definable)
- Toggle
- Sort on specific fields (fixed versus self-definable)

Scrollable record display
Display only nonempty fields
Hide specific fields
Customize display fields
Customize display fields order
Change fields or sort order ad hoc
Range records for display
Field tags (length, customizability)
Use different fonts, WYSIWYG, etc.
Foregrounding (bold, underlined, italics, etc.)
Highlight hit terms
Search next/previous hit term
Mark hits for (de)selection
Display images, graphics

- Zoom in/zoom out
- Thumbnails

Printing and Downloading
Selection of records to be printed or downloaded

- Complete set
- Records marked during display
- Specific (record) numbers
- Range of (record) numbers

Printing and downloading limits (fixed versus customizable)
Free choice of downloading station, path, and file name
Append versus overwrite option
Different output formats

- Availability: integrated versus separate "group" module versus individually marketed
- Representativity: Are the sources you need included?

- Quantity: How many "styles"?
- Quality: How accurate is the result?
- Flexibility: Can they be modified?
- Expandability: Can you create new ones?

Generic exchange formats (which)
PBS-compatible formats (which)
Internet-based output: format HTML bibliographies
E-mail results
Select

- Fields
- Number of words
- Number of characters
- Types of characters
- Include (fixed) record number

Specific field formatting features

- Author formatting [e.g., invert initials, (conditional) selection of author numbers]
- Journal formatting (e.g., full title versus abbreviations; blanks versus periods)
- Date formatting

Conditional instructions (if . . . then . . . else . . .)
Nesting of instructions
Ranging diacritics (fixed verus customizable)
Numbering references (independent versus fixed record number)
Indention of records (fixed versus customizable)
Positioning of characters

- Horizontal verus vertical
- Absolute versus relative

Page formatting

- Margins (horizontal, vertical)
- Heading
- Numbering pages
- Style-specific versus ad hoc

Easy selection of available formats and parameters
(Multiple) Device drivers [screen versus printer type (how many, which) versus ASCII file versus word processing file (how many, which), etc.]
Preview onscreen
Output to general interchange format
Automatic bibliography generation from word processing

- Degree of integration
- Which programs: Word, WordPerfect, others

Interface-Related Issues
Operating system

- DOS
- Windows
- Macintosh
- Others (which)

Navigation devices

- Cursor
- Function keys
- (<Alt>-or drop-down/pop-up) menus
- Mouse

Multilingual interface

- Multiple language versions
- Fully integrated

(Multiple) Mode(s)

- Easy—menu
- Expert—command mode
- Query by form versus intuitive navigation

Help functions

- Separate versus integrated
- General versus context-specific
- DOS escape
- Compatible programming language
- Display the progress of (time-consuming) actions (retrieval, printing, downloading)
- Possibility to cancel (time-consuming) actions (retrieval, printing, downloading)
- Error message (relevant)
- Customizable screen colors

Security

- Unconventional key combinations for delete or exit operations
- Prompt for confirmation for delete or exit operations
- Sensitivity to power failure
- Password protection: at one level versus at several levels (e.g., specific functions or fields)

Usage feedback (statistics)

- Date
- Hour
- Total time
- User(group)s
- Database
- Number of records retrieved
- Number of records displayed
- Number of records printed
- Number of records downloaded

Database contents related statistics (number of records, index items, magnitude, etc.)
Networking

- Full networkability, including record locking
- Limited capacities (e.g., file locking)
- Strictly single user

Linking with external systems; Z39.50 provisions
Mounting PBS databases on a Webserver

Miscellaneous
Cost (price of PBS versus "learning' 'time)
Support (representatives, telephone, Internet)
Training (availability, distance, price)
Manuals (clarity, relevance, index)
Upgrading policy scaled-down versions
Demo-versions (free, downloadable)
Inexpensive read-only version for file distribution
Memory requirements (RAM, disk space)

APPENDIX 2: A LIST OF INDIVIDUAL PBS

This list includes the names of programs that can or have been considered to be (partially) PBS. Not all of them feature all of the major PBS characteristics. For instance, some of them may not format bibliographies, but they are admirable in their capacity for storing and retrieving bibliographic records. Some are distributed by well-organized companies and have sold tens of thousands of copies, whereas others may rather be hobbylike projects of creative individuals. Some of them have been dead for many years. This list includes a number of programs the author has personal experience with, although many others were found in the literature only. Eric Sieverts et al. (*1–7, 11*) and Sue Stigleman (*9, 12–14*) especially have extensively evaluated many of these PBS. Different program version or variants for different platforms (e.g., "for Windows") are represented by a single mention. This list is meant only to illustrate the diversity of this software niche and does not intend to claim any degree of completeness or quality indication.

