Special-purpose hardware and effective information processing

Maristella Agosti discusses the development of new architectures for information processing systems

The main aim of this paper is to study the characteristics of current information processing systems and to consider if the introduction of special-purpose hardware can increase the range of functions supported by future systems. The present nonnumerical applications of information processing systems are reviewed and a spectrum of applications is introduced to cluster the systems by the nature of data they manage. The systems that present the most unresolved problems of management are identified and a study of their functional requirements and characteristics is proposed; the functionalities for their effective implementations are underlined. Special-purpose hardware that is being proposed in the literature is considered and a grouping of its different architectures is introduced to reflect its differentiation from the von Neumann architecture. The special-purpose hardware devices that are being developed are considered, to show what aspects of future information processing systems their introduction in computer systems can improve.

Keywords: information processing systems, information systems applications, database management, information retrieval, special-purpose hardware

The main aim of this paper is to study the characteristics of current information processing systems, and to consider if the introduction of special-purpose hardware can increase the range of functions supported by future systems. Attention is concentrated on the study of nonnumerical applications. These have as their primary objective the management of data for information processing systems.

Considered in terms of the structure of the data and of the types of processing supported, information processing systems represent a very large class. Consequently, a categorization is provided. The systems that have been studied least are those systems with the least formal structuring of their data, such as office automation systems, and they are identified as less structured data systems (LSDS).

To achieve the aim of the paper, it is necessary to show the distinctive characteristics of LSDS in contrast to other types of information processing systems. The characteristics that distinguish LSDS and their functional requirements are also discussed.

The characteristics of the special-purpose hardware that is being designed and developed for use with orthodox computer systems is examined, and a functional classification is introduced to group the special-purpose hardware devices by their main functions. Their impact on conventional computer systems by improving the systems' capabilities for information processing is discussed.

The role of special-purpose hardware in LSDS is described and its probable effect on such systems in the next decade is considered.

University of Padua, Istituto di Statistica, Padova, Italy
Received 23 September 1983

INFORMATION PROCESSING SYSTEMS

This paper concentrates on nonnumerical applications. These applications have as their primary objective the management of data for information processing systems, in contrast to numerical applications which use the computer principally to perform calculations.

The distinctive features of numerical applications in respect of the computational resources they need are:

- a fast processor for performing calculations,

 0144–817X/84/010003–12$03.00

- very large amounts of available computational time (in multiuser environments),
- large main memory areas for the storage of partial results,
- very little I/O and secondary storage utilization.

In contrast, the characteristics of nonnumerical applications in respect of their utilization of computer resources are:

- little use of computational time,
- the execution time is not critical,
- very heavy I/O, in particular between main memory and disc memory,
- the size of main memory is not critical, but higher application performance can be achieved by the use of larger memory,
- the size of secondary memory, mainly disc memory, is usually essential.

The nonnumerical applications have as their primary objective the management of data for information processing systems. Considered in terms of the structure of the data and of the types of processing supported, information processing systems represent a very large class. But each actual implementation of these systems represents a compromise between an ideal system that accomplishes all the users' requirements and a possible real finished product that can be implemented within the limited capabilities of present computer systems. In consequence there is a wide range of systems, each of which specializes in dealing with a group of specific real-world applications, because each group of applications has some intrinsic characteristics.

Each type of system implements the users' principal requirements in an efficient way and the other requirements less efficiently. There are many users' requirements that have to be taken into consideration during the design phase of a system, but two types of requirements play major roles, namely the acceptable respone time to users' queries and the nature and quantity of data that has to be managed.

Response time

The achievement of a rapid response time is incompatible in present systems with the management of large volumes of data. Therefore, although it is possible to implement fast systems, they have to manage very well structured and 'objective' data, or if the response time is not critical, then less structured and 'descriptive' data can be managed.

It seems appropriate to introduce an example for each of these types of system to clarify the distinctions.

An example of the first type of system is an automated inventory system. The system manages information about the goods stored in a warehouse. The data held for each type of stock item can include: a unique inventory number, date of arrival of the goods in the warehouse, dimensions, weight and a short description of the item. The users can query the system about individual stock items or on statistics derived from the stored data. The users utilize the answers obtained from the system to assist in making management decisions about the operation of the business.

Nature and quantity of data

An example of the second type of system is the automated information retrieval (IR) system that manages the description of documents in a library. Here the objects being processed by the application are the documents. In present day IR systems only descriptions of the documents are stored, essentially because of disc storage space. There are only a few exceptions to this, where the full document is stored together with its description, but they are usually for specific applications such as legal IR systems. In general, for each document the data stored could be: author(s), title, type of publication, date of publication and keywords or index terms to identify the content of each document. The users can query the system using these items. The users do not usually use the answers to take decisions or actions, but use the retrieved information to locate the original documents in the library. The final targets of the users, therefore, are not the descriptions of the documents held by the system, but the original documents whose existence and location are revealed by the system.

Between the extremes typified by these two examples, it is possible to distinguish many types of system that implement different application types.

The inventory control example, which processes details of goods stored in a warehouse, has shown that this type of system manages the relevant attributes of the entities of the application; but for this type of application the descriptions are only of external characteristics of the subjects. This means that, in general, attribute values are not the result of subjective judgement when the data is collected. Because of their deterministic nature these values can be called *data.*

At the other end of the range, the second example shows that there are systems that manage the descriptions of the information content of the application entities. For example, to describe the contents of a document is not a deterministic process, because either the contents can be seen and described in different ways by different people or because the descriptions have to reflect different users' requirements. These descriptions can be identified by the term *paradata.* The prefix 'para' is used to denote that the stored descriptions are subsidiary to the complete information on the entity that the user of the system is looking for. The complete information can be stored internally in the system (as in office automation systems) or it can be held externally to it (as for many IR systems), but in both kinds of systems the users utilize the answers to their queries as references to the final useful information.

