INTEGRATED SYSTEMS
I. DESIGN PRINCIPLES

Report No. 20 of the Series
Automatic Systems for Physical Sciences

Technical Report CSE-90-06

P. A. D. de Maine, K. D. Bradley, M. M. deMaine,
S. M. Jodis, H. W. Carlisle III, K-H Chang,
W. B. Day, C. M. Pancake, and W. B. Dress

Department of Computer Science and Engineering
Auburn University
Auburn, AL 36849-5347

September 13, 1990

INTEGRATED SYSTEMS
I. DESIGN PRINCIPLES

Prepared by:

P. A. D. deMaine, K. D. Bradley, M.M. deMaine,
S.M. Jodis, H.W. Carlisle III, K-H Chang,
W.B. Day, C.M. Pancake, and W.B. Dress

Report No. 20 of the Series:
Automatic Systems for Physical Sciences
Department of Computer Science and Engineering
Auburn University
Auburn, Alabama 36849

September 13, 1990

INTEGRATED SYSTEMS I. DESIGN PRINCIPLES

by

P.A.D. de Maine[#], K.D. Bradley, M.M. de Maine, S.M. Jodis,
H.W. Carlisle III, K-H. Chang, W.B. Day, C.M. Pancake
Computer Science and Engineering Dept.,
108 Dunstan Hall, Auburn University, AL 36849

and

W.B. Dress
The Oak Ridge National Laboratories
Oak Ridge, TN 37831

[#] To whom correspondence should be addressed (Auburn University).

TABLE OF CONTENTS

FIGURES

TABLES

ABSTRACT

The following design principles are being used in an ongoing project to realize an integrated family of expert systems that can be used separately or together in different combinations to solve problems common to many different disciplines. Some essential features of this family are:

(1) Individual members can be used in the normal way as expert systems or they can be transparently invoked by other expert systems without interrogating users.

(2) The knowledge (or rule) bases of key members do not mimic the perceived mode of human thought; therefore, they can predict events that cannot be predicted by state-of-the-art alone.

(3) The Law of Conservation of Mass/Energy is used to detect and correct computational errors.

I. INTRODUCTION

The physical and engineering sciences are essentially applications of subsets of the rules for combinatorial mathematics and classification theory. The rules for interpreting results and the various terminologies are characteristic of each discipline. Four problems common to many disciplines are:

(a) Manipulation of graphs. Computer assisted synthesis of chemical compounds in which the graphs represent chemical entities, medical diagnoses, and the design of computer chips are examples.

(b) Evaluation of graphs where the nodes designate states and edges transformations. Systems of equations and parsing trees are examples.

(c) Classification. Examples include the determination of controlling parameters and the classification of documents.

(d) Recognition. In the classic pattern recognition problem one either identifies objects with specific characteristics or determines objects that have unspecified common characteristics.

The family of general expert systems that has evolved consists of four Special Purpose Expert, SPE, systems and nine Artificial Intelligence System Support Tools (AISST) that can be used separately or together to solve class problems common to many different disciplines [1] (See Figure 1). The four SPE systems (SCANMAT, FRANS, AUTOLRN and AUTOREC) solve the four class problems in the order above. The nine AISST perform special tasks like curve-fitting, high-speed information management and data compression. All thirteen systems have user friendly interfaces, and can also be invoked by another expert system without interrogating users. There are prototypes for ten systems/tools and six of those are in production use. Details are found in [1] and the current status is given in Appendix V.1 of [4]. Here we shall describe the design principles

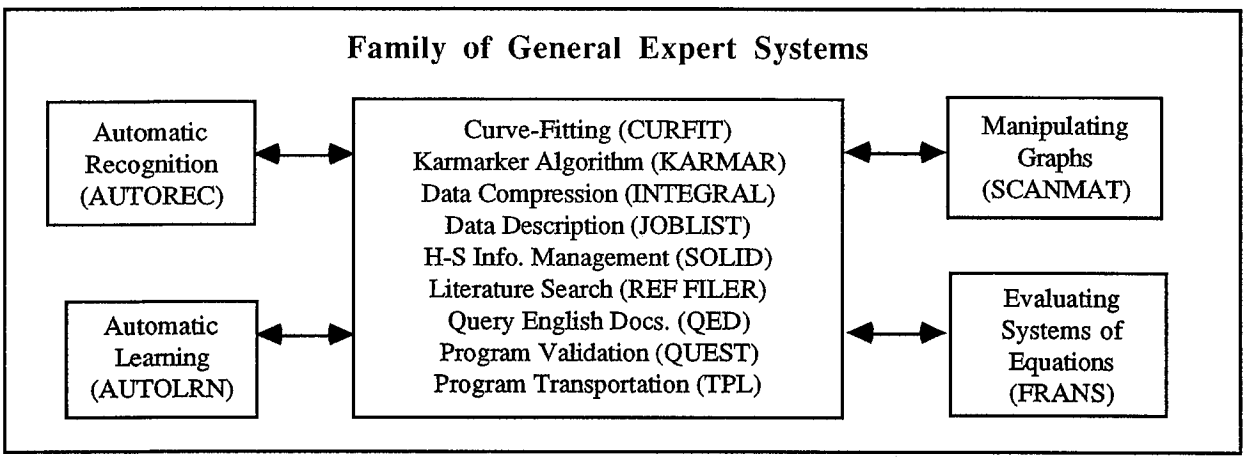

Figure 1. Relationships between the component Expert Systems for the planned Engineering and Science Expert Systems Package. The nine systems in the central box are the Artificial Intelligence System Support Tools (AISST) each of which can be used on a standalone basis. SOLID is a high-speed information management system. QUEST [2] and SCANMAT [3] are the work of groups headed by Dr. D. Brown and Dr. Hippe respectively. REF FILER is a propriety package that can be obtained from SOFT FOCUS. The functions of the individual components are briefly discussed in the text.

and then demonstrate fully operational systems. Details of fully operational systems will be presented in subsequent papers in this series.

II. DESIGN PRINCIPLES

The key concepts used in the ongoing work to realize a family of general expert systems that can be easily used in laboratories result from eight observations:

(1) The two mutually exclusive sets of requirements for Man-Machine interfaces, User-Friendly and Machine/System, can be achieved with a multi-tier design in which the user and system interfaces can be independently addressed.

(2) A simulated telephone network in which the queries themselves describe information path(s) that terminate with locations of the answer(s) is a high-speed, logically independent information management system.

(3) The physical and engineering sciences are essentially applications of combinatorial mathematics or classification theory that use subsets of the mathematical rules characteristic of the individual disciplines.

(4) A modified form of the fundamental Law of Conservation of Mass/Energy also applies to mathematical systems of equations.

(5) Transportation of high-level language code between different machine environments to obtain efficient object code can be achieved with a bifunctional compiler that supports an intermediate hypothetical high-level language that cannot be compiled.

(6) Scenes, which may have abstract characteristics, can be decomposed into one and/or two dimensional fundamental units, called normalized minimal views, that are the analog of letters in a natural language.

(7) The statistical based Salton method of text processing can be significantly improved by using the semantic based "aboutness" concept of Hillman and applications of neural networks to determine candidate terms or classifiers and the content of documents.

(8) Parallelism in sequential code can be exploited on high-performance workstations equipped with transputers.

Each observation is next discussed in turn.

III. DESIGN OF INTERFACES

The design of the man-machine interfaces for expert systems interrogation capability to serve (i) a variety of different classes of users from different disciplines and (ii) as service components in integrated packages is critical. The objectives of user-friendliness and integratability can be achieved with a multi-tier interface design whose two sections, User-Friendly and System, can be invoked independently of each other [5] (See Figure 2). The user-friendly section, which can be altered for new classes of users, passes information to the system section. The system section does not change for new applications and can be invoked without interrogating users.

