BALL, A.J.S.

User interfaces for future videotex systems.

2. # User Interfaces for Future Videotex Systems

A.J.S. Ball
Videotex Consulting Services
2268 Osler St
Regina, Sask. J4P 1W8

and

J. Gecsei
Département d'informatique et
de recherche opérationnelle
Université de Montréal
C.P. 6128, Succ. A,
Montréal, Québec, H3C 3J7

May 1981

This report is prepared under contract 2 OSU.36100-0-9506 for the Department of Communications, Canada.

TABLE OF CONTENTS

Pages

1. Introduction: what is videotex 2

2. Range of user interfaces 6

3.. Terminology : a clarification 10

4. Menu-based processing 12

4.1 Principle 12

4.2 Greek Restaurant example 14

4.3 Retrieval utility 14

4.4 Database construction and maintenance 18

4.5 Appropriate applications of menu systems 19

4.6 Summary 21

5. Pre-coordinate keyword processing 22

5.1 Example of a system 22

5.2 Keyword system at the Universite de Montreal 23

5.3 The Greek restaurant example 24

5.4 Discussion : suitability for videotex 25

6. Post-coordinate keyword processing 26

6.1 Explanation based on commercial systems 26

6.2 The Greek Restaurant Example 27

6.3 Discussion : flaws in current systems 28

6.4 Suitability for videotex 30

6.5 Efficiency problems 32

7. Interface languages 34

7.1 Introduction and classifications 34

7.2 Artificial interface languages 37

7.3 Natural language interfaces 41

7.4 Interface languages in videotex 45

8. Conclusions and recommendations 50

REFERENCES 53

FIGURES 60

1. Introduction : what is videotex

This report is part of a study whose aim is to clarify the connections between existing database management and data retrieval systems, and between the newly emerging technology of videotex. More precisely, we wish to examine the benefits that can be drawn from the rich experience and knowledge accumulated with traditional databases, for improving videotex database technology. In this report we focus on the interface mechanisms and languages that are employed by the users of such systems to specify and execute the desired queries and other related operations.

It might seem that the task on hand is simply to examine a well organized body of knowledge and to see what parts of it are applicable to another well defined domain. Unfortunately this is not the case. First, there is no such well organized science of database management systems. In reality many different classes and subclasses of databases exist, each with its own particularities, user characteristics and outstanding achievements. The most important categories are file systems, record oriented databases, page oriented databases such as used in current videotex systems, bibliographical reference systems, full text databases, knowledge bases and expert systems. Each of these areas has its specific terminology, literature and schools of thought; however, practically no references can be found dealing with their commonalities and no general theory covering all cases exists.

The second point is that videotex is a recent development, growing in the midst of other similar technologies which suffers from a crisis of personality. To name a few areas which partially overlap with videotex : electronic mail and messaging, office automation systems, home computers and their networks, teletext (in this study we are concerned mostly with two-way videotex), corporate (private) database systems and computer-aided instruction. So what is videotex, and what will it become in the next ten years? These are not mere rhetoric questions, since the conclusions and recommendations of our report almost entirely depend on the assumed answers to these questions. If we take the view that videotex is a way to provide public access to all kinds of data processing, then obviously all existing database technologies will have to be qualified as "suitable". If we adopt a more conservative perspective and limit videotex to a cheap, simple, general purpose retrieval system using a TV set and keypad, then the study makes no sense since such systems already exist.

In order to have a working definition, we give a list of some features which, if not individually then at least in combination, convey the flavor of what we mean by a videotex system :

1. It is at least a partly public system to which any individual or organization can freely subscribe. Business and home applications share the same system.

2. The service is low cost.

3. Access is basically to display pages, not sets of records.

4. Close link between information structure and display structure.

5. Mix of color, graphics and text on screen.

6. Initially based on existing distribution networks.

7. Large numbers of simultaneous users active, leading to limitations of transaction complexity.

8. Many divers information providers; large portions of data are quickly changing; in this respect it resembles a two-way information channel rather than a database.

9. The typical user is computer naive; therefore simplicity of the user interface is essential.

For the sake of completeness we quote the following definition of videotex found in [27] : "Systems for the widespread dissemination of text and graphic information by wholly electronic means for display on low cost terminals (often suitably equipped television receivers) under selective control of the recipient, using control procedures easily understood by untrained users.

If it is hard to define today's videotex, it appears to be next to impossible to predict its future course. Without further commentary we limit ourselves to a few possible scenarios into which videotex may eventually evolve in the next 5-10 years.

1. Videotex may become an integrated public information utility providing all kinds of computing and information services to the general public. High resolution intelligent terminals will be used that can incidentally serve also as a TV set.

2. Videotex becomes an "umbrella service", a kind of switching device providing for the use of an augmented TV set as

an access point to a collection of independent specialized services.

3. Interactive videotex will vanish and will be replaced by independent services and large teletext databases (eg. running on multiple full TV channels, containing only frequently updated public information).

4. Interactive videotex will remain essentially a collection of simple small local interest databases, with much the same interface as today's systems.

In our study we try to be as independent of definitions and forecasts as possible : we simply look at a variety of facilities and systems and see how they would lend themselves to efficient implementations of applications potentially useful to large sectors of the public and business.

The following sections examine user interfaces to database and information retrieval systems in the approximate order of their complexity : menu-based systems, pre- and postcoordinate keyword processing, command language and natural language interfaces, and question answering and expert systems.

To illustrate different interface languages and to facilitate their comparisons, we use throughout the report the same example, the problem of how to find a cheap Greek restaurant.

2. Range of user interfaces

User interfaces for interactive information retrieval systems can be approximately classified according to the quantity of information required from the user at each elementary interaction. This is illustrated on the vertical axis of Figure 1.

a) The simplest technique is the Yes-No (True-False) interface. The interaction consists of questions displayed by the system, to which the user may only reply Y or N. The method is seldom used as the sole means of interaction, but frequently in conjunction with other techniques.

b) Menu selections, which are a straightforward extension of the Yes-No type interface. Most videotex systems today use menus as the principal means of data retrieval. Menus are typically composed and formatted into display pages and linked together at database creation time; however, dynamic menu pages created at retrieval time are possible, e.g. as the result of post-coordinate processing (section 6). Progressing from menu to menu is often referred to as "navigating" the database. The main advantages of menus are (1) a relatively simple and efficient implementation, and (2) the possible use of a numerical keypad only. However, the method has some serious drawbacks, as discussed in Section 4, in particular for larger databases.

c) Many information retrieval systems use keywords as the basic means of retrieval specification. In its simplest form a keyword based system appears to the user as a large associative store capable of retrieving documents or records containing (or associated with) a given keyword. The principal advantage of this approach is that it permits the user to access "directly" the desired information, without concern about the way the data are structured. The following problem areas can be identified with keyword systems : (1) maintaining and controlling the vocabulary of permitted keywords, (2) handling of multiple-hit and no-hit situations (i.e. when too many or no documents are found to correspond to a keyword), and (3) technical difficulties of simulating associativity (usually by large and numerous indexes). Menu and keyword systems are in a sense complementary in nature. Menus can be seen as small lists of keywords currently expected by the system, which makes misses virtually impossible; however, to retrieve a document typically takes a large number of selections. Keywords provide faster access, but only if the user knows the "correct" (applicable) set of keywords from the context of his search.

Two particular instances where keywords can be used advantageously are (1) systems with very large keyword directories, such as bibliographical retrieval systems, where the miss probability is low even though the "realm of search" is, at least initially, not well defined; (2) situations where the realm of search is known, but there are too many keywords to be listed in a menu, such as eg. teleshopping for groceries. Here, using only a few hundred keywords (grocery categories and articles), the user can have access directly to the desired item with a quite low miss probability.

Two important classes of keyword-based systems can be identified : those based on pre-coordinated and post-coordinated indexing. Some form of keyword facility seems to be desirable and feasible for the enhancement of videotex systems. The French Teletel-STAR system [5] has an interesting keyword facility, and Prestel has announced plans for an "automated subject index" [1]. Telidon has a provision for keywords in its internal record format.

d) Queries and command language interfaces. Typical post-coordinate systems permit the retrieval specifications to contain logical combinations of several keywords or queries. Such queries and other information supplied by the user (such as commands to create new information files or to execute programs) are usually given in some command language. Most command languages for information retrieval and general purpose computing systems are designed for the experienced user, have rigid and unforgiving syntax, and therefore are not suitable for the average videotex user.

e) Natural language and interrogative (user-friendly) interfaces. There seems to be a growing concern among designers of information systems for the casual, nonexpert user. In effect, a number of user-friendly, or graceful interfaces have already been designed which accept commands from the user in natural or near-natural language. However, natural language is not a prerequisite of friendly interaction [20]. Such interfaces are programmed to tolerate deviations of syntax and ambiguities of meaning. When ambiguities occur, the interface requests additional information from the user.

In many instances two levels of interface exist : the database interface (typically a mixed command and query language), and the user interface, which is implemented as a translator from natural to the command language. The latter can run on a separate computer, perhaps on some kind of videotex access machine [2] serving a group of users. Simpler versions of such interfaces seem to be applicable in the videotex context; more sophisticated natural language interfaces require the ability to understand natural sentences (or to simulate such understanding), both highly complex and time consuming tasks.