If a system stores the full contents of every document, it is not sufficient for the user to have a system that only manages the contents. The system must also be able to manage the descriptions together with the contents. This type of system therefore has to support users with two

levels of access: at the first level the users need to 'browse' through the descriptions, because it is not feasible to access the complete database without some sort of indexing. This means that the users need to see the descriptions through a representational method of the application. At the second level they need to browse through a set of potentially relevant documents that have been identified by the system at the first level.

The distinction introduced between systems that manage data and systems that manage paradata is very important, because it means that the first type of system serves to inform the users directly through the managed data about the subject of the application; the second type of system, however, informs the users of the existence of items that could be relevant to them.

All the systems, through the whole range, have certain characteristics in common with the DBMS that traditionally manage highly structured data, but the different types of systems present different capabilities that are necessary for the management of systems containing data with different degrees of structuring.

A diagrammatic representation of the spectrum of information processing systems is proposed in Figure 1. The systems on the left of the scale can be characterized by restricted input selection criteria, fixed formal records and the importance of fast access to the data to give rapid answers. In contrast, the systems on the right are not response-time critical, but their importance lies in retrieving all the potentially pertinent documents to a users query. They need also to be flexible, as they manage variable format records of essentially descriptive information.

In Figure 1, DBMS-1ndicates database management systems with two levels of data representation; external and internal levels. DBMS-2 implies DBMS with three levels of data representation; external, conceptual and internal. DBMS+D/DS are the DBMS that have been developed by means of a data dictionary system (D/DS) which users can consult. D/DS indicates the data dictionary/directory systems that are used in the enterprise independently from a DBMS, for documentation purposes. DSS stands for decision support systems, OAS is the abbreviation of office automation systems and videotex indicates all the different viewdata systems (e.g. some of the systems it is possible to use in the UK are Prestel, Ceefax, Oracle). Online stands for the online bibliographic databases, namely the commercial systems that manage bibliographic references (e.g. Dialog and Orbit[1]), and IR stands for automatic information retrieval systems.

Much work has been done on the realization of effective applications through the systems depicted on the left of the dotted line in Figure 1. The special-purpose hardware that has been designed and developed so far is essentially for an introduction to computer systems that have to support applications of these systems, as discussed below.

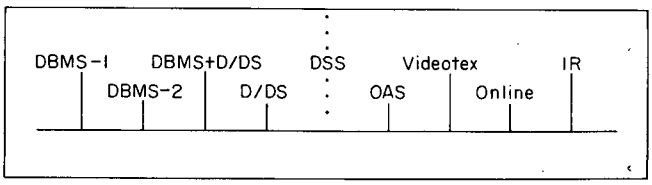

Figure 1. Spectrum of information processing systems

The remainder of the paper concentrates on the characteristics of systems to the right of the dotted line in Figure 1; identified throughout the paper by the abbreviation LSDS. Many applications of these systems have been implemented, but it is still very difficult to build effective applications, because of the limitations of the present computer equipment. In fact, it is believed that the addition to the present computer systems of hardware specifically built for these kinds of applications or the construction of architecturally new computer systems, are the routes to follow to make effective applications of the OAS, videotex, online and IR types possible.

MANAGEMENT OF LSDS

The main difference between information processing systems that manage a prevalent part of paradata, or LSDS, and systems that handle a predominant part of data, is that the LSDS have to utilize a more complex method of representation of the reality relevant to the application.

All systems depicted in Figure 1 have to use a representational method of reality of the automated application (later in the paper the short form 'representational method' will be used), but the method used needs to be more and more sophisticated ranging from left to right across the systems. In fact, a system that acts as a manager of the contents of stored documents (a document can be a letter, a report, an article, a book etc., depending on the type of LSDS), has to be able to communicate explicitly the method of document content representation used to the users of the system. The IR systems are at the end of the spectrum of these systems because they would need, to be really efficient, a representational method of the level of knowledge that is understood in the discipline or disciplines pertinent to the automated application.

The theoretical problems of knowledge representation are, as yet, unresolved issues, but one can hope for some advancements in the near future, since research is going on in very different areas, from artificial intelligence to sociology (for example, Negrotti[2] presents an innovative sociological approach).

A consequence of this fact is that all the present information processing systems have to use imperfect and empirical knowledge representation methods. These methods, however, have different levels of sophistication that can be evaluated by two significant factors:

- the representational method that the system utilizes has to be like a guide for the users of the system, and has to lead to pertinent information without preventing the users from 'browsing' into the system with unforeseen research paths,
- the representational method has to have internal capabilities to permit or make easy the dynamic redefinition and the update of the entities' descriptions.

For instance, some of the knowledge representation methods that are employed in IR-type systems are subject indexes, thesauri and semantic networks. Only a few experimental systems with an implementation of both the factors described above have been developed so far. One example is a study of the utilization of semantic networks as a representational method in an automatic IR system for a documentary application of the DOC system[3–5].

The main problem in the utilization of sophisticated representation methods is that the better representation is at present completely implemented by means of software, because the capabilities of current computer systems are not designed for the necessary operations; a short account of the capabilities of conventional computer systems is given below.

Application of 90/10 Rule to LSDS

The most common operations that a conventional information processing system has to execute on the records that store the application data are:

- the records potentially pertinent to the user query are transferred from secondary storage to main memory buffer in order to search through them in a sequential manner, thus keeping only those most likely to be relevant; the sequentiality of these operations is enforced by current computer systems architecture, because these operations would logically be performed in parallel or in an associative way,
- the user query is solved and the subset of pertinent records is extracted from the set constructed in the above operation; the user query is usually a boolean and/or an arithmetic expression of predicates, but the resolution of a query of an LSDS is more complicated than the resolution of queries of the other systems, because the query predicates are not restricted to the exact match in fixed fields and are often used for the cooccurrence of predicates within a particular context or a specified number of words[6].

The percentage of the total records stored by a database management system that is necessary to read and manage in the two operations defined above has been estimated for very large databases. Hsiao[7] introduced the 90/10 rule for the execution of both the operations in current database management systems. This rule says that only 10% of all the records transferred from secondary storage to main storage are kept for the second phase, and again, in the second phase, only 10% of all the previously kept records is really pertinent to the user query. The mean percentage of the really pertinent records in each of the two phases is thus 10% for very large databases.

