Interoperation, Mediation, and Ontologies

Gio Wiederhold
Stanford University
9 November 1994

Published in the Proceedings International Symposium on Fifth Generation Computer Systems (FGCS94), Workshop on Heterogeneous Cooperative Knowledge-Bases, Vol.W3, pages 33-48, ICOT, Tokyo, Japan, Dec. 1994.

Abstract
In this paper we address the problem of interoperation at a semantic level. We assume that emerging standards will solve most of the syntactic infrastructure problems that exist today. However, interoperation has to deal with sources from differing domains, and their semantic differences.

We present mediation as the principal means to resolve problems of semantic interoperation. Since the number of domains is large we cannot foresee a global solution to that problem, and hence will have to deal with domain interaction incrementally, and bring domains together as and when needed. To formalize the processing in mediation we will need formal models of the source domains, of the articulation points, and of the customer needs. We expect to capture these models using tools developed in the ARPA knowledge-sharing projects. The basis for these models are domain ontologies. We propose a knowledge-based algebra to manage the interoperation of information from different domains. Several research issues that arise in the formalization of semantic interoperation are indicated.

1. Introduction

Several centuries ago, few people traveled. Most worked on a farm or pursued a trade at home, and occasionally walked to the local market. Soldiers marched long distances, mariners sailed, knights rode horses, and only some adventurous merchants and the crusaders used multiple means of transport, as camels, horses, and ships. Today, traveling long distances routinely requires a variety of conveyances, each suited for its particular domain, say buses, trains, planes, ships, barges, bicycles, cars, trucks, and the like. Vehicles that have to operate in multiple domains, for example, amphibious carriers, tend to be less efficient in each mode than domain-specific vehicles. Even similar vehicles become attached to specific domains; once a tanker truck has carried gasoline, it is no longer a desirable vehicle for transporting milk. Specialization may also be needed to deal with quantitative differences. An old car may well be adequate to go shopping, but using it may be too risky for delivering supplies to customers. Driving a personal car cross-country is possible, but rarely feasible in business; for long distances air travel dominates. To switch among transport modes, interchange hubs are established, where people can wait, obtain tickets, and where goods are repackaged to suit the means of transport and the consumer. Many businesses spring up around the hubs, taking advantage of the traffic and adding value to the goods passing through.

On the information highways, we are encountering diversity of roads and vehicles as well. The diversity becomes a greater concern as we reach out beyond our homes and workplaces. While once computers operated in stand-alone mode, or used simple connections to each other, the information highways we are contemplating are likely to have millions of participating computers, and thousands of interchange points. While the notion of having similar computers, and similar data and information structures everywhere would make life simple, such a coherence is clearly not feasible, just as we could not have a single mode of transport nor a single type of container for all the goods to be shipped. Progress also requires change, and incremental changes also create inconsistencies. While Henry Ford might have been content if we all stayed with the Model T, over time most Model T's were replaced by a variety of faster, bigger, and more colorful vehicles.

2. Mediators

The software equivalent of interchange hubs are called *mediators*. We will summarize here the essential aspects of mediation, an activity which reduces data to information by applying knowledge about resources, search strategies, and user requirements [Wiederhold:92C]. Mediation is an integrating concept, combining a number of current technologies to find and transform data, and making the resulting information available at hubs along the information highways. Mediation recognizes the autonomy and diversity of the data systems and information services that support the hubs, and the user applications that utilize them. The autonomy of the participants enables the overall system to grow, since new sources, new means of transport, and novel information processes can be inserted. Incremental growth only requires that a few mediating hubs be adapted to link the new facilities into the traffic network. As the new facilities become more popular, further mediators will adapt to take advantage of them and the business they represent. Those users that need the new sources will use the adapted mediators; users that don't care remain unaffected.

Slide 3 goes here

Figure 1. Transforming Data to Information

Value-added services in a mediator include combinations of:

1 Focused access to a variety of resources [Gravano:94]
2 Selection of relevant source material [Salton:90]
3 Resolution of scope mismatches [Wiederhold:92I]
4 Abstraction to bring material to matching levels of granularity
5 Integration of material from diverse source domains
6 Assessment of quality of material from diverse sources
7 Ranking in terms of quality or relevance of material from diverse sources
8 Omission of replicated or known information
9 Seeking exceptions from expected values or trends
10 Transformation of material to make presentation effective for the customer
11 Adaptation to the bandwidth and media capabilities of the customer
12 Optimization to provide small response times or low cost

This list implies a wide range of software technology. Remote, specialized services may be invoked by a mediator, just as transportation hubs encourage specialized factories to provide value-added services. Sometimes intensive processing is needed to abstract and merge data. Such processing may be performed at yet other nodes in the network, including high-performance computers.

The implementation of mediators varies greatly. If knowledge-based processing is crucial, we may find mediators programmed in languages as LISP. If optimization is crucial to processing, the mediators may depend on packages written in FORTRAN. Maintenance would be enhanced by using declarative approaches that could be understood and modified by end-users. Many of our current mediators have been coded in the C language.

