Mediators in the Architecture of Future Information Systems

Gio Wiederhold

Stanford University

September 1991

An edited version of this report was published in

The IEEE Computer Magazine, March 1992

Abstract

The installation of high-speed networks using optical fiber and high bandwidth messsage forwarding gateways is changing the physical capabilities of information systems. These capabilities must be complemented with corresponding software systems advances to obtain a real benefit. Without smart software we will gain access to more data, but not improve access to the type and quality of information needed for decision making.

To develop the concepts needed for future information systems we model information processing as an interaction of data and knowledge. This model provides criteria for a high-level functional partitioning. These partitions are mapped into information processing modules. The modules are assigned to nodes of the distributed information systems. A central role is assigned to modules that *mediate* between the users' workstations and data resources. Mediators contain the administrative and technical knowledge to create information needed for decision-making. Software which mediates is common today, but the structure, the interfaces, and implementations vary greatly, so that automation of integration is awkward.

By formalizing and implementing mediation we establish a partitioned information systems architecture, which is of managable complexity and can deliver much of the power that technology puts into our reach. The partitions and modules map into the powerful distributed hardware that is becoming available. We refer to the modules that perform these services in a sharable and composable way as *mediators*.

We will present conceptual requirements that must be placed on mediators to assure effective large-scale information systems. The modularity in this architecture is not only a goal, but also enables the goal to be reached, since these systems will need autonomous modules to permit growth and enable them to survive in a rapidly changing world.

The intent of this paper is to provide a conceptual framework for many distinct efforts. The concepts provide a direction for an information processing systems in the foreseeable

future. We also indicate some sub-tasks that are of research concern to us. In the long range the experience gathered by diverse efforts may lead to a new layer of high-level communication standards.

1. Introduction

Computer-based information systems, connected to world-wide high-speed networks provide increasingly rapid access to a wide variety of data resources [Mayo:89]. This technology opens up possibilities of access to data, requiring capabilities for assimilation and analysis which greatly exceed what we now have in hand. Without intelligent processing these advances will have only a minor benefit to the user at a decision-making level. That brave user will be swamped with ill-defined data of unknown origin.

1.1 The problems

We find two types of problems: for single databases a primary hindrance for end-user access is the volume of data that is becoming available, the lack of abstraction, and the need to understand the representation of data; the major concern when combining information from multiple databases the mismatch encountered in information representation and structure. We will expand on those two issues.

Volume and Abstraction

The volume of data can be reduced by selection. It is not coincidental that `SELECT` is the principal operation of relational database management systems, but selected data is still at too fine a level of detail to be useful for decision making. Further reduction is achieved by bringing data to higher levels of abstraction. Aggregation operations as `COUNT, AVERAGE, SD, MAX, MIN`, etc. provide some computational facilities for abstraction, but any such abstraction is formulated within the application using some domain knowledge.

Type of abstraction	Example *Base data*		*Abstraction*
Granularity	`Sales detail`	→	`Product summaries`
Generalization	`Product data`	→	`Product type`
Temporal	`Daily sales`	→	`Seasonally adjusted monthly sales`
Relative	`Product cost`	→	`Inflation adjusted trends`
Exception recognition	`Accounting detail`	→	`Evidence of fraud`
Path computation	`Airline schedules`	→	`Trip duration and cost`

Figure 1. Abstraction functions.

For most base data more than one abstraction must be supported — for the salesmanager the aggregation is by sales region, while for marketing aggregations by customer income are appropriate. Examples of required abstraction types are given in Fig. 1.

Computational requirements for abstraction are often complicated. The groupings to define abstractions may for instance involve recursive closures. These cannot be specified with current database query languages. Application programs are then written by specialists to reduce the data. The use of specific data-processing programs as intermediaries diminishes flexibility and responsiveness for the end user. Now the knowledge that creates the abstractions is hidden and hard to share and reuse.

Mismatch

Data obtained from remote and autonomous sources will often not match in terms of naming, scope, granularity of abstractions, temporal bases, and domain definitions, as listed in Figure 2. The differences shown in the examples must be resolved before automatic processing can join these values.

Type of mismatch	Example *Domains*		
Key difference	`Alan Turing:The Enigma` *reference for reader*	versus	`QA29.T8H63` *reference for librarian*
Scope difference	`employees paid` *includes retirees*	versus	`employees available` *includes consultants*
Abstraction grain	`personal income` *from employment*	versus	`family income` *for taxation*
Temporal basis	`monthly budget` *central office*	versus	`weekly production` *factory*
Domain semantics	`postal codes` *one can cover multiple places*	versus	`town names` *can have multiple codes*
Value semantics	`excessive_pay` *per internal revenue servive*	versus	`excessive_pay` *per board of directors*

Figure 2. Mismatches in data resources.

Without an extended processing paradigm, as proposed in this paper, the information needed to initiate actions will be hidden in ever larger volumes of detail, scrollable on ever larger screens, in ever smaller fonts. In essence, the gap between information and data will yet be wider than it is now. Knowing that information exists, and is accessible creates expectations by end-users. Finding that it is not available in a useful form or that it cannot be combined with other data creates confusion and frustration. We believe that the objection made by some users about computer-based systems, that they create *information overload*, is voiced because those users get too much of the wrong kind of data.

1.2 Use of a model

In order to visualize the requirements we place on future information systems, we consider the activities that are carried out today whenever decisions are being made. The making of informed decisions requires the application of a variety of knowledge to information about the state of the world. To clarify the distinction of data and and knowledge in our model we restate a definition from [Wiederhold:86B].

Data describes specific instances and events. Data may gathered automatically or clerically. The correctness of data can be verified vis-a-vis the real world.
Knowledge describes abstract classes. Each class typically covers many instances. Experts are needed to gather and formalize knowledge. Data can be used to disprove knowledge.

To manage the variety of knowledge, we employ specialists, and to manage the volume of data, we segment our databases. In these partitions partial results are produced, abstracted, and filtered. The problem of making decisions is now reduced to the issue of chosing and evaluating the significance of the pieces of information derived in those partitions and fusing the important portions.

For example, an investment decision for a manufacturer will depend on the fusion of information on its own production capability versus that of others, of its sales experience in related products, of the market for the conceived product at a certain price, of the cost-to-price ratio appropriate for the size of the market, and its cost of the funds to be invested. Specialists will consider these diverse topics, and each specialist will consult multiple data resources to support their claims. The decision maker will fuse that information, using considerations of risk and long-range objectives in combining the results.

An effective architecture for future information systems must support automated information acquisition processes for decision-making activities. By default, we approach the solution in an manner analogous seen in human-based support systems. While we model the system based on abstractions from the world of human information processing, we do not constrain the architecture to be anthropomorphic and to mimic human behavior. There are many aspects of human behavior that we have not yet been able to capture and formalize adequately. Our goal is merely to define an architecture wholly composed of pieces of software that are available or appear to be attainable in a modest timeframe, say ten years. The demands of the software in terms of processing cost should be such that modern hardware can deal with it.

1.3 Current state

We are not starting from a zero base. Systems are becoming available now with the capabilities envisaged by Vannevar Bush for his MEMEX in 1945 [Bush:45]. We can select and scroll information on our workstation displays, we have remote access to data and can present the values on one of multiple windows, we can insert documents into files in our workstation, and we can annotate text and graphics. We can reach conclusions based on this evidence and advise others of decisions made.

Our vision in this paper is intended to carry us beyond, specifically to provide a basis for automated integration of such information. Our current systems, as well as MEMEX, do not adress that phase.

2. A Model of Information Processing

The objective of obtaining information was already clearly stated by Woodward [Woodward:55]. Information enables us to decide among several actions which are otherwise not distinguisable. Consider again a simple business environment. A factory manager needs sales data to set production levels. A sales manager needs demographic information to project future sales. A customer wants price and quality information to make purchase choices.

Most of the information needed by the people in the example can be represented by factual data and should be available on some computer somewhere. Communication networks have the potential to make data available wherever it is needed. However, to make the decisions, a considerable amount of knowledge has to be applied as well. Today, most knowledge is made available through a variety of administrative and technical staff in an institution [Waldrop:84]. Some knowledge is encoded in data-processing programs and expert systems for automated processing.