2.14 30/12/81

[illegible] 12:35 PM. 5/1/82.

✓

3. Terminology : a clarification

The following terms will be frequently used in subsequent sections. We include a brief glossary inspired mainly by [3].

Keyword : Grammatical element which conveys the significant meaning in a document. Word indicating a subject discussed in a document. In record-oriented databases a keyword may be the value of an attribute, or an attribute-value pair.

Descriptor, subject heading : Word (keyword) or combination of words having a precisely defined meaning, under which all documents on that subject are filed and retrieved from an information system (e.g. chronic tropical diseases).

Classification : Scheme for the partitioning of knowledge in a certain area displaying mutually exclusive and collectively exhaustive categories (or topics). Most classifications are hierarchical. Each entry in a classification is a descriptor.

Authority list : List of words used in indexing that provides equivalents for common vocabulary.

Thesaurus : List of words where each word is associated with other words having equivalent, broader or narrower meaning. A thesaurus can be an authority list for a particular system.

Index : Systematic guide to items contained in a collection. These items are represented by entries (e.g. keywords, descriptors) arranged in a known searchable order, such as alphabetical, chronological or numerical.

Pre-coordinate indexing (processing) : Documents that enter the database are classified according to a fixed (authority) list of descriptors; this is usually done by qualified human indexers. On retrieval, the response (documents) is found in one particular place of the index. The term pre-coordinate indicates that keywords are combined (coordinated) into subject descriptors prior to the search process. Examples are back-of-the-book indexes or the fixed arrangement of menus in typical videotex databases.

Post-coordinate indexing (processing) : Documents entering the system are described typically by a set of single words, determined by the author of the document (or by a computer algorithm). These words enter the index. Coordination of index words is performed at search time by forming logical products, sums and complements on the sets of documents found under individual index entries. The retrieval is controlled by user-supplied queries, often in interactive mode.

Full text databases : Databases which contain the full text of the book, article or report that is to be retrieved. Typically the user may search the whole database for the occurrence of particular words or phrases and then perform post-coordinate processing on the documents identified through the full text search.

The above notions are drawn from the area of bibliographical and similar non-record oriented retrieval systems. It is to be seen to what degree these techniques apply to videotex.

1:45 P.M.
31/12/81.

4. Menu-based processing

This section is concerned with a discussion of menu-based interfaces. We enter into some detail here, since frequent questions are asked about the suitability of this approach which is the central retrieval mechanism of practically all operational videotex systems.

4.1 Principle

The basic principle of menu-based processing is simple : After initialization, the system presents a sequence of menu pages, each containing up to ten numbered selections. The user indicates his choice by keying in a number, upon which the next (selected) menu will be presented until a desired document page is retrieved. Direct access to an arbitrary page is possible by keying its page identifier; the latter usually corresponds to the sequence of menu choices which lead to the page in question from the root of the tree.

Two points should be noted : (1) Stepping through a sequence of menus is a particular way of inspecting and browsing the index to a collection of documents; other methods for inspecting the same index exist and will be discussed later in this report. (2) The distinction between "index" and "document" (or data) pages is not always clear; this is particularly true in Prestel where only one type of page exists, carrying possibly a mixture of index and other information.

The discussion in this section is based mainly on evidence from the use of current videotex databases [21], and on conclusions made from a generalized menu-based system ZOG [4] developed at the Carnegie-Mellon University. ZOG can be used to create menus for any application or set of user activities such as ordering parts, paying bills or retrieving information, regardless of content. The system differs from Prestel and Telidon in that the contents of a display page are stored independently of the page format; the two are combined at database load time. It is also possible to pre-define a class of menu structures in ZOG so that given, for example, a description of a restaurant, the ZOG software could synthesize an appropriate display page and then automatically load that page into the network of menus. The connections that allow the user to search the restaurant database would be established automatically. This is in contrast to Telidon and Prestel where pages are units whose cross links must be explicitly specified. The ZOG database maintenance facilities are impressive, as demonstrated in pilot applications. However, it is basically a pre-coordinate system and the problems and advantages of this approach from the perspective of users and IP's are similar to those experienced with a Telidon-like database, as described below. However, we remark in this context the existence of simple software packages that automatically create Telidon- and Prestel-compatible databases from machine readable text sources.

4.2 Greek Restaurant example

In order to resolve the Greek restaurant problem, the information provider for a Prestel or Telidon-like database would have to structure the index into one-page menus. For example, in Fig.2 the menus (indexes) are organized so that one first chooses Restaurants from a menu of Entertainments; then from the subsequent page one would choose Inexpensive, then Ethnic Cuisine and then Greek. An alternative route to the same information (cheap Greek restaurant) might be 1.Restaurants, 2.Ethnic cuisine, 3.Greek, 4.Inexpensive. In effect many other orders are possible; the point is that in either case, the user's search pattern has to be explicitly anticipated and built into the data structure. The information provider (indexer) must therefore carefully analyze the problem and then choose and fix an access path which will be suitable for a sufficiently large number of people. The above situation is typical for pre-coordinate processing.

4.3 Retrieval utility

From the user's viewpoint menu-based systems appear to be easy to use. The user does not need to type words on a keyboard, and the permitted actions or choices are explicitly displayed on the screen. Given touch screen support (as in ZOG), the mechanical constraints on the user are reduced to pointing. This approach works well when the pattern of choices at any node of the menu network is constrained by the activity such that the number of possible choices is small (e.g. less than ten), and the meaning of choices is obvious. For instance, menus have been succesfully used as interfaces to library circulation systems [22] or in other computer command languages where the activity of the user is naturally constrained by the application context. Menus are frequently used in combination with other means of interaction such as keywords and form-filling.

The second great advantage of menu-based retrieval systems is their efficiency of retrieval (once the network of pages has been created and loaded). This is a very important factor in public databases where large numbers of users are simultaneously active.

Now we discuss some drawbacks of menu systems. In general, individually specified pages can be retrieved quickly; however, frequently the user needs to navigate the database before he finds the desired information (if ever); then the total processor and elapsed time may exceed the time required by other methods.

In some situations, such as information retrieval applications, it can happen that the number of choices exceeds ten. The resulting crowding of the screen leads to confusion and users have difficulty in choosing between large numbers of alternatives [23]. A related problem is that users easily get lost in menu networks [4]. This is due to the limited capacity of human short-term memory which cannot recall more than a few immediately preceding choices and their context. Several methods have been proposed and some implemented to alleviate this problem, like backtracking a la Telidon, user-defined checkpoints [5] and personalized page labels [24]. None of these seems to solve the problem. Another similar difficulty arises from the depth of classificatory indexes in large databases. If each menu has at most ten choices, then a database with a hundred thousand pages is at least five levels deep. This means that the user has to extrapolate downwards through four levels of terms which are branching from the general to the specific. This problem should not be minimised because the user tends to approach information retrieval with the specific rather than the general in mind (e.g. a cheap Greek restaurant). At each menu in the hierarchy the user must guess which of the possible choices might eventually lead to his particular answer. Preliminary research on a trial Telidon database [25] suggests that the average failure rate on each choice might be as high as 20%. In other words the average user has a 1-(1-0.2)**5=68% chance of finding a wrong page in an unfamiliar database with 100,000 pages, even if his choices are 80% successful at any one menu. Admittedly this is a very crude model of user behavior, but it illustrates the point that traversing long sequences of choices may lead to incorrect results.

A further constraint of effective retrieval is response time. Experiments with ZOG indicate that users tend not to browse the menus, that is, explore the database, if response time for page retrieval is longer than 0.5 seconds. This is a very discouraging result if it is confirmed, because it implies that extensive cross linking of database menus, necessary to provide multiple access pathways, will not be successful for large databases running on time shared computers with average response times in the 1-3 second range. Similar results have been reported for Telidon [25] and Prestel [26, 27].

Another set of problems derive from the limitations of the pre-coordinate indexing inherent in menu-based information retrieval systems. The Greek restaurant example illustrates some of the difficulties that the indexer has in reconciling multiple access pathways for the user with the practicalities of creating a network of explicitly connected menus. Such problems are amplified both for the user and indexer if the restaurant information is embedded in a larger network of menus with many alternative Entertainments and many more entries related to restaurants and facets of ethnic sub-cultures. Under conditions where two or more IP's may be placing relevant information into different parts of the same database and/or there are multiple databases in a videotex network [28], the retrieval problem is virtually intractable using only pre-coordinate, menu-based indexing. We note that the above scenarios may lead to situations, where a user is supposed to find information essentially without being led by an information provider.

Still further difficulties are experienced with menus based on subject heading lists because the database appears hierarchical to the user (although it may be a network), which is generally exagerated by the natural tendency of humans to think in terms of hierarchical structures. In Telidon this illusion is encouraged by the navigation commands and the database structure. However, not all kinds of knowledge and information fit well into a hierarchical pattern. This may turn out to be detrimental to the efficiency of retrieval.

