Design Issues in Automatic Translation for Online Information Retrieval Systems

Introduction

One objective of computer intermediary systems is to minimize incidental and accidental differences among the many distinct languages found in online bibliographic retrieval. Three classes of languages are identified: access protocols, retrieval commands/responses, and database structures. Each class has its own characteristic requirements for automatic translation. In developing one intermediary product—the Sci-Mate Searcher—distinct translation approaches proved most effective for each class: a procedural language for access protocols, customized coding for retrieval commands/responses, and a knowledge-based table for database structures. Despite differences in translation methods, users are presented with a consistent view throughout the product.

The Problem: Online Babel

Online bibliographic information retrieval, from a systems point of view, is not user friendly. Using many heterogeneous online bibliographic services can be difficult for professional searchers and nearly impossible for occasional end users for several reasons. These include the problems of database selection, strategy development, and the overwhelming and sometimes contradictory details of usage and syntax. This paper addresses this last source of difficulty, one that is more likely to be solved in the near future by automation than the semantic and subject-knowledge issues.

Online services provide access to enormous amounts of information but at the same time pose linguistic barriers to their own broad usage. The number of services with distinct protocols and languages continues to grow. There are now five major packet switching networks in the United
States and one in every European country. At least fifteen bibliographic database hosts in the United States, Canada, and Europe offer hundreds of databases with many of the specialized databases found only on a single host. Besides containing unique information, each database is structured with distinct field designations and data coordination conventions.

The linguistic conventions used in online searching are by design terse and cryptic. With cost a function of time spent online, brevity is mandatory. For searching to be cost effective at even the relatively fast transfer rate of 2400 baud, a minimal user environment is preferable. However, this does not excuse the great diversity and incompatibility of commands, codes, and conventions.

From one system to the next, a given function usually is invoked by a keyword entirely unique to the system. With a few exceptions, there is no opportunity to define synonyms or otherwise improve consistency among the systems used. The babel of distinct protocols and language conventions now being used by online systems derives from the history of their development.

The packet switching networks—Telenet, Tymnet, Uninet, and Infonet—were developed independently from one another as competitive services. Each interacts with users using their own distinctive protocols and conventions.

In the early and mid-1970s, development efforts in bibliographic search software were independently conducted by several firms, notably Dialog (Lockheed), Orbit (SDC), and Bibliographic Retrieval Service (BRS). Dialog evolved out of Recon, funded by the National Aeronautics and Space Administration, while Orbit evolved out of Elhill under contract with the National Library of Medicine. BRS Search, originally derived from IBM's STAIRS software, has always been a commercial search service. During this independent development, there was no coordination of language terminology and syntax.

Many of the early commercial bibliographic databases derived from federal and private publishers of printed tertiary indexes. It is remarkable that data from so many diverse sources were brought together and made to work under one or more vendors' retrieval software. The data in many cases were not initially intended for distribution as an online database. It is quite understandable that most databases follow their own distinctive indexing and fielding conventions.

At the present time, economic and technical constraints work against significant change in the online systems’ software. The networks, retrieval hosts, and database publishers all have invested thousands of man-years in software and data. Customers who have learned to use these systems depend on them remaining stable. All involved are understandably reluc-
tant to look for and convert to any proposed standard, such as the European Common Command Language.¹

Given this situation, the information retrieval specialists often must make choices that are not fully satisfactory. The specialists may limit themselves to one or two host systems and a few select databases. The capabilities of the systems and the databases' content and structure can thus be thoroughly mastered to accomplish all that is possible with the selected facilities. But in so restricting themselves, vital information on hosts and databases not used will be missed.

On the other hand, specialists may choose to learn how to use a broad selection of host systems and databases in order to access all possible sources of relevant information. Much of what must be learned are details of access protocols, commands, and database syntax—the only means to getting the information itself. Learning and maintaining proficiency with many different systems is costly and may result in specialists with more diffuse and less expert skills than those who choose a more limited scope.