IV. LOGICALLY INDEPENDENT INFORMATION MANAGEMENT SYSTEM

The success or failure of systems that require near continuous access to large databases (e.g. for engineering and science applications, defense applications, etc.) depends on high-speed general information management systems [6]. All attempts to realize applications with large and/or dynamically changing databases appear to be

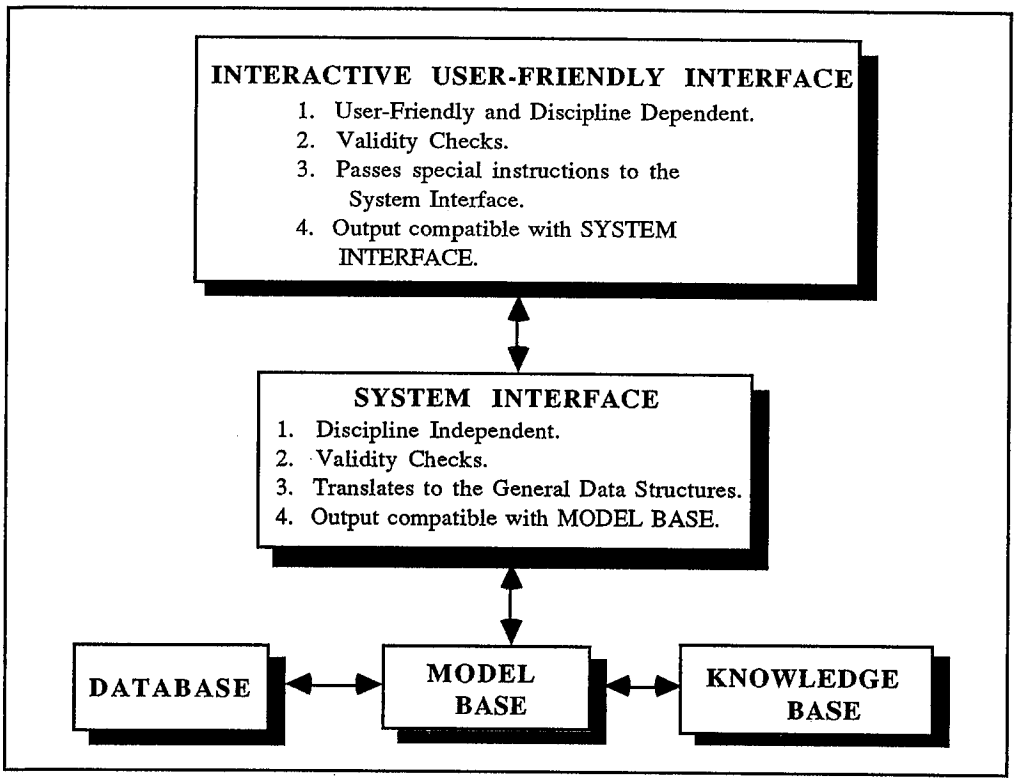

Figure 2. Multi-tier Interfaces. The expert system can be invoked by users via the Interactive User-Friendly Interface or transparently by another expert system via the System Interface.

of limited usefulness because they use data management methods whose efficiency is determined by the size and type of database (i.e. they are not logically independent). Large-scale pattern recognition systems like [7,8] also: (i) cannot handle objects that have abstract characteristics like the description of a function's behaviour; (ii) need excessive amounts of storage even for a modest library of objects; (iii) require excessive computer power and are frequently compute bound, requiring special machines like supercomputers [9]; and (iv) do not have easily invoked security mechanisms for the entire and/or parts of the system and its databases. Retrieval systems like [10-22], in addition to having inadequate management systems, suffer from faults (ii), (iii) and (iv). In addition document retrieval systems perform poorly because their indexing methods are inadequate.

The fundamental need for fast and efficient access to information in a variety of computer-based applications has directly influenced the design of information management systems. For example, the use of specially-designed database machines [23], data-flow machines [24], and data-compression techniques [25,26] have significantly improved the performance of such systems. Distributed computer systems that can efficiently manage large-scale, highly-distributed data and knowledge bases have been proposed [27]. There remains a need for the design of a fifth generation information system to manage all kinds of information (e.g. text, image data, voice data, etc. [28]) at very high speeds. Such a capability would significantly improve the usefulness of existing information systems

[29] and widen their applicability. Such a General Information Management, GIM, System must meet the following six general specifications.

(a) The GIM system should be information independent (i.e. data independent [30]).

(b) The GIM system should be question-type independent. An information management system is question-type independent if the structure or format of a request does not depend on the logical or physical organization of information managed by the system. Logical data independence [30] is equivalent to question-type independence.

(c) The time required by the GIM system to honor any request should be small and bounded. The GIM system should be able to service any request in a time virtually independent of either the type of query or the amount of information managed by the system.

(d) The GIM system must provide a security system that is "foolproof", "easy-to-use" and also ensure all databases are "virus proof".

(e) The GIM system should economically and efficiently utilize both primary and secondary storage and data communications facilities.

(f) The GIM system should be modularly designed so that it may function in either a distributed or stand-alone environment.

There is neither a known existing system that can meet all the above requirements nor one that can meet the critical requirements (b), (c), and (d). For example, the popular relational dBase III/IV [13] and INGRES [14-17] systems appear to satisfy only the first and sixth requirements. Document retrieval systems like SMART [12], LEADER [11], DIALOG [18,19], and MEDLARS/MEDLINE [20-22] are not question-type independent, do not have bounded retrieval times, and do not have an integrated, easy-to-use security system. The SOLID system, which will fully meet all the above requirements, is described next.

IV.1 The SOLID System:

The SOLID system [31,32] mimics the behavior of a communications network. During any sufficiently small interval of time, certain nodes of the network are sources of information ("transmitters") with matching sinks of information ("receivers"). Other nodes relay information (i.e. "a relay station", or "substation") while others are completely inactive (i.e. not serving as a source, sink, or relay station). All transactions have an associated set of nodes used to establish the path of communication. Assuming that source-to-sink communication contains at least one relay station, as in a conventional telephone network, the set of nodes corresponding to any communication path contains a source, sink, and one or many relay station nodes. To borrow from graph theory, this set of nodes is said to denote an information path through the communications network.

This model of a communications network is analogous to the design for an information retrieval system because: (1) A request for information, called a query, can be formulated as a sequence of subpaths. (2) The sequence of subpaths comprising the query corresponds to an information path through the communications network. (3) The terminating point of an information path specifies, either directly or indirectly, the location of the requested information.

By incorporating such network-like features into the design of the SOLID information retrieval system, the following important advantages are realized:

(a) The information retrieval system is information independent.

(b) The information path corresponding to any query may be traversed in a time that is virtually independent of the total number of nodes which exist in the network.

(c) Because an information path (or query) can be viewed as a sequence of subpaths, those subpaths which are common to a collection of similar information paths need be represented only once.

(d) Such a system is a positive retrieval system, in that the time required to determine that an information path does not exist is less than the time required to determine that it exists.

IV.2 The SOLID System Simulated Network:

The SOLID system utilizes three hierarchically-ordered files for storing information paths and information-path referenced information called the Registry File, the Address File, and the Main File (See Figure 3).

The Registry File (RFILE). The RFILE contains all the information paths. It terminates with indirect "pointers" (called Registry Numbers) to information-path referenced information. This file is the simulated communications network (Figure 3). The RFILE is a collection of memory blocks organized such that each information path is completely contained in one memory block. When an attempt is made to store an information path within any memory block which has approached its storage capacity, the SOLID system algorithmically partitions the "nearly full" memory block into two or more "half empty" memory blocks, making the SOLID system self-organizing.