4.4 Database construction and maintenance

Menu-based systems (as generally all computerized pre-coordinate systems) are costly to maintain and produce. Production of Prestel database pages averages 10-15 pages per work day [29]. That this low productivity is not due to inclusion of color and graphics is supported by results from PROMIS, a ZOG-like system which contains about 35.000 menus of text (medical information), estimated to be the result of over 100 man-years of work [30]. Given a very stable high use database, this high cost of production might be eventually recovered. But since many of the proposed videotex applications include ephemeral information, and videotex databases will need to grow rapidly to a useful size to be marketable [21], the prospects for cost-effective maintenance and production are not good for large commercial databases. Some preliminary calculations for Prestel, which still appears to be subsidized support this conclusion [27, 31].

Also of concern, particularly to Telidon, but to a lesser degree to Prestel is that the database structures are closely linked to a hierarchical tree organization. Each page has a logical number which reflects its position in the tree. Reorganization of the menus (index) alters these numbers. It is therefore not easy to use such numbers as shortcuts to particular pages because the information associated with a particular number will change over time. This problem is discussed in detail in [6]. We remark that the ZOG system does not suffer from this shortcoming because the page numbers are associated with the content rather than a particular node in the menu network. This allows the database manager much more freedom to reorganise the database and index cost effectively without affecting established user patterns or printed user aids [4, 28]. The separation of content, page structure and menu-network structures from each other as well as from the physical database structure permits this flexibility for ZOG-based applications.

4.5 Appropriate applications of menu systems

Most of the preceding discussion concerned problems of large and potentially changing menu-based databases. There exist, on the other hand, a large number of well established applications for which the Prestel and Telidon-like technology seems very suitable. The common characteristic of these applications is that they are relatively small and they feature stable question-answer patterns well known to the general public. In these services the user always expects to find the same kinds of information or options under standard headings, although the details may be varied by the information provider.

Such small and stable services can be roughly divided into two groups : retrieval and transactional. Evidently the distinction between the two is fuzzy; for instance transactional applications may involve accessing large databases, as in banking applications. In any case, the requirements on the user interface are similar and our remarks in the preceding paragraph apply equally to both groups of services.

Indexing for such stable services can be very cursory and retrieval will be quite efficient with simple subject headings and limited hierarchies within categories. Examples for information retrieval are weather reports, news headlines or the daily horoscope. The index could be displayed as a short series of menus and choices indicated with a numeric keypad. Due to their limited information domain and size, such applications seem to be ideally suited for implementation on teletext. These services are stable enough for paper indexes, such as the ones provided for CEEFAX, to facilitate access to the appropriate page.

Examples of limited transactions are reserving a theatre ticket or ordering a pizza for home delivery. For such processing the supporting system must be two-way (videotex). It is important that the number of alternatives presented to the user at any decision point is small, e.g. four sizes of pizza or eight choices for topping. Also the transaction hierarchies must be shallow, or else the user looses track of previous choices and becomes disoriented. Such transactions are easily supported by menu software which has been designed to provide pre-coordinate indexes, provided that in a large database it is easy to reach the root node for the transaction. (For example, Prestel solves the access problem to entry points by advertising the numeric address of certain important nodes.) Thus a hierarchical menu based system with a numeric keypad for user input can support simple activities like those described above without over-extending the user's tolerance of a potentially tedious and disorienting interface and without seriously distorting the natural pattern of a transaction.

4.6 Summary

In summary, we can say that the key to the successful application of menu-based systems in videotex lies in conceiving the database as a well-established collection of well-established services with little cross-linking between them.

This in turn implies a segmented database with each sub-database providing a particular service or group of related services eg. fast food ordering. The user could be directed to the correct sub-database either through a stable printed guide or through a logon procedure and account/subscription information. In this way the information span (sub-database) to be covered by the simple menu indexes remains within the scope of this technology.

5. Pre-coordinate keyword processing

5.1 Example of a system

The Teletel-STAR experimental system [5] being developed in France is a hybrid videotex system which allows the use of both menus and keywords. For a detailed comparison of the user interfaces for STAR, Prestel and Telidon, see [32]. The STAR database is organized in a unique way. The information content displays are organized in pages much like Prestel. A large number of pages can form a multi-paged document and documents can be cross-linked within a database. The unique features of STAR derive from the ability to store a program with the first page of any document. This program is executed when the document is displayed to the user, and it controls the way in which subsequent document pages are retrieved in response to the user's input. The various features of the programming language in which the program is written allows the IP to take into account not only the current page and the current menu choice but also previous choices. If the user inputs a keyword, the document control program can search for a match to the keyword in any of three dictionaries in a specified order. Each dictionary is a table of keywords where each keyword is paired with a pointer to a single page in the document or to another document. No facilities are provided for keywords to point directly to more than one page. However, this dictionary facility is unique in that keyword searches are restricted to a particular local area of the database. The aggregate of these facilities makes the STAR system responsive not only to the user's immediate context but also to the pathway by which the user arrived at the current page.

5.2 Keyword system at the Universite de Montreal

It appears that some form of keyword capability is definitely desirable to enhance retrieval operations in videotex. In order to test the idea, an experimental project is under way in our group at Universite de Montreal. The aim is to add keyword capability to a standard Telidon database system. In order to avoid lengthy searches in large global directories, it was decided to limit the size and scope of directories and to provide (for the IP) a facility for defining a directory at any desired node. When the user enters a keyword (he is free to do so at any time) while examining a given page p, then the directory to be searched is either the one associated with p (if it exists), or the one associated with the closest ancestor of p on the path toward the root page. Each directory (d) has a list of up to ten alternative directory numbers associated with it. These directories are searched when no match is found in d. There is always a directory associated with the root node of the database, whose purpose is to serve as a directory of applications (services) contained in the database. The other directories may serve users within individual applications. This system is simpler than STAR in that it has no control programs and only one dictionary is searched for a given keyword. On the other hand, provisions are made for multiple pages (up to ten) pointed to by a dictionary entry. Multiple hits are presented sequentially to the user.

5.3 The Greek restaurant example

Figure 3 shows how one might structure a Restaurant database in STAR. In the first instance it might be possible to find the Restaurant page by direct keying of Restaurant as a keyword. The first display might be a menu similar to Figure 2 in which a numerical choice could be made or a keyword entered. For example the choice of Inexpensive (cheap but chic) might lead to a page which, in turn would solicit keyword input and search that word against a dictionary of cuisine types. Each entry in this dictionary would point to a list of appropriate restaurants. Alternatively the document control program could take any keyword input in response to the first menu against the same dictionary of cuisine types. This provides a shortcut for the experienced user, protects the impetuous user from inappropriate responses and also provides a more directed path for the naive or uncertain user. It should be mentioned that "help" pages showing which keywords are expected next by the system are also available in STAR.

This technique can also be used to mimic the boolean operations of post-coordinate keyword systems. For example, if in response to the first menu the user inputs Cheap, this word might be found in a second keyword dictionary which lists synonyms or related terms. These could lead to another page soliciting further input at which point Greek would lead, via a third dictionary, to a list of Cheap Greek Restaurants. This latter directory look-up executes an implicit AND operation. If each restaurant were allocated a separate page, the control program could be used to extract any subset by way of anticipated and pre-programmed boolean operations. Note however, that the user's terminology and particularly his search strategy must be anticipated by the programmer (IP) as in menu systems.

In the above sense the STAR indexes are pre-coordinate. However, the keyword dictionaries plus the document control program allow the IP to overcome some of the deficiencies of pre-coordinate processing. For example synonym access can be relatively easily provided and new access paths can be incorporated via the control program without disturbing the document pages.

Unfortunately at the present time there is little experience available with the use and acceptance of STAR. This will gradually change in the near future when the system will be tested in field trials.

5.4 Discussion : suitability for videotex

Pre-coordinate keyword processing as in the above two examples can be a useful complement to existing videotex systems, providing shortcuts to selected points in the database or to sub-databases. However, this type of system is still too close to the underlying menu based retrieval system and it is not capable of providing sufficient precision in indexing or of supporting numerous user views and search strategies [32]. In new system designs the use of keywords should be carried further, to include post-coordinate facilities.

6. Post-coordinate keyword processing

6.1 Explanation based on commercial systems

Since the late 1950's, alternatives to pre-coordinate (classificatory and subject heading) approaches to information retrieval have been developed. These new systems are based on two assumptions. The first is that single words (keywords) extracted from the source documents can be used as tokens to represent the knowledge content of the document concerned. The most extreme form of this is the uni-term approach of Taube [33], permitting exclusively the use of single words. In practice, some keywords must be combinations of words, like for example "character string", where the meaning of the combination is different from the combination of the meanings of the words individually. KWOC systems [3] are a well known example of this.

The second assumption is that mechanical combination of document sets retrieved by such keywords can produce collections of documents that contain knowledge or information equivalent to that represented by the combination of keywords used for searching. Because the indexing method relies on combination of single terms at the time of search, such systems are called post-coordinate, often abridged to "keyword systems". (Thus the term "keyword" is generally used with several meanings!).

The clear difference from pre-coordinate systems is that indexers now have only to describe the content of the database, without having to anticipate the user's most likely search strategy. The onus is on the search mechanisms of the coordination system, usually based on boolean algebra, to allow the user to establish an ad hoc search strategy that will extract information from the database. This approach generally simplifies indexing (especially when the indexer is not the IP) and complicates searching, a form of externalisation of work from the IP to the user and the computer.