This evaluation cannot be directly applied to LSDS, because the first operative phase can be realized in a completely different manner in the LSDS, so that its direct extension could be wrong. A corresponding evaluation for some of the LSDS has not yet been done, but the main difference is the use of a more sophisticated representation method in the LSDS, as shown below.

The representation method acts in the first operative phase of LSDS as a filter to identify the potentially relevant records. This filter can be implemented in many different ways, but its effect on the architecture of an operational system is sometimes underestimated when studying the introduction of special-purpose hardware in the architecture of future computer systems to be used in managing information processing systems. The following example of a class of automatic IR systems can help in clarifying this difference.

The example is based on the IR and online systems that are implemented through a system which manages the data in an inverted file organization by means of the systems that are based on the inversion of all the record attributes the users can use in the formulation of their queries; in these systems the inversion is always implemented for the keyword or index term attribute. The system then creates and updates a directory for each attribute on which the inversion is made; the directory containing all the lists of pointers referring to the records containing a specific occurrence of the attribute. The storage space necessary to keep all the directories so constructed is equal to four times the storage space necessary to maintain the data records themselves in systems of a comparable size with databases investigated by Hsiao. This means that in the first phase of such a system, there are extracted subsets of these directories and not of the data records, and the operations of the query resolution of the second phase are made from these lists. Each operation of list transfer from mass storage to main memory does not imply a transfer of 100% useful data, because of the I/O buffers' dimensions and the lists' records organization, but the major part is pertinent. The unnecessary data has not been experimentally calculated in existing systems, but it can be estimated to be around 10% of all the transferred data, and in this case the 90/10 rule is inverted.

At the beginning of this section it was shown that the query to the LSDS can be more complicated than in other systems. Consequently, the 90/10 rule remains applicable for the LSDS only if the query is of the same complexity as in other systems. If the query is more complicated, it can be necessary to iterate the first phase more than once. If this does happen, comparability with other systems no longer exists, and this case cannot be considered. When the user query has been resolved, and the list of pointers to pertinent data records has been created, the data records are transferred from secondary memory to main memory, in order to make them available to the user. This is a third phase that does not exist for the database management systems, because at this stage the set of pertinent records has been created in DBMS.

In conclusion, the LSDS that are implemented with an inverted file organization need to have access to less records during the first and the second phases, but it is necessary to add a third phase to complete the answer for the user.

The previous example has shown that, for a specific implementation class of LSDS, the 90/10 rule may not be applicable, depending on the data these systems have to

manage. Consequently, when an LSDS has to be designed, it is very important to study its peculiar functional characteristics in advance. It is undesirable to transfer directly the results of an implementation study done for other systems, particularly if an attempt is made to add to the study only an analysis of the functional characteristics of text management. Some other peculiar functional characteristics of the LSDS are illustrated below.

Evaluation criteria for LSDS

A characteristic feature of LSDS is that if a logical/semantic connection has not been implemented during the insertion phase of the data into the system, this data can not be retrieved in that semantic context. This means that the evaluation criteria are different for LSDS. In fact, for the evaluation of the efficiency of one of these systems, it is necessary to use the 'classical' contingency table shown in Table 1, and the measures of effectiveness that can be derived from it.

If the LSDS stores the whole documents, then N in Table 1 is the number of stored documents; if the system stores a description of each document, N indicates the number of the document descriptions kept by the system. The same distinction between document and document description applies to sets A and B; later on in the paper it is mentioned as the document for both the cases, whole document or description of it. A is the set of relevant documents to the user's query that the system is storing. $\overline{A}$ is the set of nonrelevant documents that are stored in the system. B is the set of documents that the system is able to retrieve to answer the user's query. $\overline{B}$ is the set of non-retrieved documents by the system, both relevant and nonrelevant ones.

Only the precision and recall effectiveness measures that can be derived from the contingency table are reported here, because an in-depth treatment of the other measurements is beyond the scope of this paper. A good study of evaluation of IR systems can be found in Reference 8.

The precision measure is defined as the ratio of the number of relevant documents retrieved to the total number of documents retrieved by the system to answer the user's query, i.e.

$$\text{Precision} = \frac{|A \cap B|}{|B|}$$

Table 1. Contingency table, so-called even though it has little in common with the homonymic statistical tables

	Relevant	Nonrelevant	
Retrieved	$A \cap B$	$\overline{A} \cap B$	B
Not retrieved	$A \cap \overline{B}$	$\overline{A} \cap \overline{B}$	$\overline{B}$
	A	$\overline{A}$	N

The recall measure is defined as the ratio of the number of relevant documents retrieved to the total number of relevant documents that have or have not been retrieved by the system (the term 'relevant documents' means all the documents existing in the system) to answer the user's query, i.e.

$$\text{Recall} = \frac{|A \cap B|}{|A|}$$

Some conclusions can be now drawn; special-purpose hardware that has to be used as part of a computer system devoted to the running of LSDS has to be designed to fulfill the operations that in many current systems are processed by a type of filter lying between the data records and the users, in the form of software. The new LSDS that use special-purpose hardware have to improve upon the present measures of effectiveness, especially in providing better precision and recall measures.

Functionalities and functional requirements of LSDS

This section deals with the functionalities and functional requirements with which an LSDS must comply under contemporary design. It is evident that some functionalities and requirements are dependent upon the specific application for which the LSDS is used. It is not the aim of this section to deal with all the different sets of requirements that every type of LSDS must comply, but its scope is to identify the subset of requirements that each LSDS needs to have independently from the specific application for which the system is used. This is a useful subset of requirements to bear in mind when the introduction of special-purpose hardware is envisaged in a computer system to be used for LSDS. Where it is possible, the checklist form is used for the exposition.