Technical end users are even more restricted by the linguistic barriers posed by conventional database systems. They can afford less time than specialists to devote to learning details of access, retrieval, and database structure. They are frequently bewildered by the diversity of options. Most of the time, end users will turn to specialists to meet their information needs even though they know the technical terminology better and are better equipped to judge the results of the search.²

A Natural Solution: Intermediate Translating Computers

For more than ten years, various automated solutions to the linguistic problem have been proposed and implemented. These usually consist of a computer placed between users and one or more host systems. Known generally as "computer intermediaries," these systems function in part as translators that mask incidental and accidental differences between languages in the access and retrieval process.

Computer intermediaries provide a set of services that usually go beyond translation. Frequently a richer and more consistent user environment adds customized value to the entire process. Uploading allows locally stored and maintained strategies to be sent to the host. Downloading allows results retrieved from host systems to be locally saved and processed. Assistance is given in selecting databases; online descriptions of the systems and databases are available; accounting subsystems are provided; and results can sometimes be transformed and factored back into queries.

Examples of mediating systems involving software running on stand-alone dial-up mainframes include the experimental CONIT, at MIT, and the former Chemical Substances Information Network (CSIN), funded by
the National Library of Medicine. Another switching and mediating service that is dialed up but uses microprocessor hardware is EasyNet.3

Examples of software that can run on the user's own microcomputer include several packages that are no longer actively marketed: OL'SAM from the Franklin Institute, InSearch and ProSearch from the Menlo Corporation, and SearchMaster from SDC.4 Microcomputer software currently available includes the Sci-Mate Searcher (Version 2.0) from the Institute for Scientific Information (ISI), Micro-CSIN and the Grateful Med from the National Library of Medicine, and Search Works from Online Research Systems.5

In developing ISI's Sci-Mate Searcher, the use and structure of network access, retrieval languages, and databases suggested distinct methods for automatic translation. The characteristics of use and structure and the methods developed to accommodate them will be described in the next four sections of this paper. Particular features of the Sci-Mate Searcher will not be described. Rather, general principles of automating online language translation will be described.

Two Interfaces: The User and the External System

When performing translation functions, an intermediate computer must manage two distinct language interfaces: one with the user and one with the external systems. Recognizing these two interfaces as distinct and isolating their distinctive operations in separate modules is essential for successful design.6

The user interface provides all significant retrieval functions and capabilities to the user. Controlled here is what the user can request and the way it can be requested. Also controlled here is what the user is given from the external system and the way it is presented.

"The external system" refers to everything beyond the serial port of the intermediate computer. The external system interface defines what modems, networks, and host systems are supported. It also defines the functions and capabilities that will be used in interacting with these external systems. The commands issued on the serial line are constructed in the external system interface modules, and the responses from the external systems are first processed by these modules too.

Intermediate translating software could be designed and written to directly convert user input into a form required by the current host and directly convert the current host responses into a standard form for the user. It is tempting to design and code this way. However, as more entities and capabilities are added to the external system, designing and writing yet another module for direct translation becomes quickly untenable; the mediation software soon becomes an intricate and incomprehensible network of code; and maintenance becomes impossible.
Design and implementation is much more easily managed if intermediate data structures are defined for all transactions. These structures store inputs from both users and systems in a standard form. The structures are accessed in separate steps for constructing acceptable commands for the network and host systems and for presenting consistently formatted information from the host to the user. The data structures serve as a buffer between the two interfaces. In the Sci-Mate Searcher, these data structures have been called the intermediate language.

With the intermediate language, user input destined for the host is accepted without regard to which host is currently online. Only in the system interface step that follows are the particulars for the current host added to data extracted from the intermediate structure. Similarly, responses from the current host are stored in the intermediate language data structures. These are then accessed in a separate step as host-independent data which are transformed and presented to the user.

Access Protocols: Description and Requirements

Access protocols here refers to the process of negotiating modems, networks, and host system passwords. Access is often viewed as a rote but necessary nuisance. It involves a series of steps which depend upon the details of particular hardware and systems. These steps include the following:

1. dial the network node, manually or through modem control;
2. inform the network about the terminal speed and type;
3. instruct the network about flow control and line padding;
4. specify the name or address of the host system;
5. negotiate the password(s) with the host; and
6. answer any standard host questions about news, etc.