During the 1970's very large post-coordinate, keyword based systems were developed for bibliographic information. The three largest are ORBIT [34], DIALOG [35] and BRS [36]. Computerized searches of these and numerous other similar systems are now a routine service in many public, university and special libraries.

6.2 The Greek Restaurant Example

The restaurant problem could be handled by a post-coordinate system in a number of ways. A straightrforward approach would be for the restaurant owner to assign descriptive keywords to his advertisement, prompted by the IP who has some knowledge of the users' search habits. A list of keywords e.g. Restaurant, Greek, Inexpensive, Dionysian, Outremont could be stored in an inverted index together with pointers to the advertisement. At the time of search the user might type, using ORBIT style syntax :

SET1 = Restaurant AND Greek AND Inexpensive

The retrieval system would retrieve all documents that were indexed by all three words. If too many documents were found, then the user might add a further condition :

SET2 = SET1 AND Outremont

so as to reduce the number of restaurants to only those in the Outremont district of Montreal. In more advanced systems the user might be able to search the content of the advertisement for particular expressions so as to retrieve more precise information.

6.3 Discussion : flaws in current systems

The commercial success of post-coordinate bibliographic systems indicate the potential of this approach for large, diverse videotex systems. However, before approaching this subject we have to discuss some flaws and problems of current systems. Major problems for searchers derive from both the system interface and the indexing method. User command languages are the subject of the next section; here we concentrate on the second group of problems. It should be noted that technically such database systems could handle records (pages) of Telidon PDI's right now because these are just ASCII bytes. Only the terminal handling is different.

One aspect of the indexing problem is terminology. In some areas of knowledge there may be disagreement over what terms should be used to describe particular concepts or phenomena. The converse is also true in very large databases; the same word may be used by different IP's to describe unrelated phenomena. The concepts of authority lists and thesauri have subsequently been introduced in keyword systems in an attempt to control this terminological problem. Thesauri can aid a searcher either offline or online by automatically expanding a keyword search to include all synonyms to the user's input or by advising the user that similar or related information is stored under keywords of broader or narrower scope.

A related problem is that in free-vocabulary keyword indexes the searcher usually has to enter all variants of a keyword in order to maximize the amount of information retrieved (e.g. computer, computing, computed, computational, etc.). Various solutions have been proposed to overcome this and similar problems such as spelling variations (color, colour), various transliteration problems and spelling mistakes. The methods proposed include word truncation, wild-card characters, word stemming and various closeness of fit criteria [7, 37]. Similar lexical processing is found in the French Directory System [57].

A combination of these methods with thesauri will simplify the user interface in terms of mapping the user's spontaneous vocabulary to that of the more considered vocabulary of the indexer. Such techniques will have to be considered for future videotex systems right from their initial design since the algorithms are difficult to add on to large databases. However, it should be remarked that string manipulation algorithms require, in general, large amounts of computing power; this aspect is of extereme importance in public systems which have to serve large numbers of users efficiently.

6.4 Suitability for videotex

First we remark that bibliographical databases are not the only systems using post-coordinate processing. For example, many record-oriented database management systems [8] provide logical operations to facilitate retrieval, which is an essential criterion of post-coordinate indexing. In particular, relational databases may be thought of as systems where keywords are arranged in tables (relations composed of tuples). A relational search is, in essence, the retrieval of tuples (documents in bibliographical systems) which contain pre-specified logical combinations of (other) keywords in specified columns. The principal difference is that in the relational database queries and results of retrieval operations are composed of similar entities (i.e. keywords), whereas in bibliographical systems the retrieved documents are qualitatively different from keywords.

There are many other similarities between record-oriented and other types of databases; unfortunately the literature tends to treat each class separately.

Another characteristic inherent in the post-coordinate mode of processing is the dynamic synthesis (or at least formatting) of response information instead of just following one of the possible predetermined paths. On retrieval the necessary information is extracted from the database and a response is formulated according to the user's query and the interface language of the system in question. A simple example of such dynamic synthesis applied to a typical videotex situation would be the creation of a page containing the list of all cheap Greek restaurants (instead of displaying all individual advertisements sequentially).

In our opinion, the above mentioned types of post-coordinate processing are both applicable and desirable for future videotex applications. Clearly, such processing may be provided by third party databases (which are not covered in this report); however, specifically videotex oriented applications would also greatly benefit from such capability. In effect, granted the inclusion of keywords, post-coordinate manipulation of retrieved page sets comes as a natural extention, as illustrated above by the restaurant example.

N. [illegible]

Ideas from record-oriented database processing are applicable to videotex in a number of ways [6]. Beside maintaining keywords in a separate, perhaps relational database, one might use preformatted records for Prestel-like "response pages" or other form filling applications. As an example an experimental interest matching project is under way in our laboratory, employing a relational database management system RISS [9]. Standard relations are built for various "services" such as real estate or cars, and used to structure the interface. It is intended that the users input be used as a search argument for matches in a relational database of "wanted" or "for sale" relations as appropriate. Such a facility does not distinguish between users and IP's; it provides a matching process to facilitate personal transactions, truly two-way videotex.

N.B.

6.5 Efficiency problems

From the above discussion it follows that there are two problem areas which have to be addressed before some form of post-coordinate processing can be included in large-scale commercial videotex systems. The first is that of the easy and friendly interfaces that such systems should present to their (mostly naive) users. We deal with interface languages in the next section; however, here we can say that a large body of theoretical and experimental work exists in this area, since other systems also recognize the necessity for habitable interfaces. It seems that suitable interfaces can be established for the small to medium complexity retrieval and transactional applications envisaged for inclusion in near future videotex systems (see Figure 1).

The second problem area is that of efficiency. Since videotex has to serve many users at low cost and fast response time, efficiency becomes a prime concern in such large public systems. Moreover, it seems that many deficiencies of keyword systems mentioned above as well the requirement for acceptable user interfaces can be translated into requirements for increased computing power. We believe that in this context the answer lies in multiprocessing, which is inherent in distributed videotex networks. The overall tendency toward distribution in such systems was discussed in [2]. A natural way of structuring distributed processing is the provision for "videotex access machines" (VAM) which would execute a number of user-specific functions (such as logon, statistics, accounting, handling of the user context), for a limited number of users. Interaction between VAM's and the main database(s) would be on a strictly transactional basis (only individual pages requested and returned)[2]. In such a distributed system the main burden of processing keywords and implementing a suitable interface would be placed on VAM's. These can be built modularly with as much microprocessor-based computing power as necessary for a given set of functions and a given number of users.

The above mentioned interest matching project is actually a development in this direction since all relational processing is done in a separate remote computer. In this respect one should also keep in mind the developments of special-purpose, high performance hardware for relational and string-matching purposes [10], [37].

7. Interface languages

7.1 Introduction and classifications

The interfaces outlined in section 2, Figure 1, seldom appear in practice in their "pure" form. In real data retrieval and transactional systems they are typically embedded in an interface language with its own syntax and semantics. These languages have different names such as command languages, query languages, interrogative languages, data languages etc. Each of these have their particular flavor, but their application areas generally overlap and bear in one way or another on videotex technology. For this reason we will simply use the term "interface language" and attempt to classify them according to the following important characteristics and criteria.

a) Artificial vs. natural language interfaces. Most of the actual languages are artificial, for the simple reason that they are close to the programming languages used to implement the underlying system (sometimes the interface language is an integral part of a data retrieval system). Therefore it takes less programmer effort to design the language. Frequently such languages suffer from a lack of user-friendliness and they are hard to learn and to remember especially for the naive user. Natural languages on the other hand require complex processing in order to translate natural sentences into simpler and unambiguous commands acceptable to the rest of the system.

b) Involvement of artificial intelligence (AI) techniques in the process of interpretation and execution of interface language statements. This applies mostly (but not uniquely) to natural languages. AI can be used for parsing, understanding (or rather simulation of understaniding) the sense of a statement, and for searching and representing data in sophisticated knowledge bases and expert systems. An excellent overview of this area is found in [19].

c) Levels of system-user interface. An increasing number of systems now feature more than one level of interface language. One way of creating multiple levels is to provide simply nested subsets of available commands; only the simplest and most frequently used facilities are available on the first level, etc. This is analogous to various "easy" subsets of PL/1, Pascal and other programming languages. Another, and more important method for multiple interfaces consists of inserting a "user agent" between the user and the database thereby creating two interfaces : one between the database and the agent and the other between the agent and the user. The former (called database interface) can be optimized for database interaction and typically has a concise syntax and command structure. It is primarily an interface for use by the user agent and other application programs, but it may also be also available to the expert user. The other interface (user interface) is designed for the casual user with friendliness, robustness, ease of use and similar human factors in mind; the user agent then becomes in effect a translator between the two language levels. In addition, the notion of user agent lends itself naturally to system distribution : it can be implemented in intelligent terminals or in videotex access machines (in which case they are shared by multiple users).

d) Procedural vs. specification interface languages [11]. In a procedural language the programmer has to describe how to obtain the given objective; in specification languages one has to describe (specify) the objective, but not the method to obtain it. The difference between the two is often blurred. As an example, we can say that data retrieval by Telidon-like menus is more procedure-oriented because the user has to give a sequence of choices, that describe how to get to the desired data. Retrieval by keywords is specification oriented because a query specifies essentially only what should be retrieved. It is not clear which of these two types of languages is more suitable for videotex; it is possible that this parameter is irrelevant in the long view.

e) Retrieval vs. transactional languages. Here again the distinction is blurred and as with interactive systems in general, it is safer to speak of (retrieval or transaction) "orientation" or "flavor". Transactional applications usually involve some "side effects" (such as delivery of ordered merchandise or transfer of money), in addition to consultations of a database. However, we remark that this report is primarily aimed at database retrieval systems and it does not fully cover interfaces to interactive systems.