An illustration is developed below (Figure 2) by analysing a system into three main functional components:

- the input/query component which can be broken down into the two subcomponents of documents description/data entry and users' queries,
- the processing/storage component, which uses the same representation method for processing the data as that used in the input/query component but also stores or retrieves the documents according to whether it has triggered an update or a request,
- the output component which answers to the user's queries; the system answer can request a feedback in both the input/query subcomponents and in between them. In fact, the answer can request a revision of document descriptions (this revision can be due also to changes in the user's requirements), since it is difficult to use the present document descriptions properly in the formulation of the query or in the understanding of the answer. Alternatively, it can request a modification (extension or reduction) of the initial query because the user now knows more about the subject through the received answer.

The feedback that has been identified in Figure 2 with the input/query component operates in all current LSDS, but the two remaining feedbacks introduced here have not yet been implemented.

Input/query component

This component can be broken down into two subcomponents; data entry and user query processor. This is a crucial component, not only because it is the main user interface with the system, but also because the choices which are made at this level affect the operation of the overall system. As has been shown above, the knowledge representation method is fundamental to the functioning of the system, and the decision on the representation method has to be taken at this level. It is dependent directly upon this decision whether the whole document is to be searchable or nonsearchable, if only keyword and/or index terms are searchable or if a simultaneous searching of the whole document and of its description is to be allowed. If a description of each document is entered into the system, it is important to note that the description is a transcription of the conceptual analysis in a controlled language.

The conceptual analysis of the document is carried out bearing in mind the informative users requirements, and the main target of each description is then to foresee the keywords or terms the users are going to use to retrieve the documents. Each description, however, is prepared at a specific time and the users requirements are changing all the time. It is therefore necessary to introduce into the component two feedback mechanisms, as in Figure 2; between the output component and the document description part of the input/query component, and feedback between the query subcomponent and the document description. In

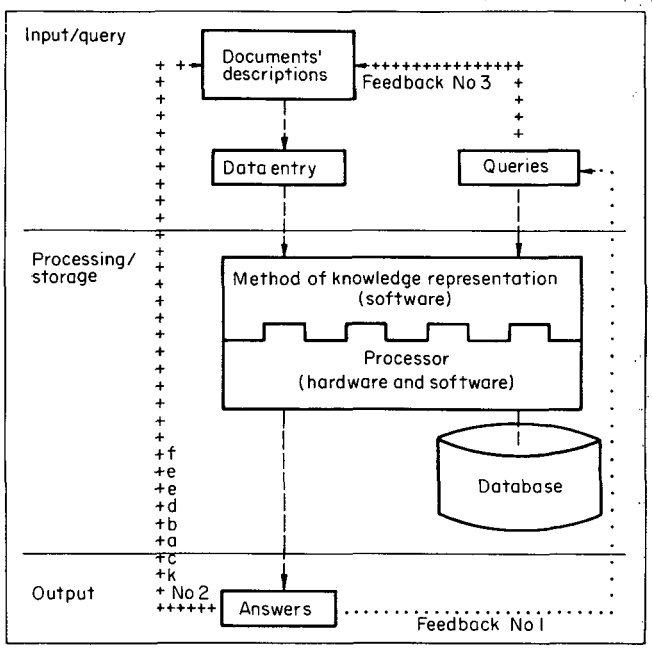

Figure 2. Main functional components of an information processing system

current systems, neither of these two feedback mechanisms are usually implemented. In fact, both of them imply the implementation of an automatic mechanism to trigger the manual redefinition of the document description, and this second action can be much more difficult to implement.

Another decision that has to be taken at this level is whether the system has to be able to communicate with other systems. This could involve a node of a distributed system or sophisticated interfaces to be used by users who are cognisant with different systems but not with the language and conventions of this specific system. If the latter is the case, a specific representation method can be chosen to facilitate vocabulary compatibility, while specific research paths can be chosen and implemented to make the query formulation easier for different users.

Finally, the data that has to be entered has some common characteristics; most data is textual in completely or partially free format, and has to be retrieved through either exact or partial match, in which case it is necessary to define the roots of terms of variable lengths.

Processing/storage component

The main characteristics of the datafiles include the following.

- The file sizes are larger than those of most other applications, and they accumulate information because the data is seldom deleted from the system. This is certainly true for the IR and online systems, and is true for a limited subset of the data stored by videotex systems and for a substantial part of the data managed by DSS and OAS.
- The most common type of update has one characteristic that is directly related to that above; the usual update operations are the insertions of new data records, followed by the modifications of previously stored records because of input errors and, finally, delete operations, which rarely occur.
- The number of record types is usually very small.
- Most of the fields are of variable length and it can be difficult to foresee the maximum length of most of them.
- The records can contain one or more keys, but the key concept is not typical of this data, being enforced for operational reasons.
- The users cannot insert or modify the data records in most of the LSDS, but they can extract (not in online type systems) a data subset from the system, modify and manipulate it in their work area and store it in their disc area.
- For most of these systems, problems of security can arise if personal details of users are entered on files.

Output component

The need for a feedback mechanism between this component and the query subcomponent has been referred to above.

An important feature of this output component is that it

has to be able to support the user properly in the query formulation. In understanding the answer (some systems are very obscure) and in the possible query reformulation, the user has to be aware of all the system's capabilities, beyond which different interactive levels of help have to be set up to facilitate effective communication with different categories of users.

The output component has to provide different output media, in particular if the system is cooperating with other systems (so that it can communicate by cable, satellite etc.), but also because it can be useful to produce the output on a medium that is easy to send, e.g. microfiche or computer output microfiche (COM).

ARE CONVENTIONAL COMPUTER SYSTEMS SUITABLE FOR INFORMATION PROCESSING?

In the late '60s and early '70s two main factors brought about the question of whether or not conventional computer systems are suitable for information processing:

- the cost of hardware showing a steadily decreasing trend for the future,
- the increase in the use of computer systems for non-numerical applications and, in particular, through database management systems (DBMS-1 and DBMS-2) and information retrieval systems (IR and online).