The process of generating supporting information does not differ much in partially automated or manual systems. In manual systems the decision maker obtains assistance from staff and colleagues who peruse files and prepare summarizations and documentation for their advice. In automated systems the staff is likely to use computers to prepare these documents. The decision-maker rarely uses the computer, because the information from multiple sources is difficult to integrate automatically.

2.1 The processing and the application of knowledge

A technician will know how to select and transfer data from a remote computer to one used for analysis. A data analyst will understand the attributes of the data and define the functions to combine and integrate the data [deMichiel:89]. A statistician may provide procedures to aggregate data on customers into groups that present distinctive behavior patterns. A psychologist may provide classification parameters that characterize the groups. Finally, a manager has to assess the validity of the classifications that have been made, use the information to make a decision, and assume the risk of making the decision. A public relations person may take the information and present it in a manner that can be explained to the stockholders, to whom the risk is eventually distributed. Since these tasks are characterized by using data and knowledge gathered in the past, with the objective of affecting the future, we will refer to these tasks as *planning*. Our definition of planning is broader than that used in Artificial Intelligence research [Cohen:82], although the objectives are the same. To be able to deal with these planning actions in a focused way, we will model the information processing aspects.

There is a recurring activity here:

1 Data are made available. These are either factual observations, or results from prior processing, and combinations thereof.
2 Knowledge is made available. It derives from formal training and experience.
3 Knowledge about the data and its use is applied in two phases:
  i Selection: subsets of available data are
    (1) defined, (2) obtained, and (3) merged.
  ii Reduction: the data found are summarized to an appropriate level of abstraction.
4 Several such results are made available and fused.
5 The combined information is utilized in two ways:
  i Actions are taken that will affect the state of the world.
  ii Unexpected results will be used to augment the experience base of the participants, and others who receive this information, increasing their knowledge
6 The process loops are closed when
  i The actions and their effect is observed and recorded in some database.
  ii The knowledge is recorded to affect subsequent data definition, selection, or fusion.

The two distinct feedback loops and their interaction are illustrated in Figure 3. The data loop closes when the effects of actions taken are recorded in the database. The knowledge loop closes when recently gained knowledge is amde available so it can be used for further selection and data reduction decisions.

The interaction is of prime concern for future systems. Tools for the other steps are easy to identify, we list some in Section 3. We now consider in detail aspects of Step 3 identified above.

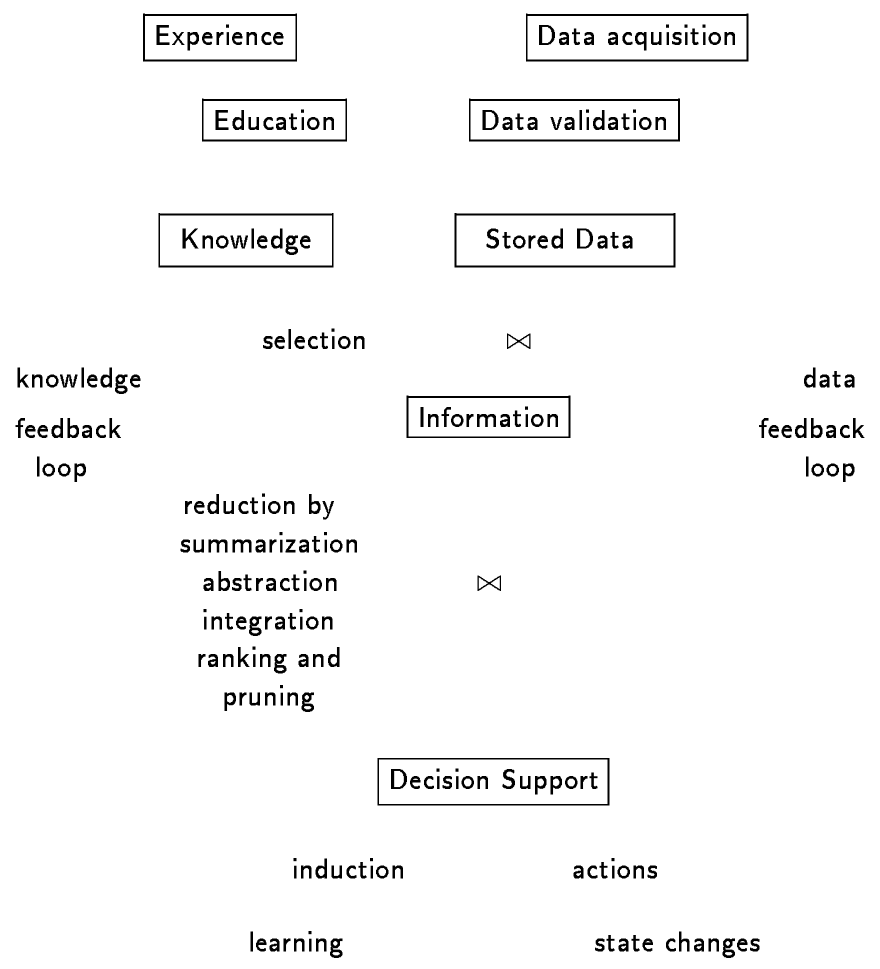

Figure 3. The knowledge and data feedback loops, and their interaction.

2.2 Creation of information

Let us identify where during processing information is created. Since getting information means that something has been obtained that was not known prior to the event, one or more of the following conditions have to hold:

1 The information is obtained from a remote source and was previously not known locally (Step 3.i.2 above). Here the information system must provide communication support.

A special case of this condition is when a database is used for recall, to provide data we knew once, but cannot remember with certainty. Here the database component is used to communicate over time — from the past to the present.

2 Two facts, previously distinct are merged or unified (Step 3.i.3 above). A classic, although trivial, example is finding one's grandfather via transitivity of parents. In

realistic systems, unification of data also involves computation of functions, say the average income and its variation of groups of consumers.

3 Multiple results are fused using pragmatic assessments of the quality and risks associated within Step 4. Here abstractions are combined rather than facts, and the processing techniques are those associated with symbolic processing in rule-based expert systems, although they are also found coded in application programs. In our example the market specialist may want to unify incomes of current consumers with their reading habits to devise an advertising strategy.

Databases record detailed data for each of many instances or events. Reducing this detail to a few abstract cases, raises the information content per element. Of these abstractions only a small, feasible number of justified results is brought to the decision-maker. For instance, the market analyst has made it possible that decisions do not have to deal with individual consumers, but with consumer groups. Certain groups may be unlikely purchasers and are not targeted for promotions.

While the behavior of any individual may not be according to the rules hypothesized in the prior steps, the expected behavior of the aggregate population should be close to the prediction. Failures occur only if we have many errors in the underlying data or serious errors in our knowledge. Uncertainty, however, is common.

2.3 Uncertainty

We cannot predict the future with certainty. For automation of full-scale information systems the processing of uncertainty must be supported, although there are subtasks which can be precisely defined. Uncertainties within a domain may be captured by a formal model. Although we have the argument that all uncertainty computation can be subsumed by probabilistic reasoning [Cheeseman:85] [Horvitz:86], it seems that the variety of assumptions made is based on differences in domain semantics. During analysis uncertain events or states have to be combined for extrapolation into the future. We still have no overall model to predict which uncertainty-combining computations will be best for some domain.

To assess the extent of uncertainty affecting predictions we must combine the uncertain precision of source information, and the uncertainty created at each step where we combine them. In the various steps, we collect observations based on some criterion — say people living in a certain postal-code area. We also have data to associate an income level with that postal-code area, and perhaps even an income distribution. At the same time we may know the income distribution of people buying some product.

It is hence natural to make the conceptual unification step of postal code to potential sales. Unfortunately, there is no logical reason why such a unification should be correct. We have some formal classes — namely people with a certain postal code, and some other formalizable classes based on income; in addition, there are some *natural* classes, not formalized, but intuitively known. In our example some natural classes of interest are the potential purchasers, of which there are several subgroups, namely those that bought in the past, those that will buy today, and those that will be buying the planned products. For the future class only informal criteria can be formulated.

The planner, at the decision-making node will use definable classes — by postal code, by observed and recorded purchasing patterns — to establish candidate members for the natural class of potential consumers. These classes overlap — the better the overlap the more correct the decision-maker's predictions will be. If we infer over classes that do not match well, the uncertainty attached to the generated plans will be greater. But uncertainty is the essence of decision-making and is reflected in the risk that the manager of our example takes on. We would not need a manager with decision-making skills if we only have to report the postal codes for our base group of potential consumers.

