CDS/ISIS and USMARC

Charles A. McFadden
Director
Virginia Institute of Marine Sciences
Library
Gloucester Point, VA 23062 U.S.A.

ABSTRACT

CDS/ISIS is UNESCO's free library automation software. It can be used on a PC or a larger computer (VAX). A PC network version is being tested and a UNIX version is imminent. One of the central failures of library automation systems is inflexibility. By importing and exporting records in the International Standards Organization format, it solves the communications problem. It is estimated that there are 15,000 CDS/ISIS users worldwide. Yet there are very few users in the US. Why don't more US libraries share in this universal brotherhood with its obvious benefits? One of the reasons is CDS/ISIS does not directly import USMARC records.

The paper will detail how to import USMARC records into CDS/ISIS, how the experience can be the gateway to importing any kind of record into the system (ASFA, DIALOG, WordPerfect, dBase) and why this is all part of the CDS/ISIS philosophy.

INTRODUCTION

CDS/ISIS, UNESCO's free bibliographic software, has a worldwide user base of more than 15,000, making it perhaps the most popular library automation software. However, in the US there are few users, very few in libraries not related to the United Nations.

The software has obvious advantages in addition to its broad user base. Since it is designed as the world's program, it implements the International Standards Organization (ISO 2709) format for the transmission of bibliographic information. Data exchange among users is guaranteed; the program forms a communications bond among them. Even if UNESCO folded, or if Giampaolo del Bigio fulfilled a lifelong dream and entered the Certosa di Calci, the program would live on and thrive. In an era of library automation when software can cost three times the amount of the hardware and can be in the hundreds of thousands of dollars, CDS/ISIS has a mainframe version which is free. In an era when entire systems are dumped as a prettier face appears, CDS/ISIS guarantees its longevity by refusing to commit to one

look: it is a system always in fieri, never in facto esse. Almost any function observed in a commercial system can be imitated using CDS/ISIS tools. Given a rainy afternoon and a sufficiently deviant personality, one could even design a GEAC or INNOPAC interface for one's system. A final strength is portability. There are PC standalone and network versions, and a VAX version. A UNIX version is rumored.

Why then do US hearts not flutter? First of all, US librarians have never heard of the software. But even if they had, there would be these objections:

-We already have a system;
-CDS/ISIS is small-time, a PC system for tiny libraries with a few thousand titles;
-It requires a knowledge of Pascal Programming Language for full implementation;
-It cannot import USMARC records.

The latter two objections will be answered with appropriate tedium in the body of the paper. However, one event would render all objections moot and all responses unnecessary. CDS/ISIS and the Sun UNIX Workstation are like two great magnitudes about to become a vector. A successful union would rival the conjunction of John Steinbeck and Ed Ricketts. To the union the Sun would bring muscle, UNIX network outreach, CDS/ISIS flexible bibliographic control, and ISO the internationally accepted data format. At the low, low price of $12,000.

The OPAC of the future will not be merely a finding list for materials owned by the library, but a subject bibliography at the article level, with items owned appropriately marked. We specialized libraries are particularly fortunate, since our OPAC bibliography can be so focused and exhaustive. Before too long a typical marine science library should expect to be able to afford a system capable of managing a database of 1,000,000 items. If CDS/ISIS were the common denominator, records honestly acquired could be honestly traded and shared among all interested libraries. An appropriate corollary to free software would be free records.

DECODING THE USMARC RECORD

One objection to CDS/ISIS is that it cannot directly import USMARC records. This is true, but changes can be made to the records so that they can be imported. To do this an understanding of the USMARC format is necessary.

The USMARC format is a standardized method of storing a record for transferring from one system to another. The local system, to which the record has been transferred, decodes the record into meaningful fields, which it then stores in an internal manner suitable for manipulation. USMARC format is for importing and exporting, not for internal manipulation. An OCLC USMARC record is shown in Fig. 1 as it would appear on a terminal. It is a string of letters, numbers and a few symbols, such as ! or ! or !. A file is a sequence of records, forming a single very long string of characters, each of which must be read sequentially.