The Main File. The Main File, also known as MFILE, contains all the information-path referenced information. To satisfy the GIM requirement that both secondary storage and data communication facilities be efficiently utilized, all information-path referenced

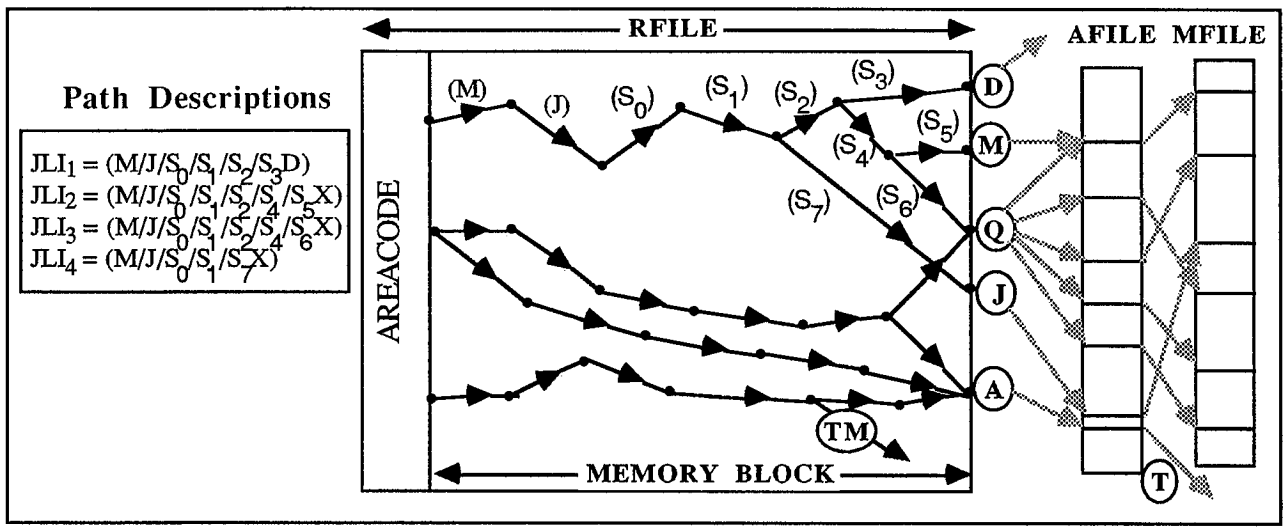

Figure 3. File Structure of the SOLID System. A, D, J, M, and T are Registry Numbers. D and T provide exits from RFILE and AFILE respectively. Requests with no answers are terminated via TM. The pages of RFILE, called memory blocks, fit in memory and information paths are restricted within memory blocks. The JLI is used to automatically select the required memory block.

information is normally stored in a form highly-compressed by the INTEGRAL family of reversible data compressors [25]. MFILE may be composed of any combination of magnetic disks, magnetic tapes, or bulk archival systems, which need not necessarily be on-line to the SOLID system host.

The Address File. The Address File, also known as AFILE, contains direct "pointers" to information-path referenced information located in MFILE. These "direct pointers", called Device Addresses, completely specify the locations of all information contained in MFILE.

To illustrate the hierarchical nature of these three files (Figure 3), consider how a simple retrieval operation is performed. First, a query in the form of an information path description (the JLI_i in Figure 3) is processed by the SOLID system. Given the information path description (i.e. query), the SOLID system utilizes a memory-resident function (called AREACODE) to determine which RFILE memory block completely contains the information path in question. If the RFILE memory-block is not memory-resident, it is fetched from secondary storage. Then the information path is traversed within the memory-resident RFILE memory-block and an associated registry number (indirect pointer) is obtained (e.g. A in Figure 3). The registry number is the terminating point of the information path. Second, the registry number obtained from the traversal of the information path in RFILE is used as a simple index into AFILE to obtain a device address. Finally, the device address found in AFILE is used to fetch the information-path referenced information

contained in MFILE.

IV.3 The JOBLIST Language:

Communication between the SOLID system and either a user-interface or other expert system or tool is provided by the JOBLIST language [31-33]. The JOBLIST language consists of four basic components: JOBLIST item (JLI), JOBLIST task (JLT), JOBLIST job (JLJ), and JOBLIST queue (JLQ). JOBLIST queues are constructed of JOBLIST jobs; JOBLIST jobs are constructed of JOBLIST tasks; and JOBLIST tasks are constructed of JOBLIST items. The hierarchical ordering of these components is defined as:

$$JLI = (M/J/S_0/S_1/S_2/ \cdot \cdot \cdot /S_{K-1}/S_K X)$$

$$JLT = (SPI, JLI_1\ IC_1\ JLI_2\ IC_2 - - - - - IC_{L-1}\ JLI_L)$$

$$JLJ = (JLT_1\ TC_1\ JLT_2\ TC_2 - - - - - TC_{M-1}\ JLT_M)$$

$$JLQ = (JLJ_1\ JC_1\ JLJ_2\ JC_2 - - - - - JC_{N-1}\ JLJ_N)$$

where IC_i, TC_i, and JC_i are item, task, and job logical connectors, respectively. In retrieval operations, these logical connectors provide the basic Boolean connectives (e.g. logical AND, logical OR, etc.). In storage, purge, and update operations, the logical connectors are simply ignored.

In the most elementary of these components, the JOBLIST item (JLI) [33] M and J are JLI control information, S_i are substrings called <u>kernel screens</u> (which are collections of <u>kernels</u>), and X is a control value used when new information paths are created. Kernels are bit strings and can be any size. They may contain any combination of special codes, keys, parts of keys or security locks. A key

may be viewed as part or the whole of an index term (or descriptor) that described referenced information. The following operations may be performed on JLI: addition, subtraction, multiplication, division, transposition, concatenation, interleave, expansion, or contraction.

Returning again to the simulated network analogy (Figure 3), a JOBLIST item, for which three forms exist [31-33], could be the description of an information path for explicit queries or a collection of paths for non-explicit queries. In a **Simple Form JLI**, which is useful for well-structured information, the kernel screens S_i's are independent of each other. The **Information Representation (IR) Form JLI** is a special linear-encoding scheme. The IR form can be used to represent static scenarios such as three-dimensional objects, graphs, or chemical formulae, as well as dynamic scenarios such as weather maps or flight-paths of ballistic missiles. The **Matrix Form JLI** is used to represent and manipulate square matrices.

IV.4 Override Codes, Information Security and Specificity:

The SOLID system utilizes nine generally-restricted codes, called override codes, for the purposes of information security and non-explicit query representation. Information security and non-explicit query representation will now be considered.

Information Security. Information security in the SOLID system is provided by static and dynamic security "locks" which can be placed in information paths and/or registry numbers contained in RFILE, in device addresses (i.e., AFILE), or within referenced

information itself (i.e., MFILE). A static security lock can be viewed as a kernel whose value and location in the JLI kernel screens are assigned by the user-manager. A dynamic security lock can be viewed as a special kernel which, when encountered, invokes an interrogation routine provided by the user-manager. These locks may be inserted during and update operations. A retrieval request will not be honored if the locations and values for the locks in the JLI are not supplied. Neither static or dynamic locks may be overridden by users, although the user-manager may update secure information paths via privileged operations. Dynamic security locks can also be used in write operations to protect secondary storage against intruders (e.g. viruses).

Specificity. In the communications network model, all information paths are assumed to be complete; there is no uncertainty in any subpath. To illustrate non-explicit queries, suppose that uncertainty is allowed in specifying an information path to be traversed (i.e., the query is now non-explicit). In this case of uncertainty (i.e., a lessened degree of specificity), multiple information paths may be traversed. Due to the seven override codes in the JOBLIST language, which denote uncertainty in the value of kernels, the location of kernels within screens, the location of screens within JLI, or any combination of these, any degree of specificity is allowed.

IV.5 Current Status:

Detailed specifications for the JOBLIST Language are given in [33]. Details of the planned role of the SOLID system in a distributed information network is described in [34]. Detailed specifications are given in [35]. Experiments with machine generated data and an IBM Basic Assembly Language, BAL, coded prototype have confirmed that the SOLID system fully meets all the specifications for a GIMS enumerated at the beginning of this section. The BAL coded prototype was used with a PL/1 coded front-end, called CSAR, to provide an online service for the Obstetrics Department of the Hershey Medical Center from the Pennsylvania State University [36].