2.4 Summary of the information model

Effective information is created at the confluence of knowledge and data. For prediction we use knowledge to conceive rules applicable to natural classes. Selection of data for decision-making is constrained by being based on formal class definitions. Uncertainty is created when formal and natural classes are matched.

Communication of knowledge and data is necessary to achieve this confluence. The communication may occur over space or over time. The information systems we consider must support both communication and fusion of data and knowledge.

Our systems must also be able to deal with continuing change. Both data and knowledge change over time because the world changes and because we learn things about our world. Rules that were valid once eventually become riddled with exceptions, and a specialist who does not adapt will find his work to become without value. An information system architecture must deal explicitly with knowledge maintenance.

3. Information System Components

We will now characterize the components available today to build information systems. All these components are positioned along a *data highway*, provided by modern communication technology. The interactions of the components will be primarily constrained by logical and informational limitations, and not by physical linkages. When we place the components into the conceptual architecture we will recognize lacunae, i.e., places where there are currently no existing, only inadequate, or uncooperative components. We will see where the components must work together. Effective systems can be achieved only if the data and knowledge interfaces of the components are such that cooperation is feasible.

3.1 Data and knowledge resources

There is a wide variety of data resources. We might classify them by a measure of closeness to the source. Raw data obtained from sensors, such as purchase records obtained by point-of-sale scanners, or, on a different scale, images recorded by earth satellites, are at the factual extreme. Census and stock reports contain data that have seen some processing, but will be considered as facts by most users.

At the other extreme are textual representations of knowledge. Books, research reports, and library material, in general, contain knowledge, contributed by the writers and their collegues. Unfortunately, from our processing-oriented view, that knowledge is not exploitable without a human mediator. The text, tables, and figures contained in such documents are data as well, but rarely in a form that can be transcribed for database processing.

If document information is stored in bibliographic systems, such as DIALOG [Summit:67] or MEDLINE [Sewell:87], only selection and presentation operations are enabled. Textual material does not lend itself well to automated analysis, abstraction, and generalization. Some types of reports, produced routinely, tend to have a degree of structure that makes some extent of analysis feasible, as demonstrated for chemical data [Callahan:81], pathology reports [Sager:85], or military event reporting messages [McCune:85].

3.2 Workstation applications

The systems environment for planning activities is provided by the new generations of capable workstations. For planning the users need to interact with their own hypotheses, save intermediate results for comparison of effects, and display the alternate projections over time. This interaction with information establishes the insights needed to gain confidence in one's decisions.

The user exercises creativity at the workstation. We hence do not try to constrain the user here at all. By providing comprehensive support for access to information we hope that the complexity of the end-user's applications can be reduced to such an extent that quite involved analyses remain manageable.

Modern operating and network systems help much with simplifying the users' tasks by removing concern about managing hardware resources. The architecture presented here is meant to address the management of information resources, residing on such hardware.

The base information for planning processes has to be obtained from a variety of data resources. A capable interface is required.

3.3 Mediation

An interface from the users workstation to the database servers which only defines communication protocols and formats in terms of database elements does not deal with the abstraction and representation problems existing in todays' data and knowledge resources. The interfaces must take on an active role.

We will refer to the dynamic interface function as *mediation*. This term includes the processing needed to make the interfaces work, the knowledge structures that drive the transformations needed to transform data to information, and any intermediate storage that is needed.

To clarify the term we will list some examples of mediation found in current information systems. This list could be greatly expanded in length and depth. The citations provide linkages for follow-up; we expect that most readers will have encountered these or similar functions. Types of mediation functions that have been developed are:

1 Transformation and subsetting of databases using view definitions and object templates [Chamberlin:75] [Wiederhold:86] [Lai:88] [Barsalou:88]

2 Methods to access and merge data from multiple databases [Smith:81] [Dayal:83] [Saccá:86]

3 Computations that support abstraction and generalization over underlying data [Hammer:78] [Adiba:81] [Ozsoyoglu:84] [Downs:86] [Wiederhold:87] [deZegher:88] [Cichetti:89] [Chen:89] [Chaudhuri:90]

4 Intelligent directories to information bases, such as library catalogs, indexing aids, and thesaurus structures [Humphrey:87] [Doszkos:86] [McCarthy:88]

5 Methods to deal with uncertainty and missing data because of incomplete or mismatched sources [Callahan:81] [Chiang:82] [Litwin:86] [deMichiel:89]

Many more examples can be added. Most readers will have used or created such interface modules at some time, since we all need information from large, autonomous, and mismatched sources. A major motivation for expert database systems is mediation. We excluded from the list simple processing techniques that do not depend on knowledge that is extraneous to the database proper, such as indexes, caching mechanisms, network services, and file directories.

The examples of mediation shown are specialized, and tied either to a specific database or to a particular application, or both. We will now define *mediators* as modules occupying an explicit, active layer between the users' applications and the data resources. Our goal is a sharable architecture.

4. The Mediator Architecture

In order to provide intelligent and active mediation, we envisage a class of software modules which mediate between the workstation applications and the databases. These *mediators* will form a distinct, middle layer, making the user applications independent of the data resources. What are the transforms needed in such a layer, and what form will the modules supporting this layer have? The responses to these questions are interrelated. We will first identify problems with generalizing the concept of mediation identified in current information systems, and then define some general criteria. We will also justify the necessity of having a modular, domain-specific organization in this layer.

4.1 The Architectural Layers

We create a central layer by distinguishing the function of mediation from the user-oriented processing and from database access. Most user tasks will need multiple, distinct mediators for their subtasks. A mediator uses one or a few databases.

The interfaces that are to be supported provide the cuts where communication network services are needed, as shown in Figure 4.

---

result → decision making

Layer 3 Independent **applications** on workstations — managed by decision makers

- - - - network services to information servers - - - -

Layer 2 Multiple **mediators** — managed by domain specialists

- - - - network services to data servers - - - - -

Layer 1 Multiple **databases** — managed by database administrators

input ← real-world changes

Figure 4: The three layers of this architecture.

Unfortunately, the commonality of function seen in the examples cited in Section 3.3 does not extend to an architectural commonality: the various examples are bound to the data resources and the end-users applications in different ways. It is here where new technology must be established if fusion at the application level is to be supported. Accessing one mediator at a time does not allow for fusion; and seeing multiple results on distinct windows of one screen does not support automation of fusion.

4.2 An inappropriate approach to mediation

The concept of mediation is related to the notion of having corporate *information centers*, as promoted by IBM and others [Atre:86]. These are to be corporate resources, staffed, and equipped with tools to aid any users needing information. The needs and capabilities of an information center, however, differ in two respects from those of automated mediators:

1 A single center, or mediator for that matter, cannot deal with all the varieties of information that is useful for corporate decision-making.

2 Automation of the function will be necessary to achieve acceptable response time and growth of knowledge and its quality over time.

3 The user should not be burdened with the task of seeking out information sources. This task, especially if it is repetetive, is best left to an interaction of application programs in a workstation and automated mediation programs.

The information center notion initiates yet another self-serving bureaucracy within a corporation. Effective staff is not likely to be content in the internal service roles that an information center provides, so that turnover of staff and its knowledge will be high. The only role foreseen in such a center is mediation — bringing together potential users with candidate information.

In order to manage mediation, modularity instead of centralization seems to be essential. Modularity is naturally supported by a distributed environment, the dominant environment of computing in the near future.

4.3 Mediators

Now that we have listed some examples of mediators in use or planned for specific tasks, we can give a general definition.

A **mediator** is a software module that exploits encoded knowledge about some sets or subsets of data to create information for a higher layer of applications.

We place the same requirements on a mediation module that we place on any software module: it should be small and simple [Boehm:84] [Bull:87] so that it can be maintained by one expert or at most by a coherent group of experts.

An important, although perhaps not essential, requirement we'd like to place on mediators is that they be *inspectable* by the potential users. For instance, the rules used by a mediator using expert system technology can be obtained by the user as in any good cooperative expert system [Wick:89]. In this sense the mediators provide data about themselves in response to inspection and such data could be analyzed by yet another, an *inspector* mediator.