Fig. 2 separates out the three sections of every USMARC record: the leader, the record directory, and the data. The first 24 characters always form the leader, which contains general information about the record, such as its total length. Next comes the record directory, which varies in length, but is always the number of fields times 12. In this case the record directory starts with 0010013, is 204 characters long (17 fields x 12 digits per field), and is terminated with the field terminator !. Finally comes the data, which begins with ocm2211 and ends with the record terminator !.

Fig. 3 uses the general information given by the leader, the specific field information given by the record directory, and the data to decode the record into meaningful fields. The first 5 digits of the leader tell the total length of the record, 703 characters, the last being the !. The 5 digits, starting at position 13 in the leader, indicate where the data part of the record begins. In this record, the data starts at position 229 from the beginning of the record. Counting starts with 0 in this scheme, so that the actual position where the data starts is 230; the o of ocm2211 is the 230th character in the string. Some of the familiar fixed field information of the OCLC record in also in the leader.

The record directory consists of a set of 12 digits for each field in the record. These sets are displayed in a column on the left in Fig. 3 under the leader. Each 12 digit set is composed of 3 numbers: 3 digits representing the familiar field tag; 4 digits representing the length of the field; 5 digits representing the starting position of the data within the data portion of the record (here also the count starts with 0, not 1). To the right of the 12 digit set representing 3 numbers is the actual field data. Note that the final character of each field is the field terminator !. ! is the subfield delimiter. The 2 digits or spaces in front of the initial ! are the field indicators. The author field has the tag 100, starts in position 185 in the data portion and takes up 24 characters.

Why is the USMARC record structure so complex? It is what is called a self-describing record. In a program like dBASE, which manages records with fixed length fields, when we create a new database we set up a record structure, describing and prescribing how many fields there are to be and how many characters in each field. But the situation becomes much more difficult if the record needs to be of variable length with a variable number of variable length fields. If the data for a field can vary drastically in length from record to record, the field length would have to be set in the record structure to the greatest possible length, which would be wasteful of space in the many cases, where the length is less. In addition, each record would be forced to have space allotted for every possible field, even though just a few fields would be used in a particular record. Fixed field programs force compromises and waste a lot of storage space. Variable data resists a predetermined structure; it requires a structure which can vary with the vagaries of each record. For this reason each record contains within itself instructions for decoding its fields, with little overhead in wasted space.

More importantly, it is not just a good idea; it is the law. Or rather the standard. USMARC is an implementation of the US national standard (NISO) for

bibliographic data transmission, which is in turn an implementation of the international standard (ISO 2709). The ISO standard prescribes everything but the tags: the tags must be 3 digit numbers, but the definition of the numeric tags is left to the specific implementation. The function of USMARC is to define the tags: 100 for author, 245 for title, etc.

IMPORTING USMARC RECORDS INTO CDS/ISIS

One of its great strengths is that CDS/ISIS incorporates the ISO 2709 standard: it imports and exports ISO records. Although USMARC uses the ISO format, the records must be slightly altered in order to import them into CDS/ISIS. Three changes must be made, and a fourth is optional. First, a carriage return and line feed must be placed after every 80 characters and after the record terminator. Second, the subfield delimiter must be changed from ! to ^. Third, certain non-numeric characters in the leader must be changed to numbers. The optional change is a way to retain the characters removed from the leader.

As was explained above, a USMARC record is a continuous string of characters. To make changes one must use a programming language to read from the source medium a sequence of characters from this string, make the changes and copy the altered sequence back to a target medium. The Pascal programming part of CDS/ISIS cannot read a file that does not have carriage returns and line feeds in it already, and cannot therefore be used to insert them. Turbo Pascal, an inexpensive programming language, can be used to do this. 80 characters are read from the tape, a carriage return and line feed are added, and the 82 characters are written to a file on the hard disk. When a record terminator is encountered, the same change is made. The first line of each record must be limited to 68 characters, not 80, to leave room for what gets added if you are saving the leader information. This is repeated until the end of the file is reached.