The design for a distributed information network described in [34,36] has been changed in the following important ways.

1. The front-end has been replaced by a hierarchical arrangement of two different types of interactive interfaces - Subject (or User-Friendly Interactive) and System [5] - that can be used either in a single machine or in a networked environment. The single System interface is used by and of the Subject interfaces to construct the JOBLIST data-structures that are used by the SOLID system to describe information paths. The Subject interface accepts easily understood queries from users.

2. Transportability of programs is now achieved with the Transportable Programming Language, TPL, system [37-39]. The completed TPL system can be used to transport code in any high-level language that is supported in the object environment. The restriction on the coding language has been virtually eliminated.

3. The formal specifications of the JOBLIST Language in [33] have been completed.

The new design for interactive interfaces that will be used in all expert systems has been described in [5]. A prototype for the SOLID system has been implemented in FORTRAN 77 on the VAX 11/780. Ramakrishnan [40] transferred the AMDAHL 470E FORTRAN version of

SOLID to the VAX 11/780 and then designed and implemented the interactive SYSTEM interface. Head [41] designed and implemented an interactive SUBJECT interface for use by Electrical Engineers in testing designs of individual VLSI and designing VLSI aggregates (or CHIPS). Edhala [42] designed and implemented a version of the Query-by-Example, QBE, system that can be used as the interactive SUBJECT interface for many different applications.

V. CATEGORIES OF EXPERT SYSTEMS

Solutions to many problems in Engineering and Science involve the construction, manipulation and evaluation of two- and three-dimensional graphs. While the meaning of the graphs and the interpretation of the numerical values are exclusively the province of the separate disciplines, there are four classes of problems associated with particular collections of graphs that are common to many disciplines. The four classes of problems are: (i) Identification of entities; (ii) Classification of entities; (iii) Construction, decomposition, alteration and combination of graphs; and (iv) Evaluation of systems of equations. An entity is defined as an object, a characteristic feature (e.g. functional groups), or an abstraction (e.g. the behavior of a mathematical expression).

The design that has evolved [1] consists of four Special Purpose, SPE, expert systems and nine Artificial Intelligence System Support Tools, AISST (which are themselves expert systems that can be used on a stand-alone basis). The four SPE systems (AUTOREC,

AUTOLRN, SCANMAT and FRANS) solve the above four class problems in the order enumerated. AUTOREC and AUTOLRN are adaptive learning systems whose performance improves with use. SCANMAT and FRANS are Autodeductive systems that can predict unprecedented events (e.g. events that are not currently in the state-of-the-art).

V.1 Classification of Expert Systems:

Belev [43] has identified five components in expert systems; User Friendly Interface, Model Base, Rule Base, Text Base, and Data Base. This contrasts with the more common identification of the User Interface, Inference Engine, Knowledge Base and Data Base. The problem of providing user interfaces for different disciplines is simplified with the multi-tier design for interfaces (See Section III).

With the Belev scheme, expert systems can be classified as Conventional Expert, Autodeductive, Conventional Learning, and Adaptive Autolearning. In **Conventional Expert** systems, the rule base is constructed by examining the current state-of-the-art [44]. Such systems ultimately mimic the <u>perceived</u> way humans reason and thus they cannot predict events outside the current state-of-the-art.

In an **Autodeductive system**, ADS, the model base is a mathematical model. The rule base is either an integral, indistinguishable part of the mathematical model or it is generated by the system itself from fundamental laws such as the Conservation of Mass/Energy. Two types of ADS are:

(a) Numeric Autodeductive Systems, like FRANS [1,45], are nonsymbolic systems that are "philosophically closed" or "mathematically complete" wherein the rule base is an indistinguishable part of the mathematical model and there is no database.

(b) Alphanumeric Autodeductive Systems, like SCANMAT [3], are symbolic systems whose rule base is generated by the system itself without reference to either the state-of-the-art or the contents of the data base. Finally the data base contains a list of entities that have been identified in the "real world".

In Learning systems used to classify and/or identify entities, the model base (or inference engine) consists of one or more clustering algorithms. They are first used to identify the classification parameters for a known "training set", and then used to classify unknown sets. Adaptive learning systems augment the training sets with unknown sets that they have classified. The classification parameters are continually recomputed, thereby improving their predictive capabilities. There are two kinds of adaptive learning systems [1]. **Conventional Learning systems** use manually classified training sets; therefore, they cannot predict events that cannot be determined from the state-of-the-art. **Adaptive Autolearning systems** classify their training sets, are not bound by the state-of-the-art, and are able to predict unprecedented events.

The adaptive autolearning system, AUTOLRN, will use a combination of the Borko-Bernick Factorization [46] and the Williams Discriminant [47] methods in the following step-wise fashion.

(a) Use the QED system that is described in Section IX to determine candidate descriptors for all documents or entities prior to constructing a weighted frequency matrix.

(b) Calculate eigenvalues from the frequency matrix.

(c) Identify categories by the ranges of real eigenvalues.

(d) Use the Williams Discriminant method to determine the descriptors for the collection. In this step candidate descriptors can be eliminated and categories can be combined.

(e) With the output of (d) repeat (b), (c) and (d) until both methods [46,47] yield the same descriptors and classification categories.

VI. DETECTING AND CORRECTING COMPUTATIONAL ERRORS

Realization that a modified form of the Law of Conservation of Mass/Energy also applies to mathematical systems of equations has led to the development of the fully implemented Error Detection and Corrective Action, EDCA, algorithm for use in solving boundary valued problems [48]. The user stipulated Computational Accuracy is used to detect mathematical instability and, if possible, to initiate corrective actions until either an acceptable computational accuracy or the limit of computational accuracy is achieved. This method has been used to greatly extend the range of the commonly used Newton-Rhapson and Hamming-Kutta-Rung methods [48,49].

VII. TRANSPORTATION OF HIGH-LEVEL LANGUAGE CODE (TPL)

The TPL method of transporting high-level language code is designed to convert code from a source environment to an object environment so that it will efficiently execute [38,39]. A bifunctional compiler installed in the source environment converts the source code into a Hypothetical High-Level Language, HPHLL. The same compiler installed in the object environment converts the HPHLL to efficient object code for use on the object machine. The advantages

of this method are: (a) the TPL system is installed only once in each new environment; (b) there is no restriction on the source or object languages; and (c) programmers do not have to know anything about the HPHLL. This method has been successfully applied for inter-language conversion [38,39] and it can be used for intra-language conversion (e.g. FORTRAN to ADA).

VIII. DECOMPOSITION OF SCENES

Virtually all picture processing or pattern recognition systems appear to decompose observations by syntactic, statistical or geometric methods into scenes and then identify entities in terms of the component scenes [50-55]. The problems associated with this approach are:

(i) Difficulties in processing entities that have abstract properties, like parts of the signatures of submarines or the properties of classes of functions.

(ii) Unmanageable databases, even for modest collections.

(iii) The generally poor performances regarding processing speed and reliability of the results.

The approach that is described in detail in [56] will be the basis for the image processing system, AUTOREC. The essential concept is that an observation can be decomposed to sets of non-decomposable patterns of interrelated identifying characteristics called minimal views. Minimal views can be regarded as "letters" that can represent abstractions (e.g. a property of a function or the color of an object) as well as two dimensional views of an n-dimensional

object. An **entity** is completely defined by an unique combination of minimal views. The immediate consequences are:

(i) There are provisions for contributions by both abstract and real (or physical) properties of entities.

(ii) Two databases are required. The Minimal View Library is small because a limited number of views describe a large number of different entities. Although the Entity Library, which specifies entities in terms of minimal views, must contain an entry for each different entity, it will be much smaller than conventional databases because minimal views are designated by numbers.