Since eventually there will be a great number and variety of mediators, the users have to be able to choose among them. Inspectability enables that task. For instance, distinct mediators which can provide the ten best consultants for database design may use different evaluation criteria; one may use the number of publications and another the number of clients.

Some *meta*-mediators will have to exist that merely provide access to catalogues listing available mediators and data resources. The search may go either way: for a given data source one may want to locate a knowledgeable mediator and for a desirable mediator we need to locate an adequate data resource. It will be essential that the facilities provided by these meta-level mediators can be integrated into the general processing model, since search for information is always an important aspect of information processing. Where search and analyses are separated — as is still common today — for instance, in statistical data-processing, trying to find the data is often the most costly phase of information processing.

For databases that are autonomous, it is desirable that only a limited and recognizable set of mediators depend on anyone of them. Focusing data access through a limited number of views maintained by these mediators provides the data independence which is necessary for databases that are evolving autonomously. Currently, compatibility constraints are hindering growth of databases in terms of structure and scope, since many users are affected. As the number of users and the automation of access increases, the importance of indirect access via mediators will increase.

4.4 The Interface to Mediators

The most critical aspect of this three-layer architecture are the two interfaces that are now created. Today's mediating programs employ a wide variety of interface methods and approaches. The user learns to use one or a few of them, and then remains committed until its performance becomes wholly unacceptable. Unless the mediators are easily and flexibly accessible, the model of common information access we envisage is bound to remain a fiction. It is then in the interface and its support that the research challenge lies. Since our hardware environment implies that mediators can live on any nodes, not only on the workstations and database hosts, their interfaces must be grounded in communication protocols.

The User's Workstation Interface to the Mediators

The range of capabilities that mediators may have is such that a high-level language should evolve to drive the mediators. We are thinking of language concepts here, rather than of interface standards, to indicate the degree of extensibility that must be provided if the mediating concepts are to be generalized.

Determining an effective interface between the workstation application and the mediators will be a major research effort in refining this model. It appears that a *language* is needed to provide flexibility, composability, iteration, and evaluation in this interface. Descriptive, but static interface specifications do not seem to be able to deal with the variety of control and information flow that must be supported. The basic language structure should permit incremental growth so that new functions can be supported as mediators join the network to provide new functionality.

It is important to observe that we do not see a need for a *user-friendly* interface. We need here a machine- and communication-friendly interface. Application programs, executing on the user's workstations, can provide the type of display and manipulation functions appropriate to their types of user. Omitting the criterium of user-friendliness avoids the dichotomy that has lead to inadequacies in SQL, which tries to be user-friendly, while its predominant use will be for programmed access [Stonebraker:88]. Standards needed here can only be defined after experience has been obtained in sharing of these resources to support the high-level functions needed for decision-making.

The Mediator to DBMS Interface

Existing database standards as SQL and RDA provide a basis for database access by mediators. Relational concepts such as selection, views, etc., provide an adequate starting point. A mediator dealing mainly with a specific database need not be constrained to a particular interface protocol, while a mediator which is more general will be effective through a standard interface. A mediator which combines information from multiple databases may use its knowledge to control the merging process and use a relational algebra subset. Joins may, for instance, be replaced by explicit semi-joins, so that intelligent filtering can occur during processing. Still, dealing with multiple sources is likely to lead to incompleteness. Outer-joins are often required for data access to avoid losing objects with incomplete information [Wiederhold:83].

The separation of user applications and databases that the mediating modules provide also allows reorganization of data structures and redistribution of data over the nodes without affecting the functionality of the modules. The three-level architecture then makes an explicit tradeoff in favor of flexibility versus integration. The arguments for this tradeoff focus on the variety of uses made of databases, and by extension, the results of mediator modules [Wiederhold:86].

1 Sharability of information requires that database results can be configured according to one of several views. Mediators, being active, can create objects for a wide variety of orthagonal views [Barsalou:88].

2 On the other hand, making complex objects themselves persistent binds knowledge to the data, which hinders sharability [Maier:89].

3 The loss of performance, due to the interposition of a mediator, can be overcome by techniques listed in Section 5.4.

These arguments do not yet address the distribution of the mediators we envisage.

Available interfaces

We need interface protocols for data and knowledge. For data transmission there are developing standards. The level of concern for mediation is within the application layer in the OSI sense [Tanenbaum:87]. We have faith that communication systems will soon handle all lower level support layers without major problems. The Remote Data Access (RDA) protocol provides one such instance [ISO/RDA:87].

Other technologies that are pushing capabilities at the interface is the National File System, being promoted by [Spector:88] at CMU. However, as mentioned in the introduction, communication of data alone does not guarantee that the data will be correctly understood for processing by the receiver. Differences often exist in the *meaning* assigned to the bits stored, some examples were shown in Figure 2.

4.5 Sharing of mediator modules

In either case, since we are getting access to so much more data, from a variety of sources, arriving at ever higher rates, automated processing will be essential. The processing tasks needed within the mediators are those sketched in the interaction model of Fig. 3: selection, fusion, reduction, abstraction, and generalization. Diverse mediator modules will use these functions in varying extents to provide the support for user applications at the decision making layer above.

The mediator modules will be most effective if they can serve a variety of applications [Hayes-Roth:84]. The applications will compose their tasks as much as possible by acquiring information from the set of available mediators. Unavailable information may motivate the creation of new mediators.

Sharing reinforces the need for two types of partitioning: one, into horizontal layers for end-users, mediators, and databases, respectively, and two, vertically into multiple user applications, each using various configurations of mediators. A mediator in turn will use distinct views over one or several databases. Just as databases are justified by the shared usage they receive, mediators should be sharable. Note that today's expert systems are rarely modular and sharable, so that their development and maintenance cost is harder to amortize.

For instance, the mediation module which can deal with inflation adjustment can be used by many applications. The mediator which understands postal codes and town names can be used by the post office, express delivery services, and corporate mail rooms.

We foresee here an incentive for a variety of specialists to develop powerful, but generally useful mediators, which can be used by multiple customers. Placing one's knowledge into a mediator can be more rapidly effective, and perhaps more rewarding, than writing a book on the topic.

We can now summarize these observations in Figure 5.

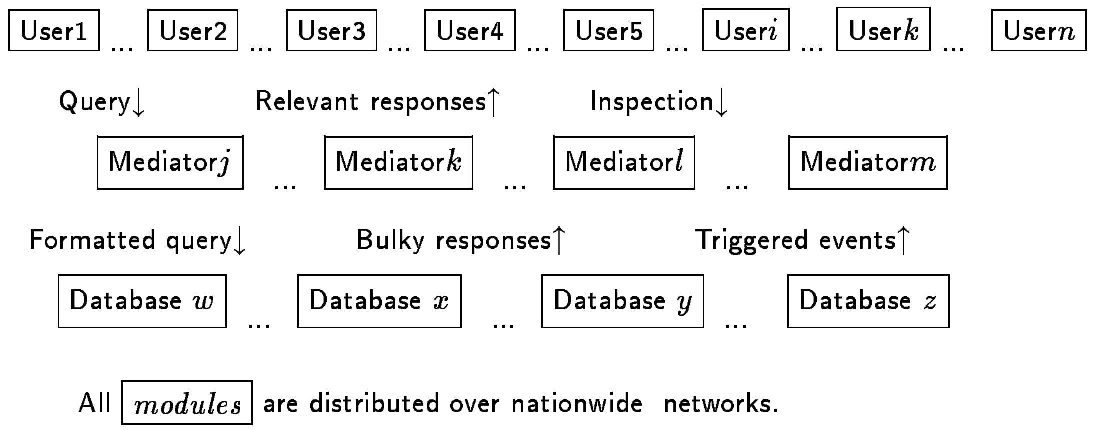

Figure 5. Interfaces for information flow.

4.6 Distribution of Mediators

We have implied throughout that mediators are distinct modules, distributed over the network. Distribution can be motivated by greater economy for access, by better locality for maintenance, and by issues of modularity. For mediators the two latter arguments are the driving force for distribution.

Why should mediators not be attached to the databases? In many cases it may be feasible; in general it is not appropriate.

1 The mediator contains knowledge that is beyond the scope of the database proper. A database programmer, dealing with, say, a factory production control system, cannot be expected to foresee all the strategic uses of the collected information.

2 Concepts of abstraction are not part of database technology today; the focus has been on reliable and consistent management of large volumes of detailed facts.