For the purposes of this study, points a) to c) above are the most relevant, although examples and references to all types of systems are found in the following sections.

7.2 Artificial interface languages

Classical examples of this class of languages are the control languages of general-purpose computer systems. These languages incorporate a large number of commands intended to direct, in addition to data and file oriented functions, a number of other activities such as program execution, resource management, etc. Computer control languages are notorious for their rigid and unforgiving syntax, conciseness and cryptic, hard-to-remember symbols. They are designed for efficient use by experts. Of course, limited subsets of control languages can be easily learnt by the casual user of general-purpose systems.

The situation is somewhat different with specialized database and data retrieval systems. Here commercial considerations dictate that the systems be easily usable by nonexpert users and considerable amounts of work have been invested in various types of user-friendly, easy-to-learn and error tolerant interfaces [11][20]. Evidently this is extremely important for videotex applications where the typical user is an average TV viewer and the possibilities for user instruction appear to be limited.

We now proceed to a brief description of some known implementations of database interface languages which have been designed with ease of use in mind. One frequently used approach is the use of menu selections to prompt the user, eg.

1. SEARCH 2. PRINT 3. LOGOFF etc.

which can be implemented with ZOG. A second possibility is to use a default interface as in the QL (Quick Law) system [39]. QL is a full text database system with a default mode of FIND together with a boolean default of AND. A request may consist solely of several keywords separated by commas, like

Greek,Restaurant,Outremont

which is interpreted to mean "Find all texts with the words Greek, Restaurant and Outremont". Text output is automatically sorted and ranked according to the aggregate frequency of occurrence for the keywords so that text containing all three terms would be listed first. This technique also works well on the Globe and Mail newspaper database [40]. Since its inception QL has also added explicit boolean algebra based query commands for citation databases. Telidon-like user interfaces can be seen similarly as default based languages.

A number of interesting interfaces have been created to relational databases. A microcomputer based system called WHATSIT [41] stores information in 3-tuples of subject, tag and object, eg. <name>, phone, <7-digit number>. Separators are either "′" or "s", giving a syntax such as :

"Bob′s phone′s?" reply : "Bob′s phone′s 737-8765".

WHATSIT can match on any domain of the 3-tuple for retrieval and the user can add new tuples at any time by typing in the new information. Even though WHATSIT′s syntax is too simple to deal with problems like the Cheap Greek Restaurant, it's approach is certainly worth pursuing further as it seems to be suitable for simple videotex applications eg. personal file service. We remark that similar attribute-object-value systems have existed for some time eg. LEAP [42].

Another well known relational database management system is System R [8]. Several interface languages have been concieved for interactive applications of R (the system can also be accessed by application programs running in batch), notably SEQUEL-2 and SQUARE. SEQUEL-2 [12] is a powerful language equipped with commands for retrieval, data creation and update, manipulation of relations and control of user access rights. Queries with attribute-value pairs and boolean operators are embedded in SELECT-type commands for retrieval. Additional facilities for the specification of queries are available such as built-in functions SUM, MAX, AVG. These are applicable in cases where the query specifies calculations involving a number of tuples possibly belonging to several relations, eg. to find the average salary of all employees in a department, or to find the cheapest Greek restaurant in Outremont. SEQUEL-2 is an example of a "non-positional" language, in that different parameters of a command can be entered in any order, each preceded by an identifier uniquely identifying a parameter.

SQUARE [13] is a language which is nearly identical to SEQUEL-2 in its capabilities; however it has a different kind of syntax called "positional". Here the composition of a command is a combination of menu selections and form filling; typically the system displays to the user a preformatted screen much like a questionnaire where different fields have to be filled in to complete the command. This kind of interface could be created using "action pages" available on Telidon.

SEQUEL-2 and SQUARE were compared in a psychological study as to their learnabilty and acceptance by both programmers and non-programmers [8]. The results indicate that SEQUEL-2 is slightly preferred by both groups; however this evidence is not conclusive and it should not be implied that "non-positional" syntaxes are preferable to positional in general. On the contrary, it would appear that the form filling approach is preferable in often mentioned potential videotex applications like advertising, interest matching and messaging. This seems to be supported by the success of Query-by-example [8] or VISICAL [43], which are two-dimensional, screen oriented positional languages.

Participants in the SEQUEL-2/SQUARE experiment suggested that SEQUEL-2 could be made more user-friendly by making it layered and by providing for spelling corrections (using synonym directories or word stemming).

An interesting example of a two-level interface to bibliographical retrieval systems is described in [15]. The systems in question are the MEDLINE, TOXLINE and CATLINE databases running under the ELHILL-III database management system of the National Library of Medicine. The three databases use a sophisticated command language with many facilities; a user-friendly version is implemented by intelligent terminals essentially by excluding some complicated commands, using form filling techniques and providing a simplified display of search results. In this context one should mention a succesful project reported in [16] aimed at creating a common retrieval language (user interface) for a group of bibliographical databases, each having different database interfaces. One can envisage the possibility of creating a similar common interface for various videotex systems in North America.

7.3 Natural language interfaces

The exciting possibility of communicating with a database system in natural language (or rather some subset) is recieving considerable attention in the database community. In a recent overview [44], 52 ongoing or completed projects are reported on the subject. According to the majority of authors natural language is the ideal means of communication with the computer; however a more cautious approach is found in [11].

Most natural language interface systems fall in two large groups : a) Two-level interfaces, where the database in question is a more or less standard system with an associated command language; the natural language input from the user is translated by the user agent. The latter then can be considered as an autonomous module suitable for distributed implementation; b) Question answering and expert systems, which are knowledge bases containing complicated forms of information represented implicitly, that is, answers to questions (queries) have to be synthesized by highly nontrivial deductive processing and searching. Both of these groups frequently rely on artificial intelligence techniques mostly in the areas of natural sentence parsing, dealing with ambiguities, data representation and search strategies (in group b). An excellent review of AI applications for data retrieval is found in [19].

The typical pattern observed at the user interface is a "clarification dialog" serving to elucidate the precise meaning and intention of a query. A simple example of such a system is SMART [45]. The system emphasizes feedback from the user as a means of refining and improving the accuracy of bibliographic searches. Although SMART is basically a command language system, it allows the user to enter value judgments about previous searches, and these judgments are automatically included into the parameters for subsequent searches. An example of feedback is assigning weights to keywords that are judged relevant to the search. Another is the automatic incorporation of additional keywords derived from citations but not used in the user's original query. SMART is a laboratory system but attempts have been made to apply these principles to the MEDLINE database. No data are available on the user acceptance [46].

A further example of the two-level interface concept is GUS (Genial Understanding System) [47]. It belongs to a class of systems based on the concept of frames [48]. GUS analyses natural language input to discover variables that fit into its various frames. The frames in turn guide the GUS programs to solicit further information to complete the frame. Once this is done, GUS can carry out an action or move on to another frame. In one application it was used with an airline reservation system to demonstrate the usefulness of the frame approach. Here a typical frame represents a flight and contains slots or variables for passenger, departure point, destination, date, etc.

The habitability of GUS is improved by its ability to cope with ambiguities in the English language. For example, it can respond correctly to inquiries such as "Is there an early flight?" or "I'll take the last one" or "How about next week?" GUS approaches the expert systems discussed below in its apparent intelligence; however its competence is limited to the information domain encompassed by its frame system, which is not easy to change or gradually modify.

An attempt has been made to produce a generalized frame system. The programming language called TAXIS [49] supports the building of specific frame-based applications in much the same way as ZOG supports menu interfaces.

ROBOT [17] is a commercially available natural language translator (user agent). It accepts the user's inquiry in English, produces a number of possible parses and then proceeds to consult the database to choose the most likely interpretation. If there is no clear winner, the user is asked to clarify the query. ROBOT handles relatively simple inquiries in a restricted commercial-style database. For example, it can process the following queries succesfully : 1. "Singles in Chicago?" 2. "Who are the secretaries?" 3. "Which of them live in Detroit?" The latter question yields a list of secretaries living in Detroit rather than the reply "No singles in Chicago live in Detroit." Interesting resource utilization data are available for this system : ROBOT requires 200 kB of memory and takes <1 sec to process a typical query (not including data retrieval).

Next we discuss briefly question answering and expert systems. The emphasis of these systems is on the ability of the system to answer questions, recommend actions or diagnose conditions rather than on natural language understanding. However, the extant expert systems such as MYCIN [50] or PROSPECTOR [51] do have natural language interfaces even if the language is a little stilted.

As an example, MYCIN is a suite of computer programs that "uses the clinical decision criteria of experts to advise clinicians... regarding selection of appropriate anti-microbial therapy"[50]. It is designed to make available the expertise of the best and most experienced diagnosticians to less experienced or less specialized practitioners.