At this point the file can be dealt with by the Pascal in CDS/ISIS. A similar tactic is used to change the USMARC subfield delimiter ! to the CDS/ISIS delimiter ^. Characters are read in and tested one by one. If the character is not ! it is copied to the target file. If it is ! then ^ is written to the file in its place. This process is repeated until the end of the file is reached.

The third change needs to be made to the leader. Listed below one another are the leader from the USMARC record and the leader from the CDS/ISIS record.

00703pam 2200229 a 45e0
00703000000000229000 4500

It seems that not all of the non-numeric characters need to be changed to zeros. The offending character seems to be the 23rd one, in this case "e". Again the file is read, the first 22 characters of each leader are read and copied, 0 is copied instead of "e" or whatever is there, and reading and copying picks up with the 24th

character. In this case the leader of each record needs to be identified so that the 23rd character can be changed. The leader of the first record is the very first part of the whole string; the leader of the subsequent records is always the line immediately after the record terminator ! of the preceding record.

The final change, which might be regarded as optional, concerns 6 of the non-numeric characters of the leader. In the USMARC leader characters 6-8 and 18-20 carry substantive fixed-field information about the record. This includes record status, type of record, bibliographic level, encoding level, descriptive cataloging form and linked record requirement. However, this information is lost when the record is imported into CDS/ISIS. One method of retaining this information is to add a field containing these 6 characters to the USMARC record before importing the record. This involves 4 logical steps. First, add the actual data at the end of the record, after the last ! and before the record terminator !. The second step is to add a 12 digit entry to the record directory, identifying the added field. The third step is to adjust the starting point of the data from 229 to 241 (229+12). Finally, the record length must be adjusted by 23 to reflect the addition not only of the data, but also of the record directory entry.

The actual programming steps are as follows. The leader for a record is located as described above. The first 5 digits represent the record length. This number is increased by 23, the total of new characters added, 12 to the record directory, 11 in the data field. It is copied into the first 5 characters of the target file. Characters 6-12 are copied as is. Characters 13-17 represent the starting point of the data. Since there is a new 12 digit entry in the record directory, this figure 229 is increased by 12 to 241 and written to the file. Characters 18-24 are copied as is. Before the present record directory is copied, the new 12 digit entry (999001100473) is written to the file. 999 is the field tag. This new field is always 0011 characters long: 2 indicators, delimiter, subfield letter, 6 data characters and field terminator; eg 0 ^apam a !. The starting position of the field (00473) has to be calculated: 726-(241+1)-11 or the new record length minus the starting point plus one (since counting starts at 0) minus the length of the field. Next, the old record directory is copied. Then the data portion up to and including the last ! is written. At this point the new data field 0 ^apam a ! is written to the file, followed by the record terminator !.

This solution is inelegant, but fairly easy. It involves a minimum number of changes to the record. The field tag used is 999, so that it can be the last field in the record and need not disturb the calculations in the record directory. The data is placed at the correct place in the data portion of the record; i.e., last. However, the record directory entry is placed first in the record directory, rather than last where it belongs, because this is easier. Once the leader information is included inside the record, it is then available for use. At this point the records are ready to be imported into CDS/ISIS.

As with any data base manager, when we create a database, we must set up

a field definition table. In CDS/ISIS the maximum number of fields is 200. Use the OCLC Books tables or the USMARC tables to select the 200 most useful fields. The USMARC tables are probably more useful. Enter the tag number and repeatability of each field into the field definition table. Make all fields alphanumeric in type. When the FDT asks for a field length, all it wants is the greatest possible length of the field. If the field is not present or is shorter in a particular record, it only uses as much space as is needed. Thus, setting up the FDT in CDS/ISIS is different than in dBASE. In dBASE disk space equal to the sum of the lengths of the fields is used no matter how much data is actually in the record. In CDS/ISIS, even though lengths are assigned to the fields, only as much disk space is used as is necessary for actual data. There seems to be no need to define subfields.