3 Intelligent processing of data will often involve dealing with uncertainty, adding excessive complexity to database technology.

4 Many mediators will access multiple databases to combine disjointed sets of data prior to analysis and reduction.

Similarily we can argue that the mediators should not be attached to the users' workstation applications. Again, the functions that mediators provide are of a different scope than the tasks being performed on the workstations. Workstation applications may use a variety of mediators to explore the data resources.

A major motivation for keeping mediators distinct is maintenance. During the initial stage of most projects which developed expert systems, their knowledge bases simply grew, and the cost of knowledge acquisition dominated the cost of knowledge maintenance. Many projects, in fact, assumed implicitly that knowledge, once obtained, would remain valid for all times. History teaches us that this is not true; although some fundamental rules may indeed not change for a long time, the application heuristics for its use change as the demands of the world change. Concepts as KNOSs [Tsichritzis:87] recognize the problem, and focus on the problem within a specific domain.

Maintenance by an outside expert of knowledge stored within an application system is intrusive and risky. The end-user should make the decision when new knowledge should be incorporated. We hence keep the mediator modules distinct.

The efficiency concerns of separating knowledge and data can be mitigated by replication. Since mediators (incorporating only knowledge and no factual data) are relatively stable, they can be replicated as needed and placed on nodes along the data highway where they are maximally effective. They certainly should not change during a transaction. As long as the mediators remain small, they can also be easily shipped to a site where substantial data volumes have to be processed.

4.7 Triggers for knowledge maintenance

We have added one aspect of mediation into Figure 5 which has not yet been discussed. Since the knowledge in the mediator must be kept up-to-date, it will be wise for many mediators to place triggers or active demons into the databases [Buneman:79] [Stonebraker:86]. Now the mediators can be informed when the database, and, by extension, the real-world changes. The owner of the mediator should ensure that such changes are in time reflected in the mediator's knowledge base.

We consider, again justified by the analogy to human specialists, that a mediator is fundamentally trusted, but is inspectable when suspicions of obsoleteness arise. For instance, an assumption, say that well-to-do people buy big cars, may be used in the marketing mediator, but it is possible that over time this rule becomes invalid. We expect then that the base data be monitored for changes and that exceptions to database constraints will trigger information flow to the mediator. In a rule-based mediator the certainty factor of some rule can be adjusted [Esculier:89]. If the uncertainty exceeds a threshold, the mediator can advise its creator, the domain expert, to abandon this rule. The end-user need not get involved [Risch 89].

5. Related Research

We have shown throughout that many ofthe individual concepts underlying this architecture have been found in earler work. Thiis is hardly suprising since the problems that future information systems must address exist now, and are being dealt with in many specific situations. Our contribution is a generalization of the practice and a focusing of more general research to these systems. We do want to point out some significant relationships, as well as our own focus.

5.1 Independent actors and agents

There is an obvious corollary between the mediators proposed here and the concept of ACTORS [Hewitt:73]. However, a considerable constraint is imposed on our mediators — namely, they do not interact *intelligently* with each other — so that a hierarchy is imposed for every specific task. The reason for this constraint is to provide computational simplicity and manageability. An actor model can, of course, be used within a mediator to implement its computational task.

The network of connections within the global architecture means that distinct tasks can intersect at nodes within this information processing structure. The same mediator type may access distinct sets of data, and information from one data source can be used by distinct mediators.

5.2 Hierarchical task control

Automation requires an understanding of the control mechanisms that maintain balance and motivate progress or growth. To what extent networks of autonomous agents can be motivated, is unclear.

Rather than extrapolating into the unknown we define an architecture that we can conceptually manage today, and keep our minds open to extensions that are beyond todays conceptual foundations. We take again a cue here from Vannevar Bush, who could identify all units needed for the MEMEX, although its components were based on technology that did not exist in 1945.

Today's organizations depend greatly on hierarchical structures. Many information processing functions, now carried out in organizations, are performed by lower levels to obtain, aggregate, use, and rank information for the decision-making levels of management. Current development of the actors model [Hewitt:88] stresses the necessary match between actors and functions seen in real-world organizations. The ORG model also follows an anthropomorphic paradigm [Malone:87], focusing on a wider set of problem-solving interactions, and provides insights into distributed mediation. In all this research, concepts that model understood organizational practices form a basis, a notion that is broadly argued by [Litwin:89].

The relationships of users to mediators is organizationally similar to that presented in the contract net model by [Smith:80], although the focus of a mediator is on partitioning for management and not on least-cost of computations. Contract net concepts may provide ideas for charging and accounting in this environment, an issue not discussed in this paper, but dealt with in the FAST [Cohen:89] project. The distributed cooperating agents of [Koo:88] deal with non-adversary contracting, a very desirable model for a future world.

5.3 Maintenance and learning

In effective organizations, lower levels of management involved in information-processing also provide feedback to superior layers.

The knowledge embodied in the mediators cannot be assumed to be static. Some knowledge may be updateable by human experts, but for active knowledge domains some automation will be important. Providing advice on inconsistencies between acquired data and assumed knowledge is the first step.

Eventually some mediators will be endowed with learning mechanisms. Feedback for learning may either come from performance measures or from explicit induction over the databases they manage [Blum:82] [Wilkins:87]. The trigger for learning is monitoring. Changes in the database can continuously update hypotheses of interest within the mediator. The validity of these hypotheses can be assessed by inspecting corollary observations in the view of the mediator [DeZegher:88]. The uncertainty of hypotheses can provide a ranking when an application task requests assessments.

5.4 Techniques

Mediators will embody a variety techniques now found both in freestanding applications and in programs that perform mediation functions. These programs are now often classified by the underlying scientific area rather than by its place in information systems.

Techniques from artificial intelligence

The nature of mediators is such that many techniques developed in AI will be employed. We expect that mediators will often use

1 Declarative approaches.

2 Capability for explanation.

3 Heuristic control of inference.

4 Pruning of candidate solutions

5 Evaluating the certainty of results

The literature on these topics is broad [Cohen:84]. Heuristic approaches are likely to be important because of the large solution spaces. Uncertainty computations are needed to deal with missing data and mismatched classes.

Techniques from logical databases

Since the use of mediators encourages a formal approach to the processing of data, techniques being developed within logical and deductive database are likely to be equally important. As long as conventional DBMS do not provide well for recursive search, the computations that achieve closure are likely to be placed into mediators [Beeri:87] [Ramakrishnan:89]. Since proper definition of the rules for stable and finite recursion is likely to remain difficult, these techniques are best managed by experts.

Another example of logical mediation is dealing with generalization, in order to return results of specified cardinality [Chaudhuri:90]. A frequent abstraction is to derive temporal interval represe4ntations from detailed event data. Here we also see a need to attach techniques that depend on domain semantics to the attributes in the database [Jajodia;90].

Techniques for efficient access

In this paper we do not focus on the efficiency of mediated access. We realize that interposing a layer in our information systems is likely to have a significant cost. We will argue that the flexibility and adaptability of a modular approach will overcome in time the inefficiencies induced by an inflexible, rigid structure. The partitioning of the tasks will also make research subtasks more managaable. It is today infeasible to carry out a really significant integrated system development in any single academic institution. Within specialized laboratories substantial systems can be developed and tested, but those are still hard to integrate into the world outside.

We can show some examples of systems research that focuses on overcoming processing bottlenecks:

1 Caching and materialized views and view indexes within the mediators [Roussopoulos:86] [Hanson:87] [Sheth:88]

2 Rule bases for semantic query optimization [King:84] [Chakravarthy:85]

3 data reorganization to follow dominant access demands [Fursin:86]

4 an ability to abandon ineffective object bindings [Gifford:88].

5 Privacy protection for sensitive data through interface modules [Cohen:88]

Sharing of these Techniques

Artificial intelligence, logical, and systems techniques can of course be shared among the mediators. Only knowledge specific to the application needs to be local to each mediator.

In the framework which we present a variety of techniques can cooperate and compete to improve the production of information.

5.5 SoDs

The KBMS Project group at Stanford is in the process of formulating a specific form of mediators, called SoDs [Wiederhold:90]. A SoD provides a declarative structure for the domain semantics, suitable for an interpreter. We see multiple SoDs being used by an application executing a long and interactive transaction. We summarize our concepts here as one research avenue in providing components for the architecture we have presented.