(iii) The certainty of identification is determined only by the weights assigned to the minimal views in the entity library.

IX. TEXT PROCESSING

With respect to both their speed of retrieval and the accuracy of answers, all existing major document retrieval systems appear to be inadequate because the database management systems that are used are slow and syntactic methods are used to determine indexes (or descriptors). For example, experiments with the most comprehensive implementation, the SMART system [12,57], were performed on a collection of about 1,000 documents. Even this small number of documents returned poor values for the ratio of Recall/Precision that measures accuracy [12]. The large-scale medical applications (MEDLINE [20,22] and MEDLARS [21]), DIALOG [18,19], and the Chemical Abstract systems are all keyword driven and thus of limited usefulness. There appears to be no system like the prototyped Query English Document, QED, system that can discover the meaning of documents (See Subsection IX.1).

The problem of determining candidate descriptors or classifiers in natural language text has both semantic and syntactic aspects. The time consuming semantic methods that were pioneered in the DEACON system [10] are not a practical basis for managing large collections of documents. On the other hand the LEADER system [11] uses philosophically derived techniques to perform high-speed semantic analysis. The most comprehensive of the many attempts to assign descriptors by predominantly syntactic methods is found in the SMART system [12,57].

The basis of the LEADER system is the "aboutness" concept described by Hillman [58]. A high-speed surface-level parser is used to convert English language sentences to simple logical constructs, referential sentences are identified, and then the referential terms in such sentences are selected as descriptors. Key problems in this approach are:

(i) There is a need for a high-speed data management system to support the libraries required for the analysis and also to manage the principal database (a library of condensed text) which contains the documents expressed in the referential sentence form.

(ii) There seems to be no natural way to assign weights to descriptors and reduce or expand the descriptor sets assigned to documents or queries.

However, reduction and expansion of semantic descriptor sets can be achieved by processing the natural language text, before it is parsed, with artificial neural networks in the following ways:

(a) Anticipatory system [59] can be used to guess the needs of the user as well as interpret what the user "really means" in case of ambiguous queries.

(b) Expectation failure, which occurs when none of the chosen pathways is realized, can be used as the basis for future predictions [60] in a self-correcting fashion.

(c) The context of a word (text) or image fragment (minimal view) is the totality of its proximity relationships with other objects. The meaning of an object resides in the restricted context currently in use. This restricted context is a dynamic subset [61] of the context obtained from the (text or pictorial) database, and changes as the user interacts with the system. That means that as major fragments of new knowledge are incorporated into the system, the overall context will have to be altered. If symbolic labels are assigned to fragments and consistent sets of fragments, a relationship (in the mathematical sense) may be constructed between the two context sets.

(d) The actions that a system produces result from the interaction of the internal states of the system and any external input. In this fashion, behavior imposes a semantics. If this behavior is learnable and modifiable by means of dynamic reclassification of the concept structure [62], then the semantics is adaptive in that it changes in time with the use of the system.

In the SMART system [12,57] statistical techniques are used to determine candidate words and phrases. Then a novel non-statistical technique is used to convert candidate words to the null stems (i.e. stems common to different words) that are is the basis of the indexing scheme. Statistical techniques are used to construct concept classes (for expanding descriptor sets), concept hierarchies (for reducing and expanding descriptor sets), and statistical phrases (to expand descriptor sets). The conventional phrase dictionary mentioned in [12] does not appear to have been implemented.

An essential prerequisite for a retrieval system for large scale applications is a high-speed database management system that fully meets the six general specifications given at the start of Section IV. The above noted inadequacies of the purely statistical (or syntactic) and the purely semantic methods of indexing can be removed by this step-wise procedure in storage operations.

(1) Use the standard frequency counting methods, with positive and negative dictionaries, to determine candidate terms.

(2) Determine the referential terms by the Hillman method, augmented by applications of artificial neural networks.

(3) Obtain the final list of candidate terms by deleting all terms from the Salton list that are not in the Hillman list and adding those in the Hillman list that are not in the Salton list.

(4) Use Salton's suffix list to obtain the null stems for all terms in the final list, obtained in (3).

(5) Use Salton's statistical methods to determine Concept Classes, Concept Hierarchies, and the two phase dictionaries (statistical and syntactical).

In retrieval operations the list of null stems obtained in (5) is augmented with information retrieved from the Concept Dictionary, the Concept Hierarchies, and the two phrase dictionaries. Answers are a ranked list of documents. The "Aboutness" of individual documents will be disclosed by an abstract that displays the key logical constructs in natural language form.

IX.1 Proposed Implementation of the Combination Method:

The text processing part of this project will yield a fully integrated package of six expert systems that consists of a principal system, AUTOTEXT, which transparently uses five service systems (AUTOLRN, CURFIT, JOBLIST, QED and SOLID). CURFIT is a completed curve-fitting system. The roles of the remaining five are discussed next.

AUTOTEXT is the driver that invokes QED to determine candidate terms (or the "aboutness") from documents or queries, executes Salton's statistical procedures (in the storage mode) and then, in the retrieval mode, retrieves the relevant documents in their logical construct form.

AUTOLRN will be invoked by AUTOLRN itself, to perform parts of the statistical analysis, and by QED when it classifies documents.

JOBLIST converts queries to the JOBLIST form that is required by SOLID. It will be transparently invoked by AUTOTEXT and QED.

QED is the application of Hillman's (augmented with applications of artificial neural networks) and Salton's methods for determining both descriptor terms and the "aboutness" of documents; and the combination Borko-Bernick and Williams method of classifying documents. AUTOLRN, CURFIT and SOLID will be used to accomplish these tasks.

SOLID is the high-speed general information management system that will be used to manage all libraries. It will be invoked by AUTOTEXT and QED.

X. USE OF TRANSPUTERS

The INMOS Transputers family of microprocessors provides on-chip multiprocessing and communication capabilities. By linking this processor and interface boards to a workstation, one builds a simple multiprocessor system that can communicate both within the system and between workstations using ethernet connections. Tools such as C, FORTRAN , PASCAL and OCCAM compilers support software development on these systems. Operating system extensions such as Cornell University's TROLLIUS provide the C and FORTRAN functions for internal (in the box) and external (out of the box) communications.

Scalability is a benefit of Transputer systems. The TRAM system approach now utilized by SGS Thompson's INMOS division allows new modules to be plugged into existing systems to increase processing capabilities. Thus prototype systems may be easily expanded into production environments.

Our goal is to produce systems that can be easily used on any machine configuration with sufficient storage for the application. To reduce the cost and improve accessibility, all developmental work will be done on a network of high-performance workstations, each equipped with transputers. To exploit parallelism, algorithms will be developed to convert transportable sequential code for use on each separate workstation.

XI. APPLICATIONS

Versions of the curve-fitting, CURFIT, and modeling, FRANS, systems are in production use on PC AT and VAX 780 machines. Here are two examples, taken from a presentation prepared for the American Institute of Chemical Engineers [73], intended to direct attention to publications that illustrate use of our systems to solve hitherto intractable problems.

XI.1 CURFIT System:

In typical regression analysis, constant parameters are calculated from variable parameters by statistical or graphical curve-fitting methods. In such methods average measures, such as average deviation or residuals, are the basis of the "goodness of fit" criteria and are used to reject spurious data (or outliers) and to detect unsuspected curvatures. However, comprehensive experiments with both machine generated and laboratory data [63] have established that such methods frequently yield ambiguous and even spurious

answers. Improvement comes only with methods that use a measure of reliability for every item of raw data.