Adlib
Archivist
Artfile
Articles
Author-Title
Autobiblio
bib
bibIX
Bibl
Biblio
BiblioFile
Bibliog
Bibliographer
Bibliography Generator
Bibliography Maker
Bibliography Manager
Bibliography Writer
BiblioStax
Bibliogic
Bib/Rite
BiB/SEARCH
BibTeX
BIBTOOLS
Bookends (Pro)
BookMinder
BRS-Search
Cardbox (Plus)
CDS/ISIS
Citation
Citation for WordPerfect
CITES
Concordance
DB/Textworks
DBF-Papers
dms4-cite
Edibase
Endnote (Plus)
File-It
Find-it Quick
Freebase (Prof)
Get-A-Ref
Headfast

History Database
HyperBib Tex
IdeaList
InfoBank
Inmagic
Interactive Bibliography
Interactive Reference Inventory
JEPRS (JEEPERS)
Journalog
KAware
Keylibrarian
Lars II
Library/Literature
Database Manager
Library Master
Library Mate
Lit
Literary Reference Inventory
MacDewey
Magazine Article Library
Mainbib
Manuscript Manager
Medilib
MEDLIB
Micro-OPC
Norman
Nota Bene
Notebook II (Plus)
Nutshell (Plus)
Paperbase (De Luxe)
Paperfinder
Papers
Papyrus
Personal Bibliographic
Reference
Personal Library Manager
Personal Reference Catalog
PFS 2000
(Mikro) Polydoc

Pro-Cite
Publish or Perish
Q&A
Quicksite
Quicksearch Librarian
Ref-11
RefBase
RefBase0DB
RefCard
Ref-Ed
Refer
Reference File
Reference Gateway
Reference Management System
Reference Manager
RefFiler
REFLIST
RefMaker
Refmenu
REFS
Refsys Super
Research Assistant
Research Notes System
Resnoter
Sapana Cardfile
Scientific Reference System
Scholars Librarian
Sci-Mate
Searchlit
STN-PFS
Strix
Subject List
Sysbib
Thesys
tib
TinMan (TinLib...)
3by5
25;02
VCH Biblio
WinRefer
WP Citation

REFERENCES

The references listed are highly relevant for the issues discussed in this article. Reviews discussing the pros and cons of individual PBS are not included.