Specific features and constraints imposed on SoDs are:

1 The knowledge should be represented in a declarative form

2 SoDs should have a well-formed language interface

3 They contain feature descriptions exploitable by the interpreter

4 They should be inspectable by the user applications

5 They should be amenable to parallel execution

6 They access the databases through relational views

7 During execution source and derived data objects are bound internally

8 They consider object sharing.

By placing these constraints on SoDs as mediators, we hope to be able to prove aspects of their behavior and interaction. Provable behavior is not only of interest to developers, but also provides a basis for prediction of computational efficiency.

However, the modularity of SoDs causes two types of losses:

1 Loss in power, due to limitations in interconnections

2 Loss in performance, due to reliance on symbolic binding rather than on direct linkages.

We hope to offset these losses through gains obtained by having structures that enable effective computational algorithms. An implementation of a demonstration using frame technology is being expanded. The long-range benefit is of course that small, well-constructed mediators will enable knowledge maintenance and growth.

An Interface Language

Our research identifies the language problem as a major issue. If we cannot express the high-level concepts encapsulated in the mediators well, then we will not be able to implement the required servbices. For application access to SoDs we start from database concepts, where high-level languages have become accepted [WWHC+:89]. The SoD access language (SoDaL) will have the functional capabilities of SQL, plus iteration and test, to provide Turing machine level capability [Qian:89]. New predicates are needed to specify intelligent selection. Selection of desirable objects requires an ability to rank objects according to specified criteria. These criteria are understood by the SoD and are not necessarily predicates on underlying data elements, although for a trivial SoD that may be true. These criteria are associated with result size parameters as 'Give me the 10 best X', where the 'best' predicate is a semantically overloaded term interpreted internally to a particular SoD.

The format of SoDaL is not envisaged to be user-friendly — after all, other subsystems will use the SoDs, not people. It should have a clear and symmetric structure which is machine friendly. It should be easy to build a user-friendly interface, if needed, on top of a capable SoDaL.

6. Limits and Extensions

The separation into layers reduces the flexibility of information transfer. Especially structuring the mediators into a single layer between application and data is overly simplistic. Precursors to general mediators already recognize hierarchies, as the contract net [Smith:80], and general interaction, as actors [Hewitt:73].

A desire to serve large-scale applications is the reason for the simple architecture presented here. To assure effective exploitation of the mediator concepts we propose to introduce complexity within their layer, and the associated processing cost, slowly, only as the foundations are established to permit efficient use.

Structuring mediators into hierarchies should not lead to problems. We already assumed that directory mediators could inspect other mediators. Inspection of lower-level mediators is also straightforward. Low-level mediators may only have database access knowledge, and understand little application domain semantics. On the other hand, high-level mediators can take on minor decision-making functions.

More complex is lateral information sharing among mediators. Some such sharing will be needed to maintain the lexicons that preserve object identity when distinct mediators group and classify data. Optimizers may restructure the information flow, taking into account success or failure with certain objects in one of the involved SoDs.

Fully general interaction among mediators is not likely to be supported at this level of abstraction. Just as human organizations are willing to structure and constrain interactions, even at some lost-opportunity cost, we impose similar constraints on the broad information systems we envisage.

Learning by modifying certainty parameters in the knowledge-base is relatively simple. Learning of new concepts is much more difficult, since we have no mechanisms that relate observations automatically to unspecified symbolic concepts. By depending initially fully on the human expert to maintain the mediator, then moving to some parameterization of rule priorities, we can gradually move to automated learning.

Efficiency is always a concern. Once derived data are available the need to analyze large databases is reduced, but the intermediate knowledge has to be maintained. Binding of the knowledge to provide the effect of compilation of queries is an important strategy. Work in rule-based optimizers and automatic creation of expert systems points in that direction [Schoen:88]. These tactics exacerbate the problems of maintaining integrity under concurrent use. Research into truth-maintenance is relevant here [Filman:88] [Kanth:88].

Requirements of data security may impose further constraints. Dealing with trusted mediators however, may encourage database owners to participate in information sharing to a greater extent than they would if all participants would need to be granted file-level access privileges.

7. Summary

We envisage a variety of information processing modules residing in nodes along the data highways that advances in communication technology can now provide. A conceptual layering distinguishes nodes by function: decision-making exploration, information support by mediation of data by knowledge, and base data resources.

The mediation function, now seen in a variety of programs, is placed into explicit mediation modules, or *mediators*. For clarity we place all the mediators into one horizontal layer. These mediators are to be limited in scope and size to enable maintenance by an expert as well as inspection and selection by the end-user. Mediators are associated with the domain expert, but may be replicated and shipped to other network nodes to increase their effectiveness. Specialization increases the power and maintainability of the mediators and provides choices for the users.

Applications obtain information by dealing with abstractions supported by the mediators, and not by accessing base data directly. A language will be needed to provide flexibility in the interaction between the end-users' workstation and the mediators. We discuss the partitioning of artificial intelligence paradigms into pragmatics (at the user-workstation layer) and the formal infrastructure (in the mediation layer) further in [Wiederhold:90].

For query operations the control flow goes from the application to the mediator, who would interpret the query to plan optimal database access. The data would flow to the mediator, be aggregated, reduced, pruned, etc., and the results reported to the query originator. Multiple mediators serve an application with pieces of information from their subdomains.

The knowledge-based paradigms inherent in intelligent mediators indicate the critical role of artificial intelligence technology foreseen when implementing mediators. Knowledge sources of the mediation examples listed in Section 3 range from type information to business rules. The mediating modules we are developing, SoDs, stress structure and and declarative domain semantics for interpretation and inspection.

Mediators may be strengthened by having learning capability. Derived information may simply be stored in a mediator. Learning can also lead to new tactics of data acquisition and control of processing.

The intent of the architectural model is not to be exclusive and rigid. It is intended to provide a common framework under which many new technologies can be accommodated. As shown throughout, many existing concepts can be viewed as implementations of mediation. Especially the techniques listed in Section 5.4 have validity within and outside of this framework, but will become more accessible and sharable. Now they are at best enbedded within intelligent application programs.

An important objective of the architecture is the ability to utilize a variety of information sources without demanding that they be brought into a common format and with only minimal requirements on their interfaces. Maintenance of the knowledge bases in the mediators requires specialization and a manageable size.

In a recent report the three primary issues to be addressed in knowledge-based systems were maintenance, problem modeling, and learning and knowledge acquisition [Buchanan+:89]. The architecture we presented here contributes to all three issues, largely by providing a partitioning that permits large systems to be composed from modules that are maintainable, that can implement specific submodels, and that access domain data for learning and knowledge validation.

8. Acknowledgement

This paper integrates and extends concepts developed during research into the management of large knowledge bases, primarily supported by DARPA under contract N39-84-C-211. Useful insights were gathered by interaction with researchers at DEC (project title 'Reasoning about R1ME') and with the national HIV-modeling community [Cohen:88]. The DARPA Principal Investigators meeting (Nov.88, Dallas TX) helped with solidifying these concepts, in part by providing a modern view of the communication and processing capabilities that lie in our future. Robert Kahn of the Corp. for National Research Initiatives encouraged development of these ideas. Further inputs were provided by panel participants at the DASFAA 1 (Seoul, April 1989), VLDB 15 (Amsterdam, August 1989) and IFIP 11 (San Francisco, August 1989) conferences. Research work on triggers is being carried out by Tore Risch at the Hewlett-Packard Stanford Science Center and his input helped clarify salient points. Andreas Paepke of HP commented as well. Dr. Witold Litwin of INRIA and the students on the Stanford KBMS project, especially Surajit Chaudhuri, provided a critical review and helpful comments.

References

[Adiba:81] Michel E. Adiba: "Derived Relations: A Unified Mechanism for Views, Snapshots and Distributed Data"; *VLDB* 7, Zaniolo and Delobel(eds), Sep.1981, pp.293–305.

[Atre:86] Shaku Atre: *Information Center: Strategies and Case Studies*, Vol. 1; Atre Int. Consultants, Rye NY, 1986.

[Barsalou:88] Thierry Barsalou: "An Object-based Architecture for Biomedical Expert Database Systems"; *SCAMC* 12, IEEE CS Press, Washington DC, November 1988.

