40 Personal Bibliographic Systems (PBS)

Dirk Schoonbaert and Victor Rosenberg

CONTENTS

INTRODUCTION

History of the Software Genre

Before personal computers appeared in the late 1970s there were a few attempts to write software that would organize and punctuate bibliographic references. As the Apple II and the IBM PC began to proliferate in the early 1980s one of the obvious applications for these machines was the software

category that we call bibliographic software. Two major products were developed almost simultaneously, Reference Manager, and the Personal Bibliographic System (PBS), later renamed ProCite. Reference Manager was the brainchild of a researcher in hematology, Ernest Beutler, and ProCite was developed by Victor Rosenberg at the School of Library Science at the University of Michigan. A somewhat later entry into the market was EndNote, developed by Rich Niles. All three of these products were subsequently purchased by Thomson Scientific (http://www.isiresearchsoft.com) and all are still sold by them today. In addition, Thompson Scientific has developed two Web products, RefViz and EndNote Web.

Earl Beutler, Ernest's son, commercialized his father's idea and after selling his company to Thomson in the mid-1990s, has been instrumental in launching a new and popular bibliographic software product, RefWorks. RefWorks is now owned by CSA. A number of other software developers created bibliographic products, including Library Master, Papyrus, Nota Bene, BibTex. Library Master is available from Balboa Software of Phoenix, Arizona (http://www.balboa-software.com).

The vision for bibliographic software was a personal program that mirrored the larger bibliographic databases. This vision called for the researcher to be able to retrieve citations from many sources such as subject-specific databases or library catalogs. For example, a researcher could find and then download references from PubMed, Web of Science, Chemical Abstracts, as well as from the Library of Congress Catalog, or university library catalogs from around the world. The major obstacle to realizing this vision was that the record formats for the various bibliographic resources were all different. This problem has been largely solved by the combined efforts of the personal software developers and the efforts of the publishers of the major databases and the library automation vendors. Most major library catalogs and databases now provide a format that the major personal bibliographic products can read. This was not always so, and great effort went into creating interface software that would translate records from the large resources into the personal system.

The vision further included the idea that the researcher would want to collect references into one personal database from which he or she could retrieve the reference into a manuscript in any of the various formats that publishers require. The researcher could include personal notes in the bibliographic record to help in creating the manuscript. As technology developed and the capacity of personal computer databases increased, it became possible to include entire papers as part of the bibliographic record.

It was important for the personal system to have all the functionality of the larger institutional systems. The user of the system had to be able to organize the references in various ways, alphabetically by author, title, or subject. The user had to be able to index the data, or to retrieve information by words in the record. In addition, the user had to be able to add the references to a document in the word processor in all the various formats that different publishers demand.

Early on it became clear that users wanted not only a program to format bibliographies in a word processor, but a complete database management system specifically designed for bibliographic records. The common DBMS programs like Access or Filemaker simply would not do for bibliographic applications. Most of those involved in developing the software kept wondering when Microsoft or the publishers of WordPerfect would simply include a bibliography manager as a new feature of their program. Soon it became obvious that the complexity of the bibliographic program rivaled that of the word processor itself. Then again, PBS have become a sort of mainstream software, as they are increasingly being included in university education, a practical subject taught by librarians.

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 but that would do credit to any PBS. As both the advantages and disadvantages of individual features are

not always immediately obvious, this discussion may prove helpful when shopping for a PBS. 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. For a discussion of the major PBS themselves, we refer the reader to a number of excellent review articles. Schoonbaert (2002) lists over 100 different PBS. Although many systems have become obsolete by now, several others are flowering, releasing new major upgrades every few years.