Question answering systems are generally designed to accept natural language input and generate actual answers rather than references (citations). Some are designed to answer questions directed at a known database of restricted content, while others are attempts to produce more complete systems that "understand" not only queries but also their associated database contents. Codd [52] suggests that such interfaces should have the following capabilities : 1. be able to clarify the user's request,

2. restate the user's query, 3. separate query formulation from database search, 4. employ menu selections as a fallback and 5. provide a definition capability for unknown concepts. Rosenberg [53] adds that such an interface program should ideally be domain (content) independent and interfaceable via standard protocols to a variety of databases.

Although most authors consider natural language interfaces generally more desirable for interaction with humans than other types of interfaces, a more balanced view and a short list of disadvantages is found in [11]. These include : 1. long clarification dialogs, 2. excessive processor overhead, 3. unrealistic and ambiguous questions from the user, and 4. false impression of large machine intelligence.

7.4 Interface languages in videotex

It is not easy to evaluate the applicability and potential impact of this great variety of interface languages to videotex. There seem to be two groups of criteria for such an evaluation. The first is related to human factors such as friendliness, error tolerance and ease of learning; evidently these criteria are eminently important in an environment where the typical user is the nonexpert nonprogrammer, possibly without any inclination towards understanding the working of the system. The second kind of criteria concerns efficiency and resource utilization issues.

Another issue to be mentioned here relates to the fact that people are able to learn and manipulate quite complex systems and machines when they are sufficiently motivated (economically or intellectually). Therefore the problem of whether a given interface mechanism is applicable, or "simple enough" for videotex is closely linked to the system provider's ability to find useful and interesting applications. It is therefore very important that the interface not be studied in isolation from the application (see also Figure 1).

In the realm of artificial interface languages it seems certain that some form of keyword based querying capability (with an appropriate language) is desirable for videotex. For example, a WHATSIT type system could be provided for the management of personal files of users. Nonexpert students using SEQUEL-2 and retrieval systems like ORBIT and DIALOG have acquired sufficient skill to accomplish simple searches similar to the Greek restaurant example in very short time [54]. A similar success with computer naive users of EIES has been reported [55]. This computer conferencing system has a mixed menu and command language interface which has been readily mastered by that ubiquitous woman-in-the-street, the secretary [55]. This demonstrates that a number of relatively powerful interface languages are within the reach of the layman user; it is up to the service providers to find suitable applications.

The situation is somewhat different for natural language interfaces. These are designed with easy use in mind and the main problem becomes that of efficiency when used by many subscribers simultaneously. Also most systems are not yet mature enough to be applied commercially.

One can see many applications of existing interfaces to videotex. GUS type interfaces could be used with all kinds of reservation systems, like restaurants and travel. For travel agents and travellers it might lead to individual package tours; for the restaurant patron it could mean automatic booking of a window table and a preferred menu. More generally, such natural language interrogative interfaces could be applied to fairly complex transaction oriented services such as negotiating a consumer loan or shopping for "something for my nephew's birthday".

Expert systems such as MYCIN might be applied in the videotex context to any field of consulting eg. small business taxation or career counselling. Access to such databases through a videotex network could lead to use by para-professionals or the general public, perhaps making store-front medicine or even "bare foot doctors" a reality.

A different kind of possibility emerges for large scale videotex networks with multiple databases linked by a communication network. In fact a prototype of this kind has been developed for the U.S. Navy [56]. This system called LADDER (language access to distributed data with error recovery), provides access to various Navy databases as an aid to planners and decision makers. A videotex application for this kind of approach might be an automatic banking system for multi-state or multi-national banks which would allow a user to move funds between branches, accounts or even currencies. Analogous situations exist in the commodities market, wholesale or retail trade, or inter-library networks.

The key to successfully dealing with performance problems inherent in most natural language interfaces (1-20 seconds of processing time per query translation) is our ability to modularize, distribute and if necessary, duplicate system functions. As we stated above, this is relatively easy to achieve with two-level interfaces or with keyword systems where indexes and index lookups might be implemented separately from the main databases.

As we have seen, a great number of implementations and proposals exist for improving the interface to interactive systems in general; however, only few of them are specifically aimed at videotex. We can name the following references : A microprocessor-based intelligent terminal ETA [18], realizing a natural language interface for Telidon; the French telephone directory system, using a simple form-filling technique, and a command language for keyword searches recommended in a recent report [58].

We wish to complete this section with an observation : in our opinion too little research is being done in order to find out the reaction of the public to novel approaches, applications and interfaces for videotex. Almost all work has been concentrated on the tree-structured databases and menu choices and almost none on testing simple but different approaches like keyword or record oriented databases. Much more psychological and field-trial type of experimentation is needed. As a result of such research one would expect new versions of interfaces and databases to emerge, not just the transplantation of existing systems to videotex. After all, the tree structured page oriented database itself is non-orthodox and doesn't fit into the framework of any other standard database model.

8. Conclusions and recommandations

1. Tree-structured databases with numerical menu selections such as used in today's videotex systems, are sufficient only for small, simple and stable applications. It may be that such databases will continue to be acceptable for teletext applications. However, they are too restrictive for situations where the database content changes rapidly and/or has numerous cross-links.

2. There exist a large number of database interface projects whose purpose is to develop friendly interfaces, easy to use by computer naive and inexperienced persons. Most of these results are applicable in the videotex context. Notably the concepts of two-level interfaces (user agents) and the use of natural language are worthy of further consideration.

3. Figure 1 shows some systems discussed in this report, in the continuum of interfaces. An evident conclusion from this figure is that keyboards will be a necessity in future videotex systems.

4. Simple and relatively powerful interface languages permitting post-coordinate processing of keyword queries exist which could be superimposed even on existing menu based databases. However, in order to take full advantage of such facilities, applications should be designed with the keyword access in mind.

5. Full advantages of post-coordinate processing can be realized only by building new and specifically videotex oriented

databases and interfaces. The advantages include flexibility of retrieval specification by the user and avoidance of large pre-coordinate indexes or fixed links, which are difficult and time consuming to maintain for the information provider.

6. Psychological acceptance tests on large numbers of users should not be limited to present videotex databases, but should include new systems such as mentioned above. In our opinion the tests should be conducted with interesting and useful applications which motivate and stimulate the user to learn new system features.

7. Record oriented, eg. relational databases are useable in videotex in a number of ways, such as interest matching, handling of personal files or keyword processing.

8. Special consideration should be given to the issues of system distribution and networking. Creation of friendly interfaces is technically feasible, but they often require extensive computing power. The resulting efficiency problem can be resolved by placing user agents (interface language translators) in videotex access machines or intelligent terminals.

9. Although performance issues are not discussed in detail in this report, two points bearing on the disk acess bottleneck, typical in large videotex systems, should be mentioned. (a) I/O problems can be alleviated to a certain extent by the use of multilevel memories; but such approaches require detailed knowledge of access patterns typical for large user populations. Since almost nothing is known in this area, we recommend that

appropriate tracing and monitoring be done during the upcoming field trials. (b) It should be clear that the use of keywords for database search will not only improve the user interface, but also reduce the number of page acesses required to obtain a document. This in turn may lead to less expensive system designs, which is vital for videotex applications.

10. The possibility of creating a standard user language to access all common videotex systems by menu selections should be studied.

11. One or more experimental system incorporating some new approaches discussed in this report should be constructed, eg. along the following lines : a) access to pages based on a mixture of menu selections and post-coordinate querying using keywords; b) provision for dynamic page creation at retrieval time; c) inclusion of a simple, friendly user interface by means of a user agent in intelligent terminals or access machines, which will help clarify the user's wishes; d) allow evaluation of the problems associated with handling the keyword searches in VAM and communicating with the main database(s) in a page request (transaction) mode.

1 International Videotex Teletext News, Feb. 1981, p.4.

2 A.J.S. Ball, G.V. Bochmann and J. Gecsei "Videotex Networks", Computer, Vol. 13, No. 12, Dec. 1980, p.8.

3 E.H. Brenner, ed : "Indexing in Perspective", Unesco/Unisist, 1979.

4 G. Robertson, D. McCracken and A. Newell, "The ZOG Approach to Man-Machine Communication", Tech. Report CMU-CS-79-148, Carnegie-Mellon University, Oct. 1979.

5 D. Le Moign, "Presentation de STAR", Tech. Report, CCETT, Rennes, France, 1980.

6 F.W. Tompa, J. Gecsei and G.V. Bochmann "Data Structuring Facilities for Interactive Videotex Systems", to appear in Computer.

7 Kowalski, "Approximate String Matching", Computer Surveys, Dec. 1980.

8 C.J. Date, "An Introduction to Database Systems", Addison-Wesley 1977.

9 D.J. McLeod and M.j Meldman, "RISS : A Generalized Minicomputer Relational Data Base Management System", Proc. NCC 44, May 1975.

10 Fifth Workshop on Computer Architecture for Non-Numeric Processing, SIGIR, Vol. XV, No. 2.

11 B. Shneiderman, "Improving the Human Factors Aspect of Database Interactions", ACM Trans. on Database Systems, Vol 3, No. 4, Dec. 1978, pp. 417-439.

12 SEQUEL-2, IBM Journal of Res. and Dev., Nov. 1976.

13 R.F. Boyce et al, "Specifying Queries as Relational Expressions : SQUARE", Proc. ACM SIGPLAN-SIGIR meeting, Gaithesburg, Nov. 1973.