[Basu:88] Amit Basu: "Knowledge Views in Multiuser Knowledge Based Systems"; *IEEE Data Engineering Conference* 4, Feb.1988, Los Angeles.

[Beeri:87] C. Beeri and R. Ramakishnan: "On the Power of Magic"; *ACM-PODS*, San Diego, Mar.1987.

[Blum:82] Robert L. Blum: *Discovery and Representation of Causal Relationships from a Large Time-Oriented Clinical Database: The RX Project*; Springer Verlag, Lecture Notes in Medical Informatics, no.19, 1982.

[Boehm:84] Barry W. Boehm: "Software Engineering Economics"; *IEEE Trans. Software Eng.*, Vol.10 No.1, Jan.1984, pp.4–21.

[Bull:87] M. Bull, R. Duda, D. Port, and J. Reiter: "Applying Software Engineering Principles to Knowledge-Base Development"; *Proc. Expert Systems and Business*

87, NY, Learned Information, Meadford NJ, Nov.1987, pp.27-37.

[Buchanan+:89] B.G. Buchanan, D. Bobrow, R. Davis, J. McDermott, and E.H. Shortliffe: "Research Directions in Knowledge-Based Systems"; Stanford KSL report 89-71, to appear in J. Traub (ed.): Annual Review of Computer Science.

[Buneman:79] O.P. Buneman and E.K. Clemons: "Efficiently Monitoring Relational Databases"; *ACM Trans. on Database Systems*, Vol.4 No.3, Sep.1979, pp.368–382.

[Bush:45] Vannevar Bush: "As We May Think"; *Atlantic Monthly*, Vol.176 No.1, 1945, pp.101–108.

[Callahan:81] M.V. Callahan and P.F. Rusch: "Online implementation of the CA SEARCH file and the CAS Registry Nomenclature File"; *Online Rev.*, Vol.5 No.5, Oct.1981, pp.377-393.

[Chamberlin:75] D.D. Chamberlin, J.N. Gray, and I.L. Traiger: "Views, Authorization, and Locking in a Relational Data Base System"; *Proc.1975 NCC*, AFIPS Vol.44, AFIPS Press, pp.425–430.

[Chakravarthy:85] U.S. Chakravarthy, D. Fishmann and J. Minker; "Semantic Query Optimization in Expert Systems and Database Systems"; *Expert Databases*, Kerschberg(ed), Benjamin Cummins, 1985.

[Chaudhuri:90] S. Chaudhuri: "Generalization and a Framework for Query Modification"; *IEEE Data Engineering* 6, Los Angeles, Feb. 1990.

[Cheeseman:85] Peter Cheeseman: "In Defense of Probability"; *Proc. IJCAI*, Los Angeles, 1985, pp.1002–1009.

[Chen:89] M.C. Chen and L. McNamee: "A Data Model and Access Method for Summary Data Management"; *IEEE Data Engineering Conf.* 5, Los Angeles, Feb.1989.

[Chiang:82] T.C. Chiang and G.R. Rose: "Design and Implementation of a Production Database Management System (DBM-2)"; *Bell System Technical Journal*, Vol.61 No.9, Nov.1982, pp.2511–2528.

[Cicchetti:89] R. Cicchetti, L.D. Lakhal, N. LeThanh, and S. Miranda: "A Logical Summary-data Model for Macro Statistical Databases"; *DASFAA* 1, Seoul Korea, KISS and IPSJ, Apr.1989, pp.43–51.

[Cohen:82] P.R. Cohen and E. Feigenbaum (eds.): *The Handbook of Artificial Intelligence*; Morgan Kaufman, 1982.

[Cohen:88] H. Cohen and S. Layne (editors) *Future Data Management and Access, Workshop to Develop Recommendations for the National Scientific Effort on AIDS Modeling and Epidemiology*; sponsored by the White House Domestic Policy Council, 1988.

[Dayal:83] U. Dayal and H.Y. Hwang: "View Definition and Generalization for Database Integration in Multibase: A System for Heterogeneous Databases"; *IEEE Trans. Software Eng.*, Vol.SE-10 No.6, Nov.1983, pp.628–645.

[DeMichiel:89] Linda DeMichiel: "Performing Operations over Mismatched Domains"; *IEEE Data Engineering Conference* 5, Feb.1989; *IEEE Transactions on Knowledge and Data Engineering*, Vol.1 No.4, Dec. 1989.

[DeZegher:88] I. DeZegher-Geets., A.G. Freeman, M.G. Walker, R.L. Blum and G. Wiederhold: "Summarization and Display of On-line Medical Records"; *M.D. Computing*, Vol.5 no.3, March 1988, pp.38-46.

[Downs:86] S.M. Downs, M.G. Walker, and R.L. Blum: "Automated summarization of on-line medical records"; IFIP *Medinfo'86*, North-Holland 1986, pp.800-804.

[Doszkocs:86] Tamas E. Doszkocs: "Natural Language Processing in Information Retrieval"; *J.Am.Soc.Inf.Sci.*, Vol.37 No.4, Jul.1986, pp.191–196.

[Esculier:89] Christian Esculier: *Introduction a la Tolerance Semantique*; PhD thesis, Un. Joseph Fournier, Grenoble, 1989.

[Filman:88] Robert E. Filman: "Reasoning with Worlds and Truth Maintenance in a Knowledge-Based Programming Environments"; *Comm. ACM*, Vol.31 No.4, Apr.1988, pp. 382–401.

[Fursin:86] Gennadiy I. Fursin: "Estimating and Decision Making Techniques in Database Design"; *Control Systems and Machines*, Kiev, No.1, Jan.1986, pp.141–155.

[Gifford:88] D.K. Gifford, R.M. Needham, and M.D. Schroeder: "The CEDAR File System"; *Comm. ACM*, Vol.31 no.3, Mar.1988, pp.288–298.

[Gray:86] Jim N. Gray: "An Approach to Decentralized Computer Systems"; *IEEE Trans. Software Eng.*, Vol.Se-12 No.6, Jun.1986, pp.684–692.

[Hammer:78] M. Hammer and D. McLeod: "The Semantic Data Model: A Modelling Machanism for Data Base Applications"; *Proc.. ACM SIGMOD* 78, Lowenthal and Dale(eds), 1978, pp.26–36.

[Hanson:87] Eric Hanson: "A Performance Analysis of View Materialization Strategies"; *Proc. ACM-SIGMOD* 87, May 1987.

[Hayes-Roth:84] Frederick Hayes-Roth: "The Knowledge-based Expert System, A tutorial"; *IEEE Computer*, Sep.1984, pp.11-28.

[Hewitt:73] C. Hewitt, P. Bishop, and R. Steiger: "A Universal Modular ACTOR Formalism for Artificial Intelligence"; *IJCAI* 3, SRI, Aug.1973, pp.235-245.

[Hewitt:88] Carl Hewitt: Knowledge Processing Organizations proposal; Nov.1988.

[Horvitz:86] E.J. Horvitz, D.E. Heckerman, and C. Langlotz: "A Framework for Comparing Alternate Formalisms for Plausible Reasoning"; *Proc. AAAI*-86, 1986, pp.210–214.

[Humphrey:87] S.M. Humphrey, A. Kapoor, D. Mendez, and M. Dorsey: "The Indexing Aid Project: Knowledge-based Indexing of the Medical Literature"; NLM, LH-NCBC 87-1, Mar.1987.

[ISO/RDA:87] Working draft of ISO Remote Access Protocol; ISO/TC97/SC21 N 1926 (ANSI X3H2-87-210), Jul.1987.

[Kanth:88] M.R. Kanth and P.K. Bose: "Extending an Assumption-based Truth Maintenance System to Databases"; *IEEE CS Data Engineering Conference* 4, Feb.1988, LosAngeles.

[Kaplan:84] S.Jerrold Kaplan: "Designing a Portable Natural Language Database Query System"; ACM TODS, Vol.9 No.1, Mar.1984, pp.1–19.

[King:84] Jonathan J. King: *Query Optimization by Semantic Reasoning*; Univ.of Michigan Press, 1984.

[Koo:88] C.C. Koo and G. Wiederhold: "A Commitment-based Communication Model for Distributed Office Environments"; *ACM-COIS*, Mar.1988, pp.291-298.