The first factor has made it possible to contemplate wider uses of hardware components than in the past. The second factor has revealed the limits of the architecture of traditional computer systems. The main characteristics of conventional computer systems are summarised below to clarify the unusual characteristics of the new special-purpose hardware devices that are introduced here with a functional classification.

Main characteristics of conventional computer systems

Conventional computer systems have been built mainly to perform scientific calculations, rather than management and storage of information and data. In fact, the architecture of computer systems has remained essentially the same for many years. Computers have also been classified into four different generations, because of the diverse qualities of the technology involved in their construction. This common computer architecture is the so-called von Neumann architecture, von Neumann being the first to present a complete analysis of computer architecture from a logical point of view[9].

The von Neumann computer is basically a machine built to perform fast and reliable computations, and the architecture has been designed to optimize this basic function. Its main characteristics are:

- a unique component with logical control power over all the operations to be performed in a rigorous sequential manner,
- a single component computing, storing the data and communicating with the outside world,
- main memory cells of fixed size that are organized in a linear way,
- the cells are addressed in a linear manner,
- the machine language is a low-level language that performs simple operations on elementary operands.

Functional classification of special-purpose hardware

Since the early 1970s, a number of special-purpose hardware devices have been designed, and some of these have been built experimentally. The common design aim of these devices is to incorporate, in the hardware, some of the functions most frequently used for the management of the information processing systems. These are implemented through software within conventional computer systems (the most important ones have been illustrated above). It is not an aim of this paper to present the technological characteristics of these devices, these being already well documented; Reference 10 can be used to refer to the pertinent literature, while some introductory and survey references are References 7 and 11–15.

The aim here is to group the devices, depending on their main functions and on the possibility of their introduction into a conventional computer system, to improve the computer capabilities in supporting information processing systems.

The devices are grouped in three classes (*C*1, *C*2 and *C*3) because of their common characteristics and because of their variance from the von Neumann architecture.

C1 — 'internal' devices

These are devices that have been designed to replace a part of a conventional computer system. The replacement would essentially change the performance of the computer system by a reduction in the execution time for some operations, rather than a radical change in the overall architecture. For this reason, these devices are called internal devices; they work inside the conventional computer system architecture. Bubbly memory and associative memory are significant examples of these devices. Some of them can be effectively used as components of the machines of class *C*2.

C2 — 'external' devices

These are devices that substantially modify the conventional computer architecture, and for this reason are called external. In fact, they are external to the conventional architecture and they interact with it to create a new architecture. These are special-purpose machines, because they are designed to perform operations that are not typical of the traditional architecture. They are special purpose in data management operations such as update, insert, delete and comparison of data records. Examples of these devices are the database machine and the database computer.

C3 — 'new' generation computer systems

The typical devices to be included in this class are the new generation (sometimes called fifth generation) computer systems that are the final target of the national project launched in Japan to produce machines for the applications of the 1990s. These computer systems have a completely different architecture from von Neumann machines and they are introduced briefly below. It is also possible to include in this class machines that are being built elsewhere (some machines have been designed and built experimentally in Europe and in the USA) to achieve a substantially different architecture from the traditional architecture; some examples are given below.

Components of LSDS in which special-purpose hardware devices have been introduced

Many devices of the *C*2 class that have been developed so far have been designed essentially for data management and not to cope with the functionalities of LSDS.

Before analysing some of these devices, the classes of special-purpose hardware, introduced above, are related in this section to the functional components of LSDS that have been illustrated in Figure 2, specifying those that can be supported by the introduction of special-purpose hardware. An exhaustive analysis of the introduction of special-purpose parallel hardware for the effectiveness of IR-type systems can be found in Reference 15.

Most of the present devices of classes *C*1 and *C*2 have been designed to support some of the functions of the processing/storage component. Few of them have been designed for the input/query component or for the output component.

The functionalities illustrated in the section on the input/query component which can be improved by the introduction of special-purpose hardware devices are those related to the data and text entry process for the input subcomponent, and those of exact and partial match of index terms and keywords for the queries subcomponent. The machine presented in Reference 16 demonstrates the improvements that can be achieved in the query subcomponent with the introduction of a special hardware device; in fact one of its characteristics is to allow the users to ask imprecise questions, that is with spelling mistakes or with words missing from sentences. Some aspects of the input/query component are analysed in References 17 and 18 on the design of the devices of classes *C*1 and *C*2. In Reference 19, new hardware of class *C*1 to support both input and output functions is presented.

As referred to above, many devices have been developed for the improvement of some of the functions that are performed by the parts of the processing/storage component. Among the internal devices there are the associative memories[20, 21] that can be used to speed file searching, and many devices have been developed to improve secondary storage performances[22–28]. Some different techniques/devices have been introduced for the implementation of efficient sorting and merging mechanisms[29–34]. Many external devices have so far been developed for the realization of the functionalities of the processing/storage component, but they are generally database management oriented and specifically for relational databases.

ADVANCES IN SPECIAL-PURPOSE HARDWARE FOR MANAGEMENT OF LSDS

It is clear from the discussion above that the conventional computer system architecture is not a suitable architecture for the management of LSDS. In fact, it is not directly suitable for the management of the content of main-memory subsets of a variable number of cells which correspond to data records of variable dimensions transferred from mass storage, especially if it is necessary to make several comparisons between data records which would be processed efficiently in parallel instead of in sequence, and whose final purpose is to extract only those records that meet a predicate.

In the next sections some special-purpose hardware now being developed and built is taken into consideration, to determine whether it is suitable for insertion in a computer system used essentially to manage LSDS.

Database machines

As is implicit in their name, the database machines that have been constructed so far have been designed to execute efficiently some database operations. This means that each of these machines operates in a symbiotic way with a conventional computer system in which a database management system coordinates the management of data and communicates with the users. Database machines can be inserted in a computer system to manage LSDS, because they can be used to manage some of the operations of the processing/storage component. In this case, however, the software of the conventional computer system has to support all the peculiarities of the information that the LSDS manages.