DOCUMENT TYPES

When dealing with bibliographic references, it is essential that the system handle highly structured data. 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" toolboxes at the disposal of creative enthusiasts willing and able to spend sometime 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, and Internet-based formats such as electronic mail messages or Web sites, with clickable URLs. 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 an 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., unpublished documents or annual reports), your system must allow for more than standard book or journal article formats. Even if abundant arrays 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 and 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 non-bibliographic data). Of course, flexibility of structure can only be enjoyed if it is matched by an equal flexibility of display and output formats. It does not make 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-defined formats (e.g., journal output styles) may no longer give correct results unless they are doctored also. On the other hand, using only the default-predefined modules offers a 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 some systems, there is a limit to field lengths, but this limit can be several thousand bytes,

so it generally poses no practical problems. Some PBS offer a compromise with hybrid types featuring a number of fields of limited length, coupled with one or a few of long length.

Not all systems are equally user-friendly in exploiting their field structures. Many use short alphabetic tags of one or a few characters. They 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 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 by automatically providing full-field names for display purposes (e.g., "T" is transformed into "TITLE:").

CHARACTERS

For bibliographic purposes, it is essential that all necessary characters be adequately supported by the system. Next to the 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 most current PBS support extended ASCII and even Unicode, so the most prevalent foreign texts can be accommodated. 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 for 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, 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. 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. Another type of linking is that to records in other (more elaborate) databases (e.g., PubMed or ISI Web of Science) or to full-text sources (often PDF files), either on the local system or on an external server (e.g., PubMed Central).

DATABASE SELECTION

Database selection is 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. Having access to different database directories can be

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.

ENTERING NEW RECORDS (INPUT)

Manual Input

Keying bibliographic references in a database is a tedious chore. Any help to ease 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 sheets 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 in a default input sheet, so that when necessary, less common fields can also be added.

Input models generally make use of some kind of text editor. These may differ greatly in versatility and power. 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. 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, users may be able to use the standard clipboard and copy/cut/paste or drag and drop techniques. Another useful feature is the option to specify default content for specific fields (e.g., journal or book title) or automatically add field occurrences (e.g., keywords common to a retrieval results set) 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).

Index-assisted input, where the strings typed are compared to information that is already available within the indexes, has two major advantages. First, it guarantees alphabetic consistency because the indexes act as authority files. Second, if automatic truncation is incorporated, this can dramatically reduce 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 name 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, yet it should equally be possible to overrule this feature or temporarily neutralize it, otherwise no new index terms can be added. Although indexes 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 or more 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. Input-validation tables, defining what type of data (alphabetic, numeric, alphanumeric, etc.) are allowed in what field or whether certain compulsory fields cannot be left empty may further optimize data integrity and decrease flexibility.

Importing External Records

Ever more electronic records are becoming available from Internet databases. When these can easily be imported into the PBS, adding new records to the system almost becomes a zero-effort action, afterwards offering the advantages of fast retrieval and flexible reformatting. Many database providers are offering their data for downloading in a recyclable format.

On a technical level, the importing of electronic records is only possible if the import files 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 online 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. 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, 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 create new ones, tailored to your specific needs. Originally, these reformatting profiles were separate modules distributed individually or in group(s). Current PBS tend to integrate these often abundant profiles within the main program.

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 deciding whether two records are considered duplicates, or are prompted to judge each suspected 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 import, yet there may be mutually incompatible field or document-type definitions between two such databases. Merging complete databases is a more complex matter, as it may be necessary to avoid duplicates, 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: using only relevant (i.e., non-empty) 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 and the corresponding qualitative capacities, such as full-record modifications versus only within specific fields, and case-specific versus case-independent. Having the edit functionality available at all times is more comfortable than when it constitutes an individual module to be accessed separately. Some systems allow external editors to be used instead of the standard one provided within the system.

SEARCHING THE DATABASE (RETRIEVAL)

Index-Based Retrieval

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 abstracts or author addresses. Field-specific indexes are not only an advantage when searching but also when the input module allows index-assisted authority control. Being able to view (part of the [field-specific] index [preferably showing the number of hits for each item]) is essential for quality control. They make it feasible to systematically track down typos and alphabetically related notions. Some systems can also generate field-specific lists in descending order of frequency, so 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.