Until the early 1980s, modems were usually dialed manually. Intelligent modems which allow software to control dialing now make it possible to fully automate this task. Among intelligent modems, models from D.C. Hayes have set the standard for control language. Data networks presently used in the United States for online access include Telenet, Tymnet, Uninet, Dialnet, and InfoNet.

The user's primary requirement for automated access is to get connected to a particular host or database. Users at all levels of expertise would like to log on by simply naming the database. All the steps of the process are amenable to complete automation. However, if a failure occurs and automatic access cannot be carried out, users should be able to choose alternatives such as trying again, quitting the session, or taking over manually.
A special-purpose procedural language was felt to be appropriate for automating access in the Sci-Mate Searcher. Access is algorithmic in that it has a clear beginning and end. Translation by a procedural language is effective in access where only a small number of possible messages from the external system can be anticipated at any given moment in the process. These messages, and the lack of any message in a given period of time, are known as potential states in the process. When a possible message is received, the state is said to be realized.

The actions to be performed when a potential state is realized are few and readily defined and programmed. These actions mainly involve a message in response. After specifying the host or database, the user becomes largely an observer during automated access. This allows the intermediate computer to control and respond to other computers, a relatively straightforward task in the case of access.

**Retrieval Commands and Responses: Description and Requirements**

The retrieval languages of the bibliographic host systems consist of commands entered by users and host responses to these commands. In the United States and Canada there are about half a dozen major host systems; in Europe there are at least six more. In each case, they accept a command consisting of a command verb (sometimes implied) followed by an argument.

"Retrieval command and response" will refer here to language components that control the host retrieval software but are independent of any specific database. Most retrieval systems provide at least the following basic commands:

1. pick a database or set of databases;
2. browse inverted indexes to the database(s);
3. select terms and specify term logic;
4. display records from sets constructed;
5. request records to be printed and mailed;
6. review the sets created during the session; and
7. leave the system.

Additional capabilities are frequently provided that build on these basic ones. They include commands to:

1. make selections from the inverted index display;
2. search for complete phrases in database records;
3. limit the search to years, updates, or languages;
4. specify ranges of records and formats for displaying and printing records; and
5. save strategies—recall and use saved strategies.

The retrieval software systems always allow a series of sets to be created. These sets consist of pointers to records which can be directly reviewed. The sets of pointers can also be used in the argument of the selection command. As terms in logical expressions they result in further sets as the search strategy is refined.

Automated translation of retrieval system commands must provide a consistent syntax in place of the broad diversity of construction required by the different systems accessed. This requires at the very least a single set of command verbs or function names to be used across systems. It further requires a unified, or at least consistent, set of conventions in the construction of command arguments. Finally, responses from the host should be standardized before being presented to the user.

Commands for the host systems are constructed from: (1) a standard command specified by the user, (2) data elements entered by the user, and (3) punctuation and other connecting elements required by the host. All of these are ordered as required by the host. Responses received from the host are parsed into the intermediate language tables from which significant data elements are extracted and reconstructed in a consistent form for presentation to the user.

The intermediate system should automatically enter and leave the "modes" found on hierarchically organized command systems, such as BRS and Questel. All information about sets created and commands issued in the current session are saved for the duration of the session. As part of response parsing, the intermediate system recognizes error messages and conditions and can assert failure when an excessive time delay occurs.

In developing Sci-Mate, directly coded routines have been found to be most effective for translating intermediate language data into multiple host system retrieval commands. Here, exceptions prevail over rules. Conversely, directly coded translations from the intermediate language into a unified user presentation have been found to provide more effective direct control than a meta-representation could possibly provide.

Retrieval languages come in families: Orbit and Elhill share a common origin and form one family; Recon, Dialog, and ESA-IRS/Quest form another; BRS and DataStar (Switzerland) form a third; and there are others. In translating retrieval commands and responses, a matrix of functions by retrieval language family must be managed.