All database machines can be inserted in class *C*2, but they can be further divided into classes of database machine architectures, as in References 35 and 36, where their general performance is comprehensively evaluated for the first time. In fact, the other evaluations that have been presented earlier are for a specific machine only; in Reference 37 the RAP database machine performance[38], and in Reference 39 the DataBase Computer (DBC)[40].

Survey and performance evaluation references to most database machines are found in References 7 and 36. Some references to other database machines are found in References 41–44 (for the Sabre database computer) and Reference 45 (for the Pritton-Lee Intelligent Database Machine (IDM)).

Content-addressable file store (CAFS)

The content-addressable file store (CAFS) has been

designed and constructed by International Computers Limited (ICL). It is well documented in the literature, some of the most useful being References 46–52. The CAFS can be included in the *C*2 class of the functional classification proposed above.

CAFS is available on the market in its third generation. It is one of the few devices of its kind that is available for purchase, because nearly all of the special-purpose hardware devices that can be recognised as *C*2 or *C*3 devices are prototypes and have been developed more as research vehicles.

CAFS is a special-purpose computer that is able to process a limited number of data management operations at hardware level; it must always be connected and cooperate with a traditional computer system. This is because CAFS operates under the logical control of a database management system that is resident in a conventional computer system. This means that CAFS has been developed to be inserted in systems of DBMS-1, DBMS-2 and DBMS+D/DS types only; it has not been developed to be inserted as a component of a computer system to manage LSDS. The operations performed by this hardware are some of the most common necessities in the management of data, and these operations are implemented through software in normal database management systems. CAFS removes the first application of the 90/10 rule for the systems depicted on the left of the dotted line in Figure 1. The insertion of a device of this kind in a conventional computer system has been shown to be advantageous when very large applications have to be managed.

It is important to note that some of the features of CAFS seem to assist in the implementation of LSDS; for example, the capability for making comparisons, at hardware level, on the roots of terms (an important functional characteristic that was discussed above), or an indexing method implemented in CAFS at hardware level, effective for very large applications, that produces the results of an inverted file organization without the disadvantages of the generation and maintenance of the list directories. The insertion of CAFS in a computer system to be used for the LSDS type could be a very interesting feature, as discussed above.

Intelligent file store (IFS)

The intelligent file store is a database machine under development at the Computer Science Department of the University of Manchester, UK[53]. The designers of IFS have considered the structure of the information to be managed with machine support to be a central architectural problem. This approach is very modern, and is a hopeful sign for future complete implementations of LSDS. It is a characteristic that differentiates the IFS architecture from the database machine prototype that has been completed recently.

The IFS architecture is based on a specific data model; the hardware has been designed to manage the data represented by the binary relational model (BRM) directly. The BRM was chosen because the relationships between the data it services can be directly implemented in the hardware.

It is necessary to examine software in order to view the management of the data in other data models, and it is possible to foresee that this should not be too onerous. A database management system could be used in a computer system incorporating a database machine of IFS type, where the data would be processed by BRM, but the system must have mapping and management programs for different data models so that the user could use the data model most suitable for the application data. In this way the user would not be confined to the use of a single data model, as in most current database management systems.

IFS has not been designed to be drastically different from present database machines, but it presents some innovative characteristics. IFS can be classified as a *C*2 device, but it has to be considered a second generation database machine, because its architecture is closer to a new generation computer than the previous database machines.

IFS has been designed for insertion into an architecture to support applications by means of DBMS-1, DBMS-2 and DBMS+D/DS types, but it is also believed that IFS could be inserted in a computer system to be used for the management of LSDS, because of its flexibility in the representation of the structure of information.

Fact system

The fact system deviates from the architecture of conventional computer systems and cannot be identified with class *C*2 of the proposed functional classification. In fact, the architectural design of this system consists of two parts, a new data structure and a computer system designed to manage it.

The new data structure is based on the 'fact' (from which the name of the system derives) as a basic unit; a fact being the smallest independent item of information. A fact is a quadruple of:

- a unique identifier of the quadruple, the fact number,
- a subject,
- a relationship,
- an object.

Each fact must be unique and represents a simple relationship. It is possible to represent every level of complexity of information with different combinations of several facts. This is the basic concept on which the fact system and its architecture is founded. It can easily be seen that this system is far removed from conventional architecture; in fact, the design of this machine has been studied in an attempt to overcome the problem of information representation, and it could be used for the implementation of a knowledge representation method. This system can then be considered a possible candidate for support of the implementation of LSDS, and can be considered to be in the *C*3 class of the proposed classification.

The fact system was designed in the Department of

Computer Science of the University of Strathclyde[54]. A prototype is not yet available, but the design project includes different hardware levels to perform functions of varying degrees of complexity; from level 0, that has to manage a binary relational system, to the complexity of level 3, which can manage different levels of quadruples. The fact system is programmed to deal with problems at the highest level of quadruples.

New generation computer systems

The term 'new generation' computer systems has been introduced recently to replace the previous term 'fifth generation', in order to identify and clearly underline the main target of a project launched in Japan for the construction of architecturally new computer systems. Some of the devices that are being developed in the USA and in Europe have similarities with the machines envisaged by the Japanese project; the fact system and IFS are, in some respects, two examples of these machines. This Japanese project is unique, however, and in fact is the only national project for the construction of computer systems suitable for future applications such as the LSDS, defined above.

At present, the project can be examined in a limited manner only, because it is still in the design phase, and the available documents[55, 56] show only the main targets of the project and the main research routes the Japanese are prepared to follow. Pertinent to the discussion in this paper are References 57–63.

The first of such computer systems are expected to be ready for the early 1990s. The final targets of the project are very ambitious; in fact, the computer systems are oriented to incorporate a knowledge representation method and therefore to manage all the different types of LSDS. The main subcomponents that are under development to build these systems are:

- the inference and problem-solving component,
- knowledge base machine,
- various intelligent interfaces.