14 P. Reisner, "Psychological Evaluation of Query Languages", IEEE Trans. of Software Engineering, 1977 p. 218.

15 C.M. Goldstein, W.M. Ford "The User-Cordial Interface" Online Review, Vol. 2, No. 3, 1978, pp. 269-275.

16 R.S. Marcus, J.F. Reintjes "The Networking of Interactive Bibliographic Retrieval Systems", MIT Report ESL-R-656, March 1976.

17 L.R. Harris, "User Oriented Data Base Query with the ROBOT Natural Language Query System", Int. Journal of Man-Machine Studies, 9, 1977 pp. 697-713.

18 J. Bradford, T. Pietrzykowski "The ETA Interface Research Report CS-80-27, Dept. of Computer Science, University of

Waterloo, May 1980.

19 L.C. Smith, "Artificial Intelligence Applications in Information Systems", Annual Review of Information Science and Technology, Vol. 15, 1980, Chapter 3.

N.B.

20 P. Hayes et al., "Breaking the Man-Machine Communication Barrier", Computer, March 1981, pp 19-30.

21 Larrat, Richard, Ed., Inside Videotex : The Future -- Now, Toronto, Informant, 1980.

22 GEAC On-Line Library Circulation System : Community Access Package, Markham, Ont., GEAC Canada, 1979.

23 D.A. Phillips, et al., "Telidon Behavioural Research I", Ottawa, Dept. of Communications, Canada, 1980.

24 J. Gecsei and G. Bochmann, "Database Structures for Videotex Applications" in First Montreal Workshop on Videotex Technology, Montreal, Université de Montreal, 1980, p. 46-48.

25 E. Lee, Telidon Behavioural Research Group, Dept. of Communications, Ottawa, personal communication.

26 T. Stewart, "Economics of Videotex", Link Research Report I (7) 1980.

27 I. Young and I. Gray, "The Cultural Applications and Implications of Videotex Services in the U.K.", Strasbourg, Council for Cultural Corporation, DECS/DS(80)11-E, 1980.

28 Prestel Business Directory, London, Financial Times Business Publishing, 1979.

29 M. Lane, "Images : How Information Reaches the Screen" in Inside Videotex, Toronto, Informant, 1980, p. 52-58.

30 G. Robertson, D. McCraken and A. Newell, "The ZOG Approach to Man-Machine Communication", Carnegie-Mellon University Dept. Computer Service Report CMU-CS-79-148.

31 J. Martyn, "Prestel and Public Libraries", Aslib Proceedings 31(5), 1979, 216-236.

32 A.J.S. Ball, "Videotex : Chimera or Dream Machine", CLJ, 38(1), 1981, 3-15.

33 M. Taube and Associates, "Studies in Coordinate Indexing", Washington, D.C. Documentation Inc., Vol. 1, 1953.

34 ORBIT, Systems Development Corporation, Santa Monica, California.

35 DIALOG, Lockheed Information Services, Palo Alto, California.

36 BRS, Bibliographic Retrieval Services Inc., Scotia, New York.

37 R.M. Lea and E.J. Schuegraf, "An Associative File Store Using Fragments for Run-Time Indexing and Compression", preprint from E.J.S., St. Francis Xavier Univ., Nova Scotia.

38 D. Le Moign, "Presentation de Star", CCETT, France, 1980.

39 QL Systems Ltd., Ottawa, Canada.

40 Info Globe, Toronto, Canada.

41 Marrill, Lyall, Jr., "WHATSIT, A Self Indexing Cross Referencing Data Query System", San Francisco, Hardhat Software, 1980.

42 P.D. Rovner and J.A. Feldman, "The LEAP language and data structure", M.I.T. Tech. Report, M.I.T. Cambridge, Mass. 1968.

43 VISICAL C, Personal Software Inc., 1979, Los Angeles, California.

44 J. Minker, "Information Storage and Retrieval", A Survey and Functional Description SIGIR Forum, XII(2) 12-108.

45 G. Salban, "The SMART Retrieval System : Experiments in Automatic Document Processing", New York, Prentis Hall, 1979.

46 T.E. Doszkocs and A. Rapp, "Searching MEDLINE in English", Proc. Amer. Assn. Info, Ser., 16, 1979, 131-138.

47 D. Bobrow et all., "SUS, a Frame Driven Dialog System", Artificial Intelligence 8(2), 1977, 155,173.

48 M. Minsky, "A framework for representing knowledge" in Winston, Patrick Henz, The Psychology of Computer Vision, New York, McGraw-Hill, 1975.

49 J.L. Benson, "Dialog Organisation and Structures for Informative Information Systems", Tech. Rept. CSRG-108, Toronto, Compt. Syst. Res. Grp, Univ. of Toronto, 1980.

50 E.K. Shortliffe, et al., "Computer Based Consultations in Clinical Therapeutics : Explanation and Rule Acquisition Capabilities of the MYCIN System", Comp. Biomed. Res. 8, 1975, 303-320.

51 W. Stockton, New York Times, reprinted, The Gazette, Montreal, Dec. 13, 1980, p. 24.

52 E.F. Codd, " A Relational Model of Data for Shared Data Banks", Comm. ACM 36(6), p. 377-397.

53 R.S. Rosenberg, "Approaching Discourse Computationally : A Review", Technical Report 79-13, p. 51-53, 1979, Dept. of Computer Science, Univ. of British Columbia.

54 S.P. Harter, "Information Provided by Library Schools in Machine Readable Bibliographic Databases". Am. Soc. Info. Sci. Amer. Proc. 14, 1977, 49.

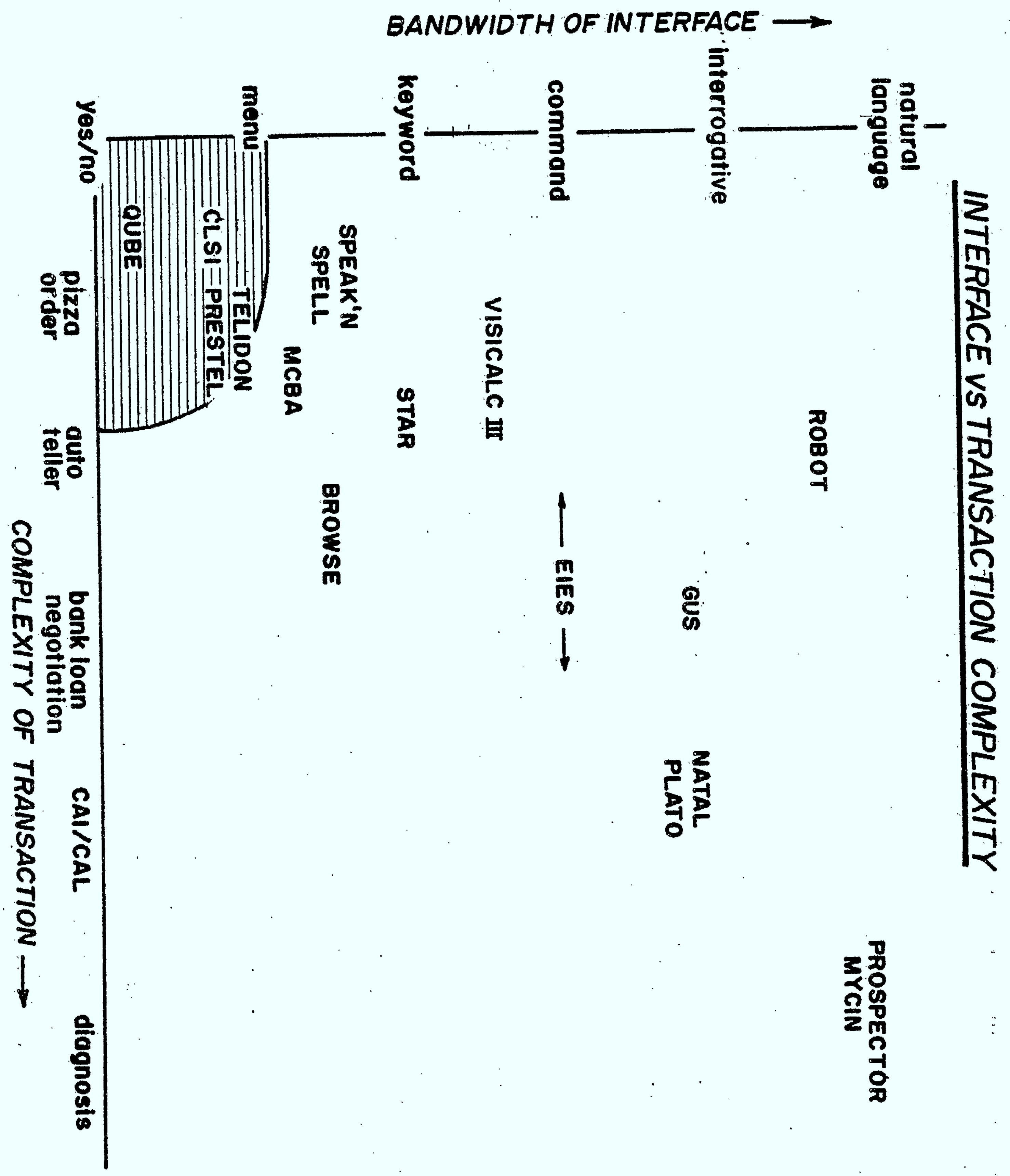
55 R. Hiltz-Star and M. Turroff, "The Network Nation" : Human Communication by Computer", Reading, Mass., Addison-Wesley, 1978.