Projects that are using mediators today include manufacturing systems, as design of portions of a new fighter aircraft at Lockheed Aeronautical Systems and gimbals for antenna positioning on spacecraft at Lockheed Space Systems. Access to data for healthcare services, managed at the University of Texas in Arlington, is slated to use mediator technology. Early applications have been in the integration of information for flexible military command systems. The technology for these projects is supplied by a variety of vendors, ISX corporation serves as a contact point for these projects [ISX:94].

Since we assume that most mediation services are performed in autonomous computing nodes, the requirement for interoperation is the ability to communicate according to some standard conventions [GK:94]. To enable a greater variety of communication actions, participants, and representations a *Knowledge Query and Manipulation Language* (KQML) has been developed [FFMM:94]. Heterogeneity of computing platforms, operating systems, and message passing infrastructures is being overcome by concerted efforts in many communities. For mediation we must also consider the terminology and representation conventions.

2.1 Domain-specificity

Just as hubs for transport have become specialized to deal with people, mail, goods, foodstuff as fruit, fish, beef, and the like, mediators will also be specialized. Specialization makes maintenance feasible; an expert can focus on one's own domain, without having to consider the different constraints imposed by handling unrelated domains, say, fish versus bicycles [Haddock:94]. Differing domains may be best served by different programming paradigms, say finance versus engineering. Other subsets of domains may use similar technology, but differ in the concepts and structure of their knowledge representations, say electronic versus civil engineering. Most of this paper will deal with the latter issue, namely the management of diverse *ontologies*.

Slide 7 goes here

Figure 2. Some Features of Mediators

2.2 Distribution

Mediation is achieved by software. A mediator transforms data available on the network to make it more suitable and relevant to the consumer. This software function can be carried out on the computer where the mediation was developed, or can be assigned to other computers on the network. Since software is easy to copy over the digital highways and mediators are not directly bound to large data files, mediators can be rapidly replicated. Software is typically much smaller than the data it processes, so that it is easy to install at any location where compatible computers are available. The ease with which software-based factories can change location implies a much more flexible configuration of the network than seen in traditional manufacturing. A supplier who fails to deliver sufficient added value can be rapidly replaced.

Replication can also enhance the technical performance of mediators. By locating the mediator near the data source communication load can be reduced, since the resulting information is typically smaller. If the processing algorithms used in mediation are costly, the software can be moved to a high performance processor on the network, again improving response time.

If demand for a particular type of mediator is strong, replicates can be distributed to additional sites in the network. Since now the communication links will shorten, response time is enhanced as well.

2.3 Caching

At a hub one often finds hotels for people and warehouses for goods. These are necessary because the variety of transport mechanisms cannot be perfectly synchronized, and some temporary storage is needed. The equivalent function in the digital systems is intermediate storage and caching of data that may be of interest to later requests. In a mediator only selected and processed information would be cached, greatly reducing the storage requirement for caching. Any stored information must be well identified with name, type, source, date-of-validity, and mediation performed, so that it can be appropriatly used. If source data has to be cached it may be wise just to keep a reference to the source in the mediator, and acquire it from the source databases when required [Roussopoulos:91].

2.4 Maintenance

Mediation adds value to the data by applying the knowledge of the expert who has created the mediator. Mediators should also be maintained by those experts, so that the mediators remain effective in a constantly changing world. As soon as an improved mediator is developed it can be advertised over the network, both to existing subscribers as well as to potential new clients. A poorly maintained mediator will lose value over time, and be a candidate for replacement by a competitor.

Existing users can continue to use the old mediator version, and not be disturbed until they decide that their application needs the upgrade. This flexibility is crucial, since now upgrades are not constrained by the effect on the existing community of users. The maintainer will, of course, try to keep the number of versions of a mediator service modest. The charges for old mediators may increase, to encourage applications that depend on old versions to upgrade.

2.5 Economics

We assume an economic model for mediation. Unless services are reimbursable, it is unlikely that they will be maintained for very long [SDKLPSS:94]. The traditional method for obtaining software has been purchase, and those purchases typically embody restrictions on replication and secondary distribution, closely following the copyright protection model [Wiederhold:94L].

More satisfactory approaches to revenue collection are being developed today [Cox:94]. The user of a mediator should be able to pay for services received incrementally, i.e., with each use. Incremental charges encourage trial use, since the financial investment will be small. Since a mediator can live remotely from the user's application workstation, the charge mechanism is isolated from the user. Use charges are easily collected in such an architecture, and fraud, common today in software acquisition, is much more difficult.

The creator and maintainer can then receive payments proportionally to the use the mediator receives. Since the fee per use will be small, it is essential that a direct and automatic electronic payment mechanism be used.

3. Methods

The concept of mediation encompasses a large number of techniques that have been dealing with segments of the problems encountered when developing or composing large information systems