In the Maximum Tolerance method of curve-fitting [1,48,64] the traditional question "Do these data fit this (user specified) equation?" is replaced by "Do these data fit this (user specified) equation to within user stipulated maximum tolerances?". The user-supplied maximum tolerances specify the maximum uncertainty (or reliability) for every item of raw data. They are used to compute the maximum acceptable deviation, called the Error Bound, of each value for every variable in the fitting equation. The error bounds define a vortex about the data-points. They are used to (i) reject spurious data, (ii) detect unsuspecting curvatures, and (iii) ultimately compute the maximum possible error, called the Maximum Error, in every value for all single-valued parameters. If unsuspected curvature is detected, the data are partitioned into separate overlapping domains.

The unambiguous criteria of "goodness of fit" are:

1. The user-supplied maximum tolerances and the rejected data-points.
2. The number of overlapping domains.
3. The calculated values for every parameter and its associated maximum error in every domain.

There is a perfect fit only if there are no rejected data-points, one domain, and all maximum errors are vanishingly small.

The CURFIT system can be used to directly and unambiguously compare either the fit of different sets of data to an equation or

the same set of data to different equations. Within available computer resources, there is no restriction on the dimension of the fitting equation or the number of data-points that can be processed. The interactive user-friendly interface is described in [65]. Examples of the use of CURFIT may be found in [1,5,48,63,64,66-67] and especially in the references cited in [68].

X.1.1 CURFIT Example:

For illustrative purposes consider the data cited in Table 7 of [64] for heat transfer outside thin plain tubes. The problem is to determine whether they are described by the equation:

$$Nu = aRe^{b}Pr^{c}r^{d}$$

(i) Nu, Re and Pr are the Nusselt, Reynold and Prandtl number respectively, (ii) r is the ratio of the viscosity of the mean fluid temperature to that at the wall temperature, and (iii) a, b, c and d are constants that are to be calculated. A linear form of the above equation is:

$$\ln(Nu) = \ln(a) + b\ln(Re) + c\ln(Pr) + d\ln(r)$$

The maximum tolerance, designated as fractions of the observables thus: MT_{Nu}, is related to the error bound for the variables in the fitting equation. Thus for y the error bound, EB_y, is:

$$EB_y = \ln(Nu(1+MT_{Nu})) - \ln(Nu) = \ln(1+Mt_{Nu}) \qquad (A)$$

The results are summarized in Table 1, which is taken from Table 8 in [64]. Two different experiments were performed. In both experiments the maximum tolerances for the Re, Pr, and r were set to zero. For the first computer experiment, all data-points were processed in

MT_{Nu}	First Domain Re Range	DPts	Second Domain Re Range	DPts	Third Domain Re Range	DPts
First Experiment:						
≤ 0.05	54-1249	11	465-84,800	14	≥ 34,200	8
0.06	54-49,000	22	≥ 95.9	18		
0.08	54-84,800	26				
Second Experiment:						
0.019	<10,000	10				
0.013	>10,000	10				

Table 1

Summary of the results from CURFIT for the fits of the heat transfer data to equation (A). MT_{Nu} = 0.08 means the maximum tolerance for Nusselt numbers are eight percent. All other maximum tolerances are zero. Re Range is the Reynold number range of the domain, and DPts is the number of data-points used. In the second experiment the data were partitioned into two physically distinct segments.

a single continuous segment, and the maximum tolerance for the Nusselt numbers was varied from one to eight percent (i.e. with MT_{Nu} 0.01 to 0.08). Data-points 8, 9 and 11 were always rejected. Such results establish that:

1. Data-points 8, 9 and 11 are not reliable.
2. To within eight percent of the Nusselt number, these data are described by equation (A).
3. For $MT_{Nu} \leq 0.06$ these data are described by two relationships defined by the magnitude of Re.

For the second kind of experiment, the data were partitioned into two non-overlapping segments, with a value of 10,000 for Re the dividing point. For each segment the value of MT_{Nu} was varied from 0.01 until no additional data-points were rejected. The results in Table 1 show that (i) the data describe two competing processes, (ii) the reliability for the data with Re < 10,000 is near 1.9%, and (iii) the reliability of the data with Re > 10,000 is near 1.3%.

XI.2 FRANS System:

The current versions of the Function Recognition And Numerical Solution, FRANS, system can solve reaction networks with any combination of nonlinear (equilibrium) and first order ordinary (rate) equations. Within the available computer resources there is no limit on either the size or complexity of the network.

FRANS operates in two modes. The Predictive mode determines the combinations of parameters whose values are required to compute a subset of parameters. It is especially useful in designing experiments. The Computative mode uses the computational error procedures

that are described in the second part of [48]. Examples of the use of FRANS will be found in [1,5,45,68,70] and especially in [69].

The user-friendly part of the two-tier interface has been described in [71], and the Error Detection and Corrective Action procedure, the EDCA algorithm, in [48,70]. The use of the predictive mode is demonstrated next.

IX.2.1 Prediction Example

An example for a proposed model for nitrosation in glacial acetic acid [72] is presented. Table 2 contains the description of the model generated by the user-friendly part of the two-tier interface. "SELECT = 1" specifies a predictive problem, and "TOTAL = 1" means that only total predictions (i.e. those in which all parameters are calculated or given) are to be displayed. The input status for every parameter may be designated as given (> 0), not measurable (< 0, the default) or may be computed or given (= 0). Thus H^+, NO_2 and RK1 are designated given, may be computed or given, and not measurably respectively. Table 3 contains the six total predictions, each corresponding to a possible experiment, that can be used to establish the validity of the proposed model. All other possible combinations (there are more than 8,000,000) either require information that is not used or cannot validate the model.

XII. CONCLUSIONS

The design principles for a set of thirteen general expert systems that can be easily used in laboratories to solve problems

```
/: PREDICTION FOR NITROSATION IN GLACIAL ACETIC ACID :/
/: PAGE 18-19 IN - NITROSATION - BY D.L.H. WILLIAMS, :/
/: CAMBRIDGE UNIVERSITY PRESS, NEW YORK (1988) :/
SYSTEM:
 SELECT=1;
EQUATIONS:
 HNO2 + H <=> H2NO2, RK1, RK2;
 H2NO2 + SUBSTRAT -> PRODUCT, RK3;
 H2NO2 + CH3COO <=> CH3COONO + H2O, RK4, RK5;
 H2NO2 + NO2 <=> N2O3 + H2O, RK6, RK7;
 CH3COONO + NO2 <=> N2O3 + CH3COO, RK8, RK9;
 N2O3 + SUBSTRAT -> PRODUCT, RK10;
 CH3COONO + SUBSTRAT -> PRODUCT, RK11;
/: IONIZATION OF GLACIAL ACETIC ACID IS NEXT :/
 CH3COOH <-> H + CH3COO, EKA;
CONSTANTS:
/: THE FOLLOWING CAN BE GIVEN :/
 H=1; CH3COOH=1; EKA=1; PRODUCT=1;
/: THE FOLLOWING MAY BE GIVEN :/
 HNO2=0; SUBSTRAT=0; CH3COO=0; NO2=0; N2O3=0; CH3COONO=0;
 H2O=0; H2NO2=0;
/: ALL OTHER CONSTANTS CANNOT BE GIVEN :/
SELECTOR:
/: ONLY TOTAL SOLUTIONS ARE TO BE DISPLAYED :/
 TOTAL=1;
STOP:
```

Table 2

This input file describes the model for nitrosation in glacial acetic acid given on pages 18 and 19 of [20]. EKA is the ionization constant for glacial acetic acid. RKx designates a specific rate constant with RK1, RK4, RK6 and RK8 for the forward reactions (i.e. RK1 is for $HNO_2 + H^+ \rightarrow H_2NO_2^+$). H, H2NO2, NO2 and CH3COO are the ions H^+, $H_2NO_2^+$, NO_2^- and CH_3COO^-. The values entered for parameters in the CONSTANTS dataset specify: (i) H^+, CH_3COOH and EKA will be given; (ii) all other reactants may be given or computed; and (iii) all rate constants default to cannot be given (i.e. RK1=-1). SELECT=1 designates a prediction problem and TOTAL=1 specifies that only complete solutions are to be displayed.