With a little creative selection of fields, one can fashion a sort of uniMARC record. Any field for any format, even 8xx holdings fields, can be included. In addition to the FDT, a format must be set up which can properly display the USMARC record on the screen. All fields in the FDT should be included in a format to display the full record, which uses as field names the actual numeric tags and displays subfield delimiters as in the OCLC record. Setting up the FDT and the format seems like a lot of tedious work, and it is, but, like a 1950's Charlton Heston movie, it feels longer than it actually is.

PASCAL: PROBLEM OR SOLUTION?

Almost no one programs any more. Systems librarians in very many libraries have never written a program for work. People who do program often have to treat it like a secret vice, hiding it from their superiors. Behind this negative pressure there seem to be two assumptions: programming is an impossibly difficult and erudite mathematical skill; a program has already been written for every conceivable purpose, which can (and should) be bought.

If by programming one means sitting down and constructing from scratch library automation software, I agree. If by programming one means Fortran or machine language, I agree. But CDS/ISIS Pascal offers a kinder, gentler approach. CDS/ISIS itself is a relatively complete library automation system, so let there be no talk of starting from scratch. Unlike commercial systems, in addition to its menued use, it exists also as a set of powerful, specialized bibliographic commands, extending the set of generic Pascal commands.

Pascal is itself a felicitous choice. Most applications today are written in C. But Pascal is to C what psychotherapy is to psychoanalysis: for 60% of the effort you can get 95% of the results. Pascal is often taught in school since it is logically pleasing and not that hard to learn, certainly not as hard as learning a foreign language. Nor must one learn all of programming, just string manipulation. There is no need for number crunching, etc. Furthermore, all of the string manipulation centers around the ISO 2709 format.

As with all secret vice, programming is pleasurable and satisfying. You will notice a certain bounce in the step, a certain cock of the hat, of someone who does not have to program, but does nonetheless. This is especially true of Giampaolo del Bigio, High Priest of ISIS. He can often be seen sitting in an outdoor cafe on the Right Bank, opposite the !le de Saint Louis, working on his algorithme. Girls, long in the leg and short in the skirt, call out, "Ciao, Giampaolo!", as they whisk by on their motor scooters. Across the table, sipping a Pernod and water, and reading Le Monde, is his companion, who looks like Capucine. Giampaolo knits his brow and then smiles, as he solves yet another problem in his adaptation of CDS/ISIS to UNIX.

The library profession has always held an attraction for those of an analytical bent. CDS/ISIS does not require that everyone be a programmer, but offers the opportunity to those who are so inclined. A CDS/ISIS site need have only one person on its staff, who takes on the task of programming.

CONCLUSION

CDS/ISIS, by its adherence to international standards, its flexibility and its broad user base, can act as a worldwide bibliographic pipeline. If its users, who have received the software free, in turn offer their records freely to one another, bibliography, which has slipped out of the hands of librarians into greedy commercial hands, will again become free.

Curiously out of touch with CDS/ISIS are US librarians. Two possible obstacles to US participation are the software's seeming hostility to USMARC records and the need to program for full implementation. However, USMARC records do adhere to the same standard as CDS/ISIS records, and, with slight changes, can be imported. Nor is programming a problem, because it is easy, applications programming, and should be viewed in the same light as learning to drive a performance car with stick shift, or learning to use a fine old camera with manual features. The skill is the human consummation of the process.

Crucial to the penetration by CDS/ISIS of the world of serious library automation is its migration to UNIX. DOS PCs have become UNIX wanna be's. Networking and multitasking are alien to DOS, and must be grafted on unnaturally. Serious computing starts at the next level of power and operating systems, which at present is represented by UNIX and perhaps the Sun Workstation. $10K - $12K is a reasonable amount of money for a small library to pay for a mulituser, multitasking computer, which can manage a database of 1,000,000 records. If CDS/ISIS helps make this happen at no added cost for software, it will have no competitor!