The only regular form or pattern in the commands across the language families is the verb-argument arrangement. Even here the verb is often implied especially in the selection function. The syntax of each argument in each language family follows no pattern that can be observed across language families. Figure 1 gives an example of just those cells of the
matrix that contain the Dialog, BRS, and NLM commands for display. Figure 2 shows the various ways in which Dialog, BRS, and NLM report sets formed.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>VERB</th>
<th>ARGUMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dialog</td>
<td>T or TYPE</td>
<td>&lt;set #&gt; / &lt;format code&gt; / &lt;item list or range&gt;</td>
</tr>
<tr>
<td>BRS</td>
<td>..P or ..PRINT</td>
<td>&lt;set #&gt; &lt;format code&gt; / DOC= &lt;item list or range&gt;</td>
</tr>
<tr>
<td>NLM</td>
<td>PRT or PRINT</td>
<td>SS &lt;set#&gt; &lt;format&gt; SKIP &lt;first record # - 1&gt;</td>
</tr>
</tbody>
</table>

Figure 1. Variations among host requirements for the Online Display Command.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dialog</td>
<td>&lt;set #&gt; &lt;postings count&gt; &lt;set formation expression&gt;</td>
</tr>
<tr>
<td>BRS</td>
<td>RESULT &lt;postings count&gt; DOCUMENTS</td>
</tr>
<tr>
<td>NLM</td>
<td>SS &lt;set #&gt; PSTG &lt;postings count&gt;</td>
</tr>
</tbody>
</table>

Figure 2. Variations among host responses to Set Formation Commands.

Modularization remains an important principle and practice. In particular, the user interface in Sci-Mate provides a separate module for each function. The host interface contains routines for all host functions in a single module with each host function managed by one or more procedures. Both the host and user interfaces draw data from and provide data to the Intermediate Language data structures where standardization ultimately takes place.

Database Structure: Description and Requirements

Online bibliographic databases show many parallel structural characteristics even across hosts. The relative simplicity and consistency of their structures make it possible to define and store fairly complete information about the structure, but of course not about the content, of most databases.

The data records themselves are textual with variable length fields. All hosts have at least one format in which the fields are labeled with prefix
tags of two to four characters. The labels are followed by one or more lines of textual data. The second and subsequent lines of data in a field are usually indented to show that they are part of the same field.

Inverted indexes provide retrieval keys for the data in most fields. A basic index containing terms from all fields is present on many hosts. Other fields have their own inverted indexes. Usually the contents of inverted indexes can be reviewed starting anywhere and continuing through the index in alphanumeric order.

There are three ways in which term coordination is handled in searching. Some fields can be searched using only single word terms. Other fields precoordinate two or more terms into searchable phrases in addition to single word terms. Finally, proximity or adjacency searching, in which terms are postcoordinated at the time the command is constructed, is allowed in most fields.

The user's primary requirement at the database level is information about the contents and structure of the database. Such information can guide the user in the selection of appropriate fields and the construction of search expressions. This information also specifies what tags to use to designate fields and the acceptable form for terms and expressions in each field.

This information can be found in the database provider's documentation and in the host system's fact sheets. After locating the information, the conventional searcher must switch attention between the manual and the terminal screen. Users can also experiment while online to determine the syntax allowed in a field, but this can be time-consuming and therefore costly. Conventional online searching can be enhanced by making these tools immediately available on the terminal screen.

A table of database information is used in the Sci-Mate Searcher to translate database syntax. Here the task consists of transforming one definition of the data structure into another. Both user and host requirements for information about the database are represented in a single entry in the table. The table may be called a "knowledge base" since it represents expert knowledge about databases.

In this knowledge base, the tags and usage of database elements required by the host are mapped to more complete names and descriptions for the same elements stored for users. Users face a mnemonic and encyclopedic problem in handling details of syntax at the database level. The knowledge base is intended to solve this problem with immediate information about the database.

In addition to information about the fields, certain global information about the database must also be stored. This includes: the name of the database as it is to be presented to the user; the name of the database as it is
known to the host; and an indication of the host on which the database is found.