1. E. G. Sieverts and M. Hofstede, "Software for Information Storage and Retrieval Tested, Evaluated and Compared, Part 1: General Introduction." *Electronic Libr.,* **9**, 145–153 (1991).
2. E. G. Sieverts, M. Hofstede, P. H. Haak, P. Nieuwenhuysen, G. A. M. Scheepsma, L. Veeger, and G. C. Vis, "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 2: Classical Retrieval Systems." *Electronic Libr.,* **9**, 301–316 (1991).
3. E. G. Sieverts, J. Figdor, S. Bakker, and M. Hofstede, "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 3: Enduser Software." *Electronic Libr.,* **10**, 5–18 (1992).
4. E. G. Sieverts, M. Hofstede, and B. Oude Groeniger, "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 4: Indexing and Full-Text Retrieval Programs." *Electronic Libr.,* **10**, 195–207 (1992).
5. E. G. Sieverts, M. Hofstede, G. Lobbestael, B. Oude Groeniger, F. Provost, and P. Sikovà. "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 5: Personal Information Managers, Hypertext and Relevance Ranking Programs." *Electronic Libr.,* **10**, 339–355 (1992).
6. E. G. Sieverts, M. Hofstede, A. Nieuwland, C. Groeneveld, and B. De Zwart, "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 6: Various Additional Programs." *Electronic Libr.,* **11**, 73–89 (1993).
7. E. G. Sieverts and M. Hofstede, "Software for Information Storage and Retrieval Tested, Evaluated and Compared. Part 7: What to Choose, or the Purpose of It All." *Electronic Libr.,* **12**, 21–27 (1994).
8. E. Sieverts, "End-User Software," in *Encyclopedia of Library and Information Science,* vol. 57, supplement 20, Marcel Dekker, New York, 1996, pp. 154–175.
9. S. Stigleman, "Bibliography Programs Do Windows." *Database,* **19**, 57–66 (April 1996).
10. G. Vâlas, "Comparison of Some Widespread CD-ROM Information Retrieval Software Packages." *Online CDROM Rev.,* **18**, 211–226 (1994).
11. E. Sieverts, ed., "*Text retrieval software: een vergelijking van bijna 50 retrieval-programma's, aangevuld met gegevens over thesaurus-software en conversie-programma's,*" Nederlandse Vereniging van Gebruikers van Online Informatiesystemen (VOGIN), 's Gravenhage, 1996. [This is a thorough discussion of both the general topic and some 50 individual information storage and retrieval programs (not all of them PBS). This study is published in Dutch only.]
12. S. Stigleman, "Bibliographic Formatting Software: A Buying Guide." *Database,* **15**, 15–27 (Feb. 1992).
13. S. Stigleman, "Bibliographic Formatting Software: An Update." *Database,* **16**, 24–37 (Feb. 1993).
14. S. Stigleman, "Bibliographic Formatting Software: An Updated Buying Guide for 1994." *Database,* **17**, 53–65 (Dec. 1994).
15. T. Hanson, ed., *Bibliographic Software and the Electronic Library,* University of Hertfordshire Press, Hertfordshire, UK, 1995.
16. R. G. Jones, "Personal Computer Software for Handling References from CD-ROM and Mainframe Sources for Scientific and Medical Reports." *Br. Med. J.,* **307**, 180–184 (1993).
17. J. A. Kelly, "Downloading Information Using Bibliographic Management Software. End-User Software." in *Encyclopedia of Library and Information Science,* vol. 59, supplement 22, Marcel Dekker, New York, 1997, pp. 111–119.
18. M. C. Miller, "Reference Management Software: A Review of Endnote Plus, Reference Manager, and Pro-Cite." *MD Comput.,* **11**, *161–168 (1994).*
19. L. H. Nicoll, T. H. Quellette, D. C. Bird, J. Harper, and J. Kelley, "Bibliography Database Managers; A Comparative Review." *Computers Nursing,* **14**, 45–56 (1996).
20. P. Nieuwenhuysen, "Criteria for the Evaluation of Text Storage and Retrieval Software." *Electronic Libr.,* **6**, 160–166 (1988).
21. R. Rabinovitz, "Point Reference." *PC Mag.,* **12**, 269–279 (1993).
22. D. Schoonbaert, "Personal Bibliographic Systems (PBS) for the PC: A Generic Survey of Features." *Electronic Libr.,* **15**, 31–46 (1997).
23. D. Slee, "Electrocopying from Databases," in *Bibliographic Software and the Electronic Library,* T. Hanson, ed. Universtiy of Hertfordshire Press, Hertfordshire, UK, 1995, pp. 84–94.

24. C. Oppenheim, "Staying Within the Law," in *Bibliographic Software and the Electronic Library,* T. Hanson, ed. University of Hertfordshire Press, Hertfordshire, UK, 1995, pp. 95–107.
25. A. Carter, "A Trial of PC Bibliographic Databases and Formatting Packages," in *Bibliographic Software and the Electronic Library,* T. Hanson, ed. University of Hertfordshire Press, Hertfordshire, UK, 1995, pp. 70–83.

DIRK SCHOONBAERT

THE PRESERVATION OF PAPER

Paper Preservation

Excellent paper preservation involves a combination of factors. These include assessment of the materials to be preserved and the facilities in which they are to be stored. It also requires excellent planning and execution of preservation activities. Finally, the acquisition and development of the human and physical resources needed to do the job properly are critical. The best materials, facilities, and policies will not work if the people assigned to their care lack the ability and/or motivation to carry out their assigned tasks. A motivated, intelligent, and well-trained staff is therefore required to manage the paper preservation process if optimum effectiveness is desired.

Assessments require evaluation of a diverse set of issues, materials, and facilities. They include an evaluation of the following:

- *The types of papers housed in a given facility, their composition, and their required performance characteristics.* This applies both to freshly produced papers as well as to their ability to retain performance capability over time.
- *The physical construction of the sties in which paper is to be stored and the environmental control of the interior spaces.* To maintain good paper preservation, these facilities require detailed specification of their construction and sufficient financial resources to meet the standards.
- *Policies and practices for intake.* Management and use of paper materials require definition and can have significant bearing on paper's life expectancy.
- *Roles and accountabilities for the persons charged with upholding sound preservation policies and practices.*

The development of enduring policies and the excellence of their execution include consideration.of the following:

- An overall preservation policy for the site or institution.
- Clearly articulated strategies for carrying out policy and for its continual upgrade as new information becomes available.
- Elements of policy include:
  - Definition of the required design and construction of storage facilities and spaces.