Not all fields are indexed the same way. Most PBS offer more than one indexing technique, making a distinction between "text" and "non-text" fields. In text fields, such as "title" or "abstract," each word is indexed separately. In non-text 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 "non-text" 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 indexes. 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 of ranging. In contrast to stop lists, some systems opt for the opposite principle; only terms specified by a go list are included in the indexes. Another related mode of indexing avoids irrelevant terms by selecting only strings explicitly marked during input or editing. Other characteristics (repeatable versus non-repeatable fields, alphanumeric versus numeric data, decimal versus integers, author versus non-author 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 database up to date immediately after modifying records, so they are fully reliable at any time. 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 disadvantage is that the modified information is not immediately incorporated in the index. 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).

Basic Retrieval Patterns

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 set building 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 needing 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 for display should also be kept as autonomous sets to 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. Being able to permanently save specific sets so they can be recalled in later sessions is also helpful. This is especially true 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.

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 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 or compulsory, which is helpful for novice users but may preclude some types of more advanced search formulations. Serious searching may involve quite intricate combinations with various sets of (nested) parentheses.

Most PBS offer the ability to limit the search to specific fields. Like the Boolean operators, this can be optional, assisted by 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 can be optional or the system default, or a combination. 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 non-text 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, or by combing some of the constituent parts (e.g., "American" and "Tropical," both in the journal name field), which is helpful when you do not know the exact name.

Using lots of different fields can guarantee highly sophisticated output formats and ultra-specific retrieval (e.g., first authors only), but sometimes it is more helpful 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 this possibility of implicitly combining similar fields under one field group tag.

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 requesting 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, but it can also be useful for alphabetical data, such as a range of author names.

A special retrieval option is that of lateral searching; while viewing records, specific strings (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.

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, "immune*" yields "immunity," "immunodeficiency," "immunology," and so forth. Left-hand truncation is less common and, if available, often only on a non-text 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 should be no more difficult to execute than right-hand truncation. A powerful variant of automatic truncation of non-text 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 "journal name") finds "journal of electron microscopy," "journal of emergency medicine," and "journal of experimental medicine." A third possibility is internal truncation or masking, in which one or more characters within a word are replaced by a wildcard. 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).

Subject-Related Retrieval Devices

Just as (field) indexes and other authority files alert for alphabetically related terms, the thesaurus shows related subject terms (synonyms, related, higher- and 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 ("explode"). For example, using the keyword "Africa" may yield only a small proportion of the relevant results. Using the explode feature, however, recall is far higher because records with keywords such as "Nigeria," "Rwanda," or "Zimbabwe" are also selected, even though the term "Africa" itself cannot be found in these records. However powerful a thesaurus may be, if the keywords are not applied in a consistent way, the thesaurus loses a lot of its functionality. During manual input, automatic thesaurus checking may minimize this, but when importing records from different external databases, the chances that keywords are used consistently may 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. This may be especially helpful when the database contains scanned and OCRed records.

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 techniques 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.

Such 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 life 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 field indicators and/or keywords systems.

Searching a PBS need not necessarily be limited to the actual database records physically stored on the computer 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. This client-server-based searching over the Internet may result in amazingly fast and diversified retrieval performances and may offer the highest degree of streamlined import of external records by making it possible to just drag and drop records from one database to the other. On the downside, using a PBS retrieval engine in this way may not match the degree of sophistication offered by searching the remote databases directly.

DISPLAY

The immediate attraction of PBS very much depends on the way the databases can be browsed and how 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. Having the choice between different presentation formats is an 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, e.g., 25 records per screen, each limited 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. 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 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).

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, available operators, and overview of search sets). 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.

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: can the images be enlarged or reduced? Are thumbnails available? etc.

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 or mark records? Limitations 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.

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. 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. 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 numerals, change all lowercase letters to uppercase, or remove all punctuation. Some formatting issues, such as absolute positioning use exact 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 formula "et al."), 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 look of the formatted bibliography, such as ranging the references: sorting on several levels (freely choosing the fields involved), combining different sort directions, and page formatting (defining margins, indenting and numbering references, 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 system-wide. 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, system-wide 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. WYSIWYG previews of printed output on the screen offer a last control and can avoid wasteful printing of incorrectly formatted reference lists.