56 G. Hendrix et al., "Developing a Natural Language Interface to Complex Data", ACM Transactions on Database Systems 3(2), 1978, 105-147.

57 The French Electronic Telephone Directory, personal communication of D. Le Moign.

58 A.H. Schabas et al., "Keyword indexing", Report submitted to Dr. E. Lee (DOC), March 30, 1981.

FIGURE 1


INTERFACE VS TRANSACTION COMPLEXITY
BANDWIDTH OF INTERFACE →
natural language
interrogative
command
keyword
menu
yes/no
COMPLEXITY OF TRANSACTION →
pizza order
auto teller
bank loan negotiation
CAI/CAL
diagnosis
QUBE
CLSI PRESTEL
TELIDON
SPEAK'N SPELL
MCBA
STAR
VISICALC III
BROWSE
EIES
ROBOT
GUS
NATAL PLATO
PROSPECTOR MYCIN

LEGEND, FIGURE 1 : INTERFACE VS. TRANSACTION COMPLEXITY

Here we have attempted to plot existing commercial and experimental information systems in a two dimensional space. The "bandwidth" is a subjective measure of the rate and quantity of information that is passed by the man-machine interface; this axis corresponds to the range of user interfaces in section 2. Not surprisingly there are a spread of systems along the diagonal from YES/NO to natural language systems. What is significant in this context is that it is easy to find examples that plot above the diagonal but we were unable to discover examples which would plot below the diagonal. This has particular significance for videotex where many authors predict the use of low bandwidth interfaces, eg. numeric pad based, for delivering sophisticated services, eg car buying. EIES, a computerized conferencing system can encompass a number of positions on the graph because it has both simple and complex interfaces and is designed to facilitate transactions between people. Perhaps this is a more useful model for videotex than any other operational system discussed.

In addition to the systems listed below, Figure 1 also shows four videotex systems, QUBE, STAR, TELIDON, PRESTEL together with an estimated "numeric key pad zone" hatched area.

CLSI : menu-based, alphabetic index online library catalogue, touch screen. CL Systems, 1980.

SPEAK'N SPELL : computerised spelling toy for children. Texas Instr. 1979.

MCBA : a range of menu-based business packages for general ledger and accounting. MCBA, 1977.

BROWSE : application of the ZOG system plus keyword option to library catalogue.

GUS : an airline reservations application of GUS.

NATAL, PLATO : computer assisted education systems.

ROBOT : natural language interface, database front end.

PROSPECTOR/MYCIN : "expert" systems that can deduce solutions to problems from evidence supplied by operator.

VISICALC III : command based system; user establishes own menus and prompts for reuse; microcomputer based.

FIGURE 2

TELIDON, MENU EXAMPLE

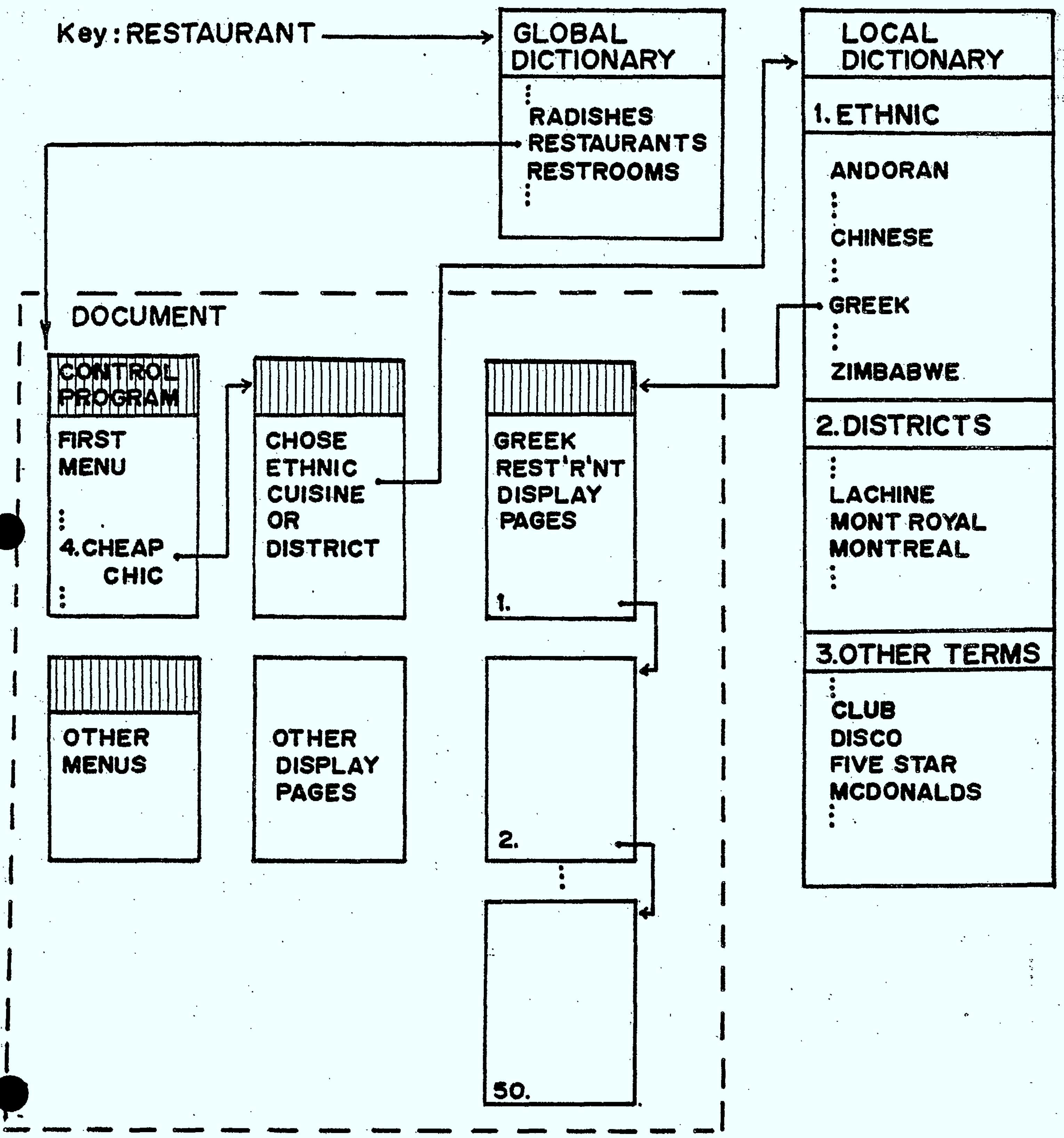
KEY	MENU DISPLAYS	PAGE N
9	ENTERTAINMENT	9
	1. CINEMA	
	2. DISCOS & ROCK CLUBS	
	3. BARS & PUBS	
	4. LIVE THEATRE	
	⋮	
	7. RESTAURANTS	
	⋮	
7	RESTAURANTS	97
	1. FAST FOODS	
	2. HAUTE CUISINE	
	3. HEALTH FOOD	
	4. CHEAP BUT CHIC	
	5. VEGETARIAN	
4	CHEAP BUT CHIC	974
	1. BY CITY DISTRICT	
	2. BY ETHNIC CUISINE	
2	CHEAP CHIC – ETHNIC CUISINE	974
	ANDORAN 9742.1	
	⋮	
	CHINESE 9742.16	
	⋮	
	GREEK 9742.25	
	⋮	
	JAPANESE 9742.47	
	⋮	
	ZIMBABWE 9742.99	
9742.25	GREEK RESTAURANTS	9742.
	ACROPOLIS – AUTHENTIC; BELLY DANCERS 2268 OSLER, RESERVATIONS...	
	⋮	

LEGEND, FIGURE 2 : TELIDON MENU EXAMPLE

This diagram illustrates one solution to the Greek Restaurant problem that could be implemented on Telidon or Prestel. The numbers and arrows on the left indicate the number keyed by the user, followed by the display of the next menu. The page numbers on the right correspond to the node in the Telidon tree at which the display page is stored. Both implicit page selection (first three choices) and explicit page selection (last choice) are illustrated.

FIGURE 3

TELETEL/STAR EXAMPLE



LEGEND, FIGURE 3 : TELETEL/STAR EXAMPLE

The first key RESTAURANT is typed by the user in response to a general instruction menu displayed at the beginning of the session. This keyword is searched in a GLOBAL dictionary that indexes DOCUMENT(S). Control of the user session is transferred to the CONTROL PROGRAM of the first page. It displays a menu page similar to that of Figure 1 (#97). A choice of CHEAP CHIC transfers control to a second program which instructs the user to key in a DISTRICT or ETHNIC CUISINE. The keyword entered is searched against the LOCAL dictionaries in the order indicated. When the keyword GREEK is detected control is transferred to a third CONTROL PROGRAM which then displays a linked list of advertisements for Greek Restaurants who's owners think their establishment is cheap but chic.