[Lai:88] K-Y. Lai, T.W. Malone, and K-C. Yu: "Object Lens: A Spreadsheet for Cooperative Work"; *ACM Ttans. on Office Inf. Systems*, Vol.6 No.4, Oct.1988, pp.332–353.

[Litwin:86] W. Litwin and A. Abdellatif: "Multidatabase Interoperability"; *IEEE Computer*, Vol.19 No.12, Dec.1986, pp.10–18.

[Litwin:89] W. Litwin and N. Roussopolous: "A Model for Computer Life"; University of Maryland, Institute for Advanced Computer Studies, UMIACS-TR-89-76, July 1989.

[Maier:89] David Maier: "Why isn't there an Object-oriented Data Model"; *Information Processing* 89, Ritter (ed), IFIPS North-Holland 1989, pp.793-798.

[Malone:87] T.W. Malone, K.R. Grant, F.A. Turbak, S.A. Brobst, and M.D. Cohen: "Intelligent Information-Sharing Systems"; *Comm. ACM*, Vol.30 No.5, May.1987, pp.390–402.

[Mayo:89] J.S. Mayo and W.B. Marx, jr.: "Introduction: 'Technology of Future Networks"'; *AT&T Technical Journal*, Vol.68 No.2, Mar.1989.

[McCarthy:88] John L. McCarthy: "Knowledge Engineering or Engineering Information: Do We Need New Tools?"; *IEEE Data Engineering Conference* 4, Feb.1988, Los Angeles.

[McCune:85] B.P. McCune, R.M. Tong, J.S. Dean, and D.G. Shapiro: "RUBRIC: A System for Rule-based Information Retrieval"; *IEEE Trans. Software Eng.*, Vol.SE-

11 no,9, Sep.1985, pp.939-945.

[McIntyre:87] S.C. McIntyre and L.F. Higgins: "Knowledge Base Partitioning For Local Expertise: Experience In A Knowledge Based Marketing DSS "; *Hawaii Conf. on Inf, Systems* 20, Feb.1987.

[Ozsoyoglu:84] Z.M. Ozsoyoglu and G. Ozsoyoglu: "Summary-Table-By-Example: A Database Query Language for Manipulating Summary Data"; *IEEE Data Engineering Conf.* 1, Los Angeles, Apr.1984.

[Qian:89] XiaoLei Qian: "The Deductive Synthesis of Iterative Transaction"; PhD thesis, Stanford University, June 1989.

[Ramakrishnan:89] Raghu Ramakrishnan; "Conlog: Logic + Control"; Un. Wisconsin-Madison, CSD, 1989.

[Risch:89] Tore Risch: "Monitoring Database Objects"; *Proc. VLDB* 15, Amsterdam, Aug. 1989, Morgan Kaufmann Pubs.

[Roussopoulos:86] N. Roussopoulos and H. Kang: "Principles and Techniques in the Design of ADMS"; *IEEE Computer*, Vol.19 No.12, Dec.1986 pp.19–25.

[Saccà:86] D. Saccà, D. Vermeir, A. d'Atri, A. Liso, S.G. Pedersen, J.J. Snijders, and N. Spyratos: "Description of the Overall Architecture of the KIWI System"; *ESPRIT'85*, EEC, Elseviers, 1986, pp. 685–700.

[Sager:85] N. Sager, E.C. Chi, C. Friedman, and M.S. Lyman: "Modelling Natural Language Data for Automatic Creation of a Database from Free-Text Input"; *IEEE Database Engineering Bull.*, Vol.8,No.3, Sep.1985, pp.45-55.

[Schoen:88] E. Schoen, R.G. Smith, and B.G. Buchanan: "Design of Knowledge-based Systems with a Knowledge-based Assistant"; *IEEE Trans. Software Eng.*, Vol.14 No.12, Dec.1988, pp.1771–1791.

[Sewell:86] W. Sewell and S. Tietelbaum: "Observations of End-user Online Searching Behavior over Eleven Years"; *J.Am.Soc.Inf.Sci.*, Vol.37 No.4, Jul.1986, pp.234–245.

[Shen:85] Sheldon Shen: "Design of a Virtual Database"; *Information Systems.*, Vol.10 No.1, 1985, pp.27–35. Rj. Query processing involves building a query tree with

[Sheth:88] A.P. Sheth, J.A. Larson, and A. Cornellio: "A Tool for Integrating Conceptual Schemas and User Views"; *IEEE Data Engineering Conference* 4, Feb.1988, Los Angeles.

[Smith:80] R.G. Smith: "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver"; *IEEE Ttans. Computers*, Vol.C-29 No.12, Dec.1980, pp.1104–1113.

[Smith:81] J.M. Smith et al: "MULTIBASE — Integrating Heterogeneous Distributed

Database Systems"; *Proc.NCC*, AFIPS Vol.50, Mar.1981, pp.487–499.

[Stonebraker:85] M. Stonebraker, D. DuBourdieux, and W. Edwards: "Triggers and Inference in Database Systems"; Brodie, Mylopoulos, and Schmidt (eds) *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies*, Springer, Feb.1986, pp.297–314.

[Stonebraker:88] Michael Stonebraker: "Future Trends In Database Systems"; *IEEE Data Engineering Conf.* 4, Feb.1988, Los Angeles.

[Summit:67] R.K. Summit: "DIALOG: An Operational On-Line Reference Retrieval System"; *ACM Nat. Conf.* 22, 1967, pp.51–56.

[Tanenbaum:88] Andrew S. Tanenbaum: *Computer Networks*, 2nd ed; Prentice-Hall, 1988.

[Tsichritzis:87] D. Tsichritzis, E. Fiume, S. Gibbs, and O. Nierstrasz: "KNOs: Knowledge Acquisition, Dissemination and Manipulation Objects"; *ACM Trans. on Office Inf. Sys.*, Vol.5 No.1, Jan.1987, pp.96-112.

[Waldrop:84] M.Mitchell Waldrop: "The Intelligence of Organizations"; *Science*, Vol.225 No.4667, Sep.1984, pp.1136–1137.

[Wetherbe:85] J.C. Wetherbe and R.L. Leitheiser: "Information Centers: A Survey of Services, Decisions, Problems, and Successes"; *Inf. Sys. Manag.*, Vol.2 No.3, 1985, pp.3–10.

[Wick:89] M.R. Wick and J.R. Slagle: "An Explanation Facility for Today's Expert Systems"; *IEEE Expert*, Spring 1989, pp .26–36.

[Wiederhold:83] Gio Wiederhold: *Database Design*; McGraw-Hill, 1983.

[Wiederhold:86B] Gio, Wiederhold: "Knowledge versus Data"; Chapter 7 of Brodie, Mylopoulos, and Schmidt (eds.) 'On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies' Springer Verlag, June 1986, pages 77–82.

[Wiederhold:86] Gio Wiederhold: "Views, Objects, and Databases"; *IEEE Computer*, Vol.19 No.12, December 1986, Pages 37–44.

[Wiederhold:87] G. Wiederhold and X-L. Qian: "Modeling Asynchrony in Distributed Databases"; *IEEE Data Engineering Conference* 3, Los Angeles, Feb. 1987.

[Wiederhold:90] G. Wiederhold, P. Rathmann, T. Barsalou, B-S. Lee, and D. Quass: "Partitioning and Combining Knowledge"; *Information Systems*, to appear 1990.

[Woodward:55] Woodward, Philip M. : *Probability and Information Theory, with Applications to Radar*; McGraw-hill 1955; Pergamon; 1964.

[WWBD:86] G.C.M. Wiederhold, M.G. Walker, R.L. Blum, and S.M. Downs: "Acquisition of Knowledge from Data"; *Proc. ACM SIGART ISMIS*, Oct.1986, pp.74–84.

[WWHC+:89] G.CM Wiederhold, M.G. Walker, W. Hasan, S. Chaudhuri, A. Swami, S.K. Cha, X-L. Qian, M. Winslett, L. DeMichiel, and P.K. Rathmann: "KSYS: An Architecture for Integrating Databases and Knowledge Bases"; in Amar Gupta (ed.), *Heterogenous Integrated Information Systems*, IEEE Press, 1989.

[Wilkins:87] D.C. Wilkins, W.J. Clancey, and B.G. Buchanan: "Knowledge Base Refinement by Monitoring Abstract Control Knowledge"; Stanford, TR. STAN-CS-87-1182, Aug.1987.