The knowledge base stores information for both the user and the host system. For the user, the complete names of the fields are presented as part of a menu selection. After a selection is made, stored descriptions of the fields and its subfields are given as prompts. The user is told whether or not phrases are allowed in the field.

For the host system, the value, placement, and punctuation of the field tag is extracted from the table and used in the construction of selection or browse commands. If phrases are allowed in the field, the appropriate adjacency or proximity symbols are supplied by the table.

**Consistent User Presentation for the Retrieval Process**

Three major language classes used in information retrieval have been identified: access protocols, retrieval commands and responses, and database structures. Numerous specific languages are found within each of these classes. For computer mediation, distinct translation methods were found to be most effective with each language class. How can these distinct translation methods be integrated for a consistent presentation to the user?

First, what is meant by a consistent user presentation? This at least means one in which the user is asked to become accustomed to a manageable, well-defined, and easily learned set of conventions. It also means one in which the transitions from one set of options to another fit logically and naturally into the user’s experience and expectations. It does not mean that some particular device or method made possible by the hardware and software is necessarily employed. ⁸

Over the past few years there has been a trend away from explicit and toward implicit language interfaces for users. ⁹ The oldest form of interactive interface and the one that traditional online information systems continue to take, is the host-prompt/user-command/host-response. This derives from the technology and economics of early timesharing. This form is one dimensional in that it looks like a simple dialogue alternating between a line from the user and one or more lines from the host system. Being mnemonic, it requires users to remember or quickly locate details about how the language can be used.

As smart terminals and microcomputers become available, much higher display transfer rates are possible. For user interaction, options can be economically presented on a menu. The two-dimensional surface of the video display unit (VDU), fully refreshed in less than a second, presents the user with explicit options for direct selection. Thus interaction requires less to be remembered and less to be entered, as a selection of an entry from a
menu is sufficient. The machine becomes proactive as it supplies direct and verbal options to its user. Much software for microcomputers, including Sci-Mate, uses this method for interacting with users.

The newest class of interactive interfaces communicates in still more implicit language. New devices and methods give a spatial and environmental feeling to user interaction. Color graphics and icons communicate concepts without using words. Pop-up windows and pull-down menus present options instantly without obliterating the current context. Pointing devices such as mice allow metaphorical navigation to every location on the screen. Foreground/background activities make users feel that they have to wait less time for the machine to complete its tasks. Memory-resident utilities allow machines to be used for diverse tasks; applications are available at the touch of a key.

No retrieval intermediary software developed to date has fully taken advantage of the new devices and techniques for interfacing with users. Almost all software in the mediation genre uses either a command- or menu-driven interface. However, consistency and ease of use are not precluded by either of these methods.

For access protocols, it is sufficient to provide users a method of selecting either the database or the host system. A further step is to automatically select a database or set of databases depending on the general subject area being researched. This was done by In-Search and is being done by EasyNet. Such a selection can be performed adequately by either a command-driven or menu-driven interface.

For retrieval commands and responses, users should be able to simply specify the function or operation to be performed. Better yet would be to provide users with recommendations for actions such as was done by the Individualized Instruction for Data Access (IIDA) project.\(^\text{10}\) In either case, a menu of possible or recommended functions continually available to the user for selection is helpful.

Finally, for database structures represented by a knowledge base, a menu can effectively provide the field names for selection. The tables are further used to present prompts for both field and subfield data.

**Conclusions**

Three distinct classes of language have been identified in online bibliographic information retrieval. These classes are: access protocol, retrieval command/response, and database syntax. Computer intermediary systems that perform language translation should recognize the special problems and requirements posed by each class and adapt mediation to fit these problems.
In developing the Sci-Mate Searcher, it was found that a special-purpose procedural language was most effective for managing access protocol; direct computer routines drawing upon intermediate data structures were most effective in handling retrieval commands/responses, and a knowledge base was most effective in dealing with database syntax. Despite the different languages and methods for their translation, the user can and should be presented with a consistent view of the whole mediated process.

REFERENCES