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 or XML format to put them on your private or corporate Web site or send search results by e-mail.

Word Processing

As discussed earlier, an important output-related feature 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 (typical 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 conforms 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.

This kind of integration can be quite sophisticated. In the best case, the PBS software, upon installation, adds extra database-related menu options to the toolbars of the word processor. Even when there is no integration whatsoever, it may prove useful to generate the reference list separately 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

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 units). Obviously, not all microcomputer-based PBS are network products. But in the best case, the system is fully network-compatible, 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. The option of mounting PBS databases on a Web server and offering access to them worldwide is clearly gaining in popularity.

CONCLUSION

Personal Bibliographic Systems are specialized software products, 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 entry 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; certain features will generally impress you, 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 amazing bells and whistles may hide fundamental shortcomings. Moreover, the usefulness of the various 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 and cons (or even the very existence) of the various features that may be encountered in this type of software. Potential users should define their own list of preferences and priorities and then select the real-life PBS that offers the best compromise.

BIBLIOGRAPHY

1. Brahmi, F.A.; Gall, C. Retrieval comparison of EndNote to search MEDLINE (Ovid and PubMed) versus searching them directly. Med. Ref. Serv. Q. **2004**, *23* (3), 25–32.
2. Brahmi, F.A.; Gall, C. EndNote and Reference Manager citation formats compared to instructions to authors in top medical journals. Med. Ref. Serv. Q. **2006**, *25* (2), 49–57.
3. East, J.W. Academic libraries and the provision of support for users of personal bibliographic software: a survey of Australian experience with EndNote. Libr. Autom. Syst. Inform. Exc. **2001**, *32* (1), 64–70.
4. East, J.W. Z39.50 and personal bibliographic software. Libr. Hi Tech J. **2003**, *21* (1), 34–43.

5. Hanson, T., Ed. *Bibliographic Software and the Electronic Library*; University of Hertfordshire Press: Hertfordshire, U.K., 1995; 136.
6. Harrison, M.; Summerton, S.; Peters, K. EndNote training for academic staff and students: the experience of the Manchester Metropolitan University library. New Rev. Acad. Libr. **2005**, *11* (1), 31–40.
7. Kelly, J.A. Downloading information using bibliographic management software: End-user software. In *Encyclopedia of Library and Information Science*; Marcel Dekker: New York, 1997; Vol. 59, Suppl. 22, 111–119.
8. Kessler, J.; Van Ullen, M.K. Citation generators: generating bibliographies for the next generation. J. Acad. Libr. **2005**, *31* (4), 310–316.
9. Koopman, A. Bibliographic citation management software for Web applications. Internet Ref. Serv. Q. **2002**, *7* (1/2), 99–112.
10. Mattison, D. Bibliographic research tools round-up. Searcher **2005**, *13* (Oct), 16–27.
11. McGrath, A. RefWorks investigated—An appropriate bibliographic management solution for health students at King's College London? Libr. Inform. Res. **2006**, *30* (94), 66–73.
12. Nicoll, L.H.; Quellette, T.H.; Bird, D.C.; Harper, J.; Kelley, J. Bibliography database managers; a comparative review. Comput. Nurs. **1996**, *14* (1), 45–56.
13. Schoonbaert, D. Personal bibliographic systems (PBS) for the PC: A generic survey of features. Electron. Libr. **1997**, *15* (1), 31–46.
14. Schoonbaert, D. Personal bibliographic systems. In *Encyclopedia of Library and Information Science*; Marcel Dek-ker: New York, 2002; Vol. 70, Suppl. 33, 326–362.
15. Sieverts, E. End-user software. In *Encyclopedia of Library and Information Science*; Marcel Dekker: New York, 1996; Vol. 57, Suppl. 20, 154–175.
16. Stigleman, S. Bibliography programs do Windows. Database April/May **1996**, *19*, 57–66.