CONCLUSIONS

The paper has illustrated some of the functions that have to be incorporated in the special-purpose hardware of the future for the implementation of effective LSDS, where an LSDS is a system that has to manage a higher percentage of paradata (descriptions of the information content of the application entities) and a lower percentage of data (descriptions of external characteristics of the application entities). The study concentrated on the functionalities and characteristics of LSDS. The still unresolved problems for the implementation of effective LSDS have also been underlined. There is the necessity for more suitable hardware to deal with nonnumerical problems, as has been shown, but some underestimated problems relating to the nature of LSDS have also been illustrated and need to be resolved.

A functional classification of current special-purpose hardware devices has been introduced and used to identify some of the machines that are under development. It has been shown that some of these machines permit researchers to foresee the implementation of some effective LSDS in the 1990s.

ACKNOWLEDGEMENTS

Sincere thanks are due to G Fitzgerald and R Johnson of the School of Mathematics, Statistics and Computing of Thames Polytechnic, UK, for their very useful discussions of this paper and for their general comments, along with those of D Cantwell. The bibliographic research necessary for this paper would not have been possible without the help of the librarians of the Main Library of the Thames Polytechnic to whom sincere thanks are also due. The research on which this article is based was sponsored under a NATO/Italian Research Council grant.

REFERENCES

1 **Hall, J L and Brown, M** *Online bibliographic databases: an international directory* Aslib, UK (1981)
2 **Negrotti, M** 'Cervello, mente, intelletto: i tre volti dell'informatica' *Proc. idi 83 Conf.,* Italy (May 83)
3 **Agosti, M, King, M, Lestuzzi, F and Wettler, M** 'Ulteriori sviluppi dei due sistemi di documentazione automatica SCRIN e DOC-4' *Proc. AICA 77 Conf.* Pisa, Italy (1977) pp 127–128
4 **Agosti, M, Crescenti, M E, Lestuzzi, F and Schiavon, P** 'Sistemi informativi: DOC-4 e SCRIN' *I Quaderni dell'Elaborazione Automatica* Vol 2 (1978) pp 49–72
5 **Agosti, M and Ronchi, M E** 'DOC-5: the bibliographic information retrieval system in CINECA library automation project' *Proc. ecodu-29 Conf.* Berlin, Germany (April 1980) pp 20–31
6 **Hollaar, L A** 'Hardware systems for text information retrieval' *Proc. 6th Ann. Int. ACM SIGIR Conf. Res. Development in Inform. Retrieval* USA (June 1983) pp 3–9
7 **Hsiao, D K** 'Data base computers' in **Yovits, M C (Ed.)** *Advances in computers* Vol 19, Academic, USA (1980) pp 1–64
8 **van Rijsbergen, C J** *Information retrieval* (2nd Ed.) Butterworths, UK (1979)
9 **von Neumann, J** *First Draft of a Report on the EDVAC* USA (June 1945)
10 **Agosti, M** *Specialised hardware for database management and information retrieval: a classified bibliography* Thames Polytechnic, UK (July 1983) p 28
11 **Bray, O H and Freeman, H A** *DB computers* Lexington, USA (1979) p 180
12 'DBMAC1' *Rapporto Tecnico 1979 DATANET/DBMAC1* CNR, Progetto Finalizzato Informatica, (1979) p 158

13 **Hsiao, D K** 'Guest editor's introduction: database machines are coming, database machines are coming!' *Computer* Vol 12 No 3 (March 1979) pp 7–9

14 **Langdon Jr, G G** 'Database machines: an introduction' *IEEE Trans. Comput.* Vol C-28 No 6 (June 1979) pp 381–383

15 **Salton, G** 'Automatic information retrieval' *Computer* Vol 13 No 9 (September 1980) pp 41–56

16 **El Masri, A, Rohmer, J and Tusera, D** 'A machine for information retrieval' *Proc. 4th Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1978) pp 117–120

17 **Haskin, R L and Hollaar, L A** 'Operational characteristics of a hardware-based pattern matcher' *ACM Trans. DB Syst.* Vol 8 No 1 (March 1983) pp 15–40

18 **Langdon Jr, G G** 'A note on associative processors for data management' *ACM Trans. DB Syst.* Vol 3 No 2 (June 1978) pp 148–158

19 **McDonnell, K J** 'Trends in nonsoftware support for input–output functions' *Proc. 3rd Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1977) pp 40–47

20 **Bandurski, A E** 'Associative memories: a trend in data base technology' in *DB management* No 24-01-12 Auerbach, USA (1976) p 16

21 **Kibler, T R** 'APCAM — a practical cellular associative memory' *Proc. 5th Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1980) pp 82–83

22 **Bird, R M, Tu, J C and Worthy, R M** 'Associative/parallel processors for searching very large textual databases' *Proc. 3rd Workshop Computer Architecture for Non-Numeric Processing* ACM, USA (1977) pp 8–16

23 **Burkowski, F J** 'A microprogrammed search controller for a text scanning processor' *Proc. 5th Workshop Computer Architecture for Non-Numeric Processing* ACM, USA (1980) pp 39–48

24 **Hoagland, A S** 'Storage technology: capabilities and limitations' *Computer* Vol 12 No 5 (May 1979) pp 12–18

25 **Lin, C S, Smith, D C P and Smith, J M** 'The design of a rotating associative memory for relational database applications' *ACM Trans. DB Syst.* Vol 1 No 1 (March 1976) pp 53–65

26 **Lipovski, G J** 'On imaginary fields, token transfers and floating codes in intelligent secondary memories' *Proc. 3rd Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1977) pp 17–22

27 **Lipovski, G J** 'Semantic paging on intelligent discs' *Proc. 4th Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1978) pp 30–34

28 **Slotnick, D L** 'Logic per track devices' in **Alt, F L and Rubinoff, M (Eds.)** *Advances in Computers* Vol 10 Academic, USA (1970) pp 291–296

29 **Chen, T C, Lum, V Y and Tung, C** 'The rebound sorter: an efficient sort engine for large files' *Proc. 4th Int. Conf. on VLDB,* IEEE, FRG (September 1978) pp 312–318

30 **Wolf, G** 'Associative mass storage for databases' *Proc. 5th Workshop on Computer Architecture for Non-Numeric Processing* ACM, USA (1980) pp 70–81