THE PREDICTIONS THAT WERE OUTPUTTED BY THE FRANS SYSTEM ARE TO BE TABLED NEXT. G - GIVEN; R - RECOMPUTED; BLANK - NOT COMPUTED; N - CANNOT BE GIVEN; NC - CANNOT BE GIVEN BUT CAN BE COMPUTED.

THE MULTIPLICITIES OF GIVEN REACTANTS ARE DESIGNATED BY N THUS :- N*G. THE PREDICTION WITH THE LOWEST MULTIPLICITY GENERALLY YIELDS THE LOWEST MAXIMUM ERRORS FOR THE REACTION CONSTANTS WHEN THEY ARE COMPUTED WITH THE CURFIT SYSTEM.

THE PREDICTOR COMMAND (PRECOM) IS 1 AND THE COMPUTATIONAL STATUS OF NO VARIABLE WAS SPECIFIED.

6 NON-REDUNDANT PREDICTIONS ARE TABLED NEXT.

NAME	1	2	3	4	5	6
HNO2	C	C	C	C	C	C
H	2*G	2*G	2*G	2*G	2*G	2*G
H2NO2	7*G	7*G	7*G	7*G	C	C
SUBSTRAT	C	C	C	C	C	C
PRODUCT	3*G	3*G	3*G	3*G	3*G	3*G
CH3COO	C	C	C	C	C	C
CH3COONO	5*G	5*G	C	C	5*G	5*G
H2O	4*G	C	4*G	C	4*G	C
NO2	C	4*G	C	4*G	C	4*G
N2O3	C	C	5*G	5*G	5*G	5*G
CH3COOH	0*G	0*G	0*G	0*G	0*G	0*G
RK1	NC	NC	NC	NC	NC	NC
RK2	NC	NC	NC	NC	NC	NC
RK3	NC	NC	NC	NC	NC	NC
RK4	NC	NC	NC	NC	NC	NC
RK5	NC	NC	NC	NC	NC	NC
RK6	NC	NC	NC	NC	NC	NC
RK7	NC	NC	NC	NC	NC	NC
RK8	NC	NC	NC	NC	NC	NC
RK9	NC	NC	NC	NC	NC	NC
RK10	NC	NC	NC	NC	NC	NC
RK11	NC	NC	NC	NC	NC	NC
EKA	G	G	G	G	G	G
	1	2	3	4	5	6

**** OUT OF 166 ORIGINAL PREDICTIONS 6 TOTAL PREDICTIONS ****
**** WITH 1 OR MORE GIVEN PARAMETERS WERE NOT REDUNDANT ****

Table 3

Pertinent part of the output from the FRANS system for the reaction model in Table 2.

common to many disciplines. The key principles result from observations that: (i) the two mutually exclusive sets of requirements for Man-Machine Interfaces can be achieved with a multi-tier design in which the user and system interfaces can be independently addressed; (ii) physical and engineering sciences are essentially applications of combinatorial mathematics and classification theory that use subsets of the mathematical rules characteristic of the individual disciplines; and (iii) a modified form of the fundamental law of conservation of Mass/Energy also applies to mathematical systems of equations.

XIII. REFERENCES

[1] de Maine, P.A.D. and M.M. de Maine, "Computer Aids for Chemists", Anal. Chim. Acta 235, 7-26 (1990).

[2] Brown, D.B., K.D. Haga and O. Weyrich, "QUEST - Query Utility Environment for Software Testing", International Test and Evaluation Association 1986 Symposium Proceedings, pp 38-43 (1986).

[3] Hippe, Z.S., "Artificial Intelligence in Chemistry: Present Status and Future Goals", Proc. 7th. Int. Conf. on Comp. in Chem. Research and Education, Eds. Josef Brandt and Ivar K. Ugi, Dr. Alfred Huthig Verlag GmbH, Heidelberg, West Germany, pp 165-178 (1989).

[4] de Maine, P.A.D. and K.D. Bradley, "Development and Application of a Family of General Expert Systems", Report No. 18 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - 40 pages (August/1990).

[5] de Maine, P.A.D., and M.M. de Maine, "A Computer Tool Kit for Chemists I. Design Considerations for Interfaces", J. Chem. Inf. & Comp. Sci. 30, 155-159 (1990).

[6] van den Burg, J., "Telecommunications-Related Information Services", Proceedings of the NATO Advanced Study Institute on Information Technology and the Computer Network, Bonas, France, August 21 - September 2, 1983.

[7] Kanade, T., "Development of Autonomous Systems", Proc. Appl. Art. Intel. VII, 569-573 (1989).

[8] Laird, J.E., "Learning from External Environments Using SOAR", Proc. Appl. Art. Intel. VII, 575-576 (1989).

[9] Sanz, J.L.C., "Is Architecture for Computer Vision and Image Processing a Dead Area of Research", IEEE Technical Committee NEWSLETTER, 6-10 (1990).

[10] Craig, J.A., S.C. Berezner, H.C. Carney, and C.T. Longyear, "DEACON: DIRECT ENGLISH ACCESS AND CONTROL", Proc. Fall Joint Computer Conference 365-380 (1966).

[11] Hillman, D.J., and A.J. Kasarda, "The LEADER Retrieval System", AFIPS Proceedings, **34**, 447-455 (1969).

[12] Salton, G., Ed., "The SMART Retrieval System: Experiments in Automatic Document Processing", Prentice-Hall, Englewood Cliffs, New Jersey (1971).

[13] Bharucha, K.D., "dBASE III PLUS: A Comprehensive User's Manual", Tab Books, Inc., Blue Ridge Summit, PA (1986).

[14] Held, G.D., M.R. Stonebraker, and E. Wong, "INGRES - A Relational Data Base System", Proceedings NCC, 44, (May 1975).

[15] Stonebraker, M.R., E. Wong, and P. Kreps, "The Design and Implementation of INGRES", ACM Transactions on Database Systems, 1, #3, 189-222 (1976).

[16] Stonebraker, M, "Retrospection on a Data Base System", A.C.M. Trans. on Database Systems **5**, #3, September (1980).

[17] Allman, E., and M. Stonebraker, "Observations on the Evolution of a Software System", I.E.E.E. Computer **15**, 3, 27-32 (1982).

[18] Bourne, C., and B. Anderson, "DIALOG Lab Workbook", Lockheed Information Systems, (1979).

[19] "Enhanced DIALOG Searching: An Introduction to DIALOG Version 2", DIALOG Information Services, Inc., Palo Alto, CA (1985).

[20] National Library of Medicine, "On-Line Services Reference Manual", U. S. Department of Health, Education and Welfare (January 1980).

[21] National Library of Medicine, "MEDLARS The Computerized Literature Retrieval Services of the National Library of Medicine", Department of Health, Education and Welfare, Publication NIH 79-1268, (January 1979).

[22] McCarn, D.B., "MEDLINE: An Introduction to On-Line Searching", Journal of the American Society for Information Science, 31, #3, 181-192 (1980).

[23] An issue devoted to data base machines, IEEE Computer, 12, #2 (1979).

[24] Boral, H. and D.J. Dewitt, "Applying Data Flow Techniques to Data Base Machines", IEEE Computer, 15, #8, 57-64 (1982).

[25] Bray, J.M., V.P. Nelson, P.A.D. de Maine and J.D. Irwin, "Data-Compression Techniques Ease Storage Problems", Computer Design, 24, #14, 102-106 (1985).

[26] Cormack, G.V., "Data Compression on a Database System", Communications of the ACM, 28, #12, 1336-1342 (1985).

[27] Kleinrock, L., "Distributed Systems", Communications of the ACM, 28, #12, 1200-1213 (1985).

[28] Neuhold, E.J., "Views on Data Management", ACM SIGMOD Record, 13, #4, 3-5 (1985).