31 **Chung, K M, Luccio, F and Wong, C K** 'Magnetic bubble memory structures for efficient sorting and searching' in *Inform. Processing 80* **Lavington, S H (Ed.)** North-Holland (1980) pp 439–444

32 **Fialkowski, K and Muraszkiewicz, M** 'Matrix for retrieval and sorting' *Inform. Syst.* Vol 5 (1980) pp 219–224

33 **Hollaar, L A** 'Specialised merge processor networks for combining sorted lists' *ACM Trans. DB Syst.* Vol 3 No 3 (September 1978) pp 272–284

34 **Muraszkiewicz, M** 'Concepts of sorting and projection in a cellular array' *Proc. 7th Int. Conf. on VLDB* IEEE, France (September 1981) pp 76–80

35 **Todd, S** 'Algorithm and hardware for a merge sort using multiple processors' *IBM J. Res. Development* Vol 22 No 5 (September 1978) pp 509–517

36 **Dewitt, D J and Hawthorn, P B** 'A performance evaluation of database machine architectures' *Proc. 7th Int. Conf. on VLDB* France (September 1981) pp 199–213

37 **DeWitt, D J and Hawthorn, P B** 'A performance evaluation of database machine architectures' *J. Digital Syst.* Vol 6 No 2/3 (Summer/Fall 82) pp 225–250

38 **Ozkarahan, E A, Schuster, S A and Sevcik, K C** 'Performance evaluation of a relational associative processor' *ACM Trans. DB Syst.* Vol 2 No 2 (June 1977) pp 175–195

39 **Ozkarahan, E A, Schuster, S A and Smith, K C** 'RAP — associative processor for database management' *Proc. AFIPS Conf.* Vol 44 (1975) pp 379–388

40 **Banerjee, J and Hsiao, D K** 'Performance study of a database machine in supporting relational databases' *Proc. 4th Int. Conf. VLDB* IEEE, FRG (September 1978) pp 319–329

41 **Banjeree, J, Baum, R I and Hsiao, D K** 'Concepts and capabilities of a database computer' *ACM Trans. DB Syst.* Vol 3 No 4 (December 1978) pp 347–384

42 **Gardarin, G** 'An introduction to SABRE, a multi processor data base computer' *Int. Congress on Applied Syst. Res. and Cybernetics* USA (December 1980) p 10

43 **Gardarin, G and Valduriez, P** 'La machine bases de donnees sabre' in **Akoka, J (Ed.)** *Management of Distributed Data Processing* North-Holland, Netherlands (1982)

44 **Karlsson, K** 'Le modele d'acces de la machine bases de donnees: SABRE' *Journees Machine Bases de Donnees* France (September 1980) p 14

45 **Karlsson, K** *'Reduced cover-trees and their application in the SABRE access path model'* SIRIUS Rep. No MBD-I-002 (February 1981) p 28

46 **Epstein, R** 'Why database machines?' *Datamation* (July 1983) pp 139–144

47 **Addis, T R** A relation-based language interpreter for a content-addressable file store. *ACM Trans. DB Syst.* Vol 7 No 2 (June 1982) pp 125–163

48 **Babb, M** 'Implementing a relational database by means of specialized hardware' *ACM Trans. DB Syst.* Vol 4 No 1 (March 1979) pp 1–29

49 **Carmichael, J W S** 'Personnel on CAFS: a case study' *ICL Tech. J.* (May 1981) pp 244–252

50 **Coulouris, G F, Evans, J M and Mitchell, R W** 'Towards content-addressing in databases' *Comput. J.* Vol 15 No 2 (1972) pp 95–98

51 **Maller, V A J** 'The content addressable file store — CAFS' *ICL Tech. J.* (November 1979) pp 265–279

52 **Maller, V A J** 'Information retrieval using the content addressable file store' *Proc. IFIP Congress 80* North-Holland (1980) pp 187–192

53 **Maller, V A J** 'Content addressing as an aid to information management' in *The fifth generation computer project* Pergamon (UK) 1983

54 **Lavington, S H and Azmoodeh, M** 'IFS — a proposal for a database machine' *Proc. 2nd British Nat. Conf. on DBs* UK (July 1982) pp 80–97

55 **McGregor, D R** 'The FACT system: a hardware-oriented approach' *Crest Course: Database — Role and Structures* UK (September 1982)

56 *The fifth generation computer project* State of the Art Report No 11:1 Pergamon Infotech, UK (1983)

57 **Moto-oka, T (Ed.)** *Fifth generation computer systems* North-Holland, Netherlands (1982)

58 **Aiso, H** 'Fifth generation computer architecture' in **Moto-oka, T** *Fifth generation computer systems* North-Holland, Netherlands (1982) pp 121–127

59 **Allen, J** 'Algorithms, architecture, and technology' in **Moto-oka, T** *Fifth generation computer systems* North Holland, Netherlands (1982) pp 277–281

60 **Moto-oka, T** 'Overview and introduction to the fifth generation' in *The fifth generation computer project* Pergamon Infotech, UK (1983) pp 3–16

61 **Suwa, M, Furukawa, K, Makinouchi, A, Mizoguchi, T, Mizoguchi, F and Yamasaki, H** 'Knowledge base mechanisms' in **Moto-oka, T (Ed.)** *Fifth generation computer systems* North Holland, Netherlands (1982)

62 **Tanaka, H et al.** 'The preliminary research of data flow machine and data base machine as the basic architecture of fifth generation computer systems' in **Moto-oka, T (Ed.)** *Fifth generation computer systems* North-Holland, Netherlands (1982) pp 209–219

63 **Treleaven, P C** 'Fifth generation computer architecture analysis' in **Moto-oka, T (Ed.)** *Fifth generation computer systems* North Holland, Netherlands (1982) pp 265–275

64 **Treleaven, P C and Lima, I G** 'Japan's fifth-generation computer systems' *Computer* Vol 15 No 8 (August 1982) pp 79–88