[29] Date, C.J., "Database Usability", ACM SIGMOD Record, 13, #4, 1 (1985).

[30] Date, C.J., "An Introduction to Database Systems", Volume I Fourth Edition, Addison Wesley Publishing Company, Inc., New York, NY (1986).

[31] Wojtyna, M.S., and P.A.D. de Maine, "SOLID: A High-Speed Data- and Question-Independent Information Management System", Proc. 1988 ACM Sixteenth Annual Computer Science Conference, 652-657 (1988).

[32] de Maine, P.A.D., and C.G. Davis, "JOBLIST - A General Data-Structure for Communicating, Manipulating and Managing Information in a Distributed Environment", Proceedings Third International Conference on Distributed Computing Systems, 98-103 (1982).

[33] de Maine, P.A.D., and D. E. Whitten, "Automatic Organization of Files III. JOBLIST Language", Management Datamatics, 4, #2, 21-47 (1975.

[34] de Maine, P.A.D., and D.E. Whitten, "Design for a Distributed Information Network", Advances in Information Systems Science 7, 89-155 (1978).

[35] de Maine, P.A.D. and K.D. Bradley, "Research Notes for the JOBLIST/SOLID System", Report No. 19 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - XX pages (September/1990).

[36] O'Kane, K.C., P.A.D. de Maine and R.J. Hildebrandt, "A Proposed Regional Health Information Processing System", Management Datamatics **4**, 139-147 (1975).

[37] Jodis, S.M., "An Interactive Environment for the Transportation of Programs", Proc. 27th. Ann. Southeast Region ACM Conf., 260-266 (1989)

[38] de Maine, P.A.D., S. Leong and C.G. Davis, "A Transportable Programming Language (TPL) System I. Overview", Int. J. Comp. and Information Sciences 14, #3, 161-182 (1985).

[39] S. Leong, S. Jodis, K. Sullivan, O. Jiang and P.A.D. de Maine, "A Transportable Programming Language (TPL) System II. The Bifunctional Compiler System", I.E.E.E. Trans. on Software Engineering 16, #6, 639-646 (1990).

[40] Ramakrishnan, R, "Design and Implementation for a High-Speed Retrieval System with Bounded Search Times", M.S. Thesis, Auburn University (1986).

[41] Head, D.M., "The SOLID - User Interface", M.S. Thesis (1987).

[42] Edhala, M., "Design and Implementation of a Query-By-Example Interactive Interface for the SOLID System", Report in preparation.

[43] Belev, R.H., In the Book: "Knowledge Representation for Decision Support Systems", North-Holland Publishing Company, Amsterdam (1985).

[44] Levine, R.I., D.E. Drang and B. Edelson, "A Comprehensive Guide to AI and Expert Systems", MacGraw-Hill College Book Division, New York (1986).

[45] de Maine, P.A.D., and M.M. de Maine, "Automatic Deductive Systems I. Chemical Reaction Models", Comp. and Chem. 11, 49-65 (1987).

[46] Borko, H., and M. Bernick, J. Assoc. Comput. Mach., 10, 151 (1963).

[47] Williams, J.H., Proc. Fall Joint Comput. Conf. 161 (1963).

[48] de Maine, P.A.D. and M.M. de Maine, "Procedures for Detecting and Correcting Computational Errors", Report No. 17 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - 45 pages (July/1990).

[49] de Maine, P.A.D. and M.M. de Maine,"A Computer Tool Kit for Chemists and Engineers II. Reaction Modeling (FRANS System)", Chimicaoggi (Italy) - In press.

[50] Gonzalez, R.C. and Safabakhsh, R., "Computer Vision Techniques for Industrial Applications and Robot Control", IEEE Computer 15,17 (1982)

[51] Lee, C.S.G., Gonzalez, R.C. and Fu, K.S., "Tutorial on ROBOTICS Second Edition", IEEE Computer Society Press - Washington DC (1986).

[52] Gonzalez, R.C. and Wintz, P., "DIGITAL IMAGE PROCESSING - 2ND. Edition", Addison-Wesley Book Express (1987).

[53] Fu, K.S., Lee, C.S.G. and Gonzalez, R., "ROBOTICS: CONTROL, SENSING, VISION, & INTELLIGENCE", McGraw Hill (1987)

[54] Eds: Young, T.Y. and Fu, K-S, "Handbook of Pattern Recognition and Image Processing", Academic Press - Harcourt Brace Jovanovich (1986).

[55] Fukunaga, K., "INTRODUCTION TO STATISTICAL PATTERN RECOGNITION", Academic Press Inc. (1990).

[56] de Maine, P.A.D., Peter A-B. Ng and M.M. de Maine, "Perception and Identification of Entities in Different Environments", Report No. 11 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - 35 pages (1987).

[57] Salton, G., and M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, Inc., 118-145 (1983).

[58] a. Hillman, D.J., "Mathematical Theories of Relevance with Respect to the Problems of Indexing. Report No. 2: An Algorithm for Document Characterization", Report to the National Science Foundation on Grant No. GN-177, 12 March 1965.
b. Hillman, D.J., "Document Retrieval Theory, Relevance, and the Methodology of Evaluation. Report No. 1: Characterization and Connectivity", Report to the National Science Foundation on Grant No. GN-451, 24 May 1966.

[59] Rosen, R., "Anticipatory Systems", Pergamon Press, Oxford, 1985

[60] Andreae, J.H., "Thinking With The Teachable Machine", Academic Press, New York, 1977

[61] Rosen, R., "Some epistemological issues in physics and biology", p. 322 Quantum Implications, Eds. B.J. Hiley and F.D. Peat, Routledge & Kegan Paul, New York, 1987

[62] Edleman, G.M., "Neural Darwinism", pp. 256-69, Basic Books, New York, 1987

[63] de Maine, P.A.D., "Automatic Curve-Fitting I. Test Methods", Computers and Chemistry **2**, 1-6 (1978).

[64] de Maine, P.A.D., G.K. Springer and R.A. Mikelskas, "Automatic Curve-Fitting II. Linear Equations", Computers and Chemistry 2, 7-14 (1978).

[65] de Maine, P.A.D., and M.M. de Maine, "CURFIT Interface Users Manual", Report No. 16 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - 35 pages (July 1989).

[66] de Maine, P.A.D., "Empirical Relationships for Random Self-Avoiding Walks on Lattices", Computer and Chemistry **2**, 53-64 (1978).

[67] de Maine, P.A.D., and M.M. de Maine, "A Computer Tool Kit for Chemists and Engineers I. Regression Analysis (CURFIT System)", Chimicaoggi - In press.

[68] de Maine, P.A.D., M.M. de Maine, B.C. Cartee and W. Chung, "Computer Methods for Experiment Design and Analysis", Report No. 7 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn AL 36849 - 230 pages (1985).

[69] de Maine, P.A.D., "Operation and Logic Manual for the CURFIT System", Report No. 2 of the Series: Automatic Systems for the Physical Sciences, Computer Science Department, The Pennsylvania State University, University Park, PA 16802 - 250 pages (1976).

[70] de Maine, P.A.D., and M.M. de Maine, "A Computer Tool Kit for Chemists and Engineers II. Reaction Modeling (FRANS System)", Chimicaoggi - In press.

[71] Wojtyna, M.S., and P.A.D. de Maine, "User's Manual for the FRANS Interactive User-Friendly Interface", Report No. 15 of the Series: Automatic Systems for the Physical Sciences, Computer Science and Engineering Department, Auburn University, Auburn, AL 36849 - 15 pages (April 1988).

[72] Williams, D.L.H., "Nitrosation", Cambridge University Press, New York (1988).

[73] de Maine, P.A.D., and M.M. de Maine, "Expert System Computer Aids for Chemical Engineers", Presented to the section on Applications of Artificial Intelligence in Chemical Engineering of the American Institute of Chemical Engineers in San Diego, CA on 21/August/1990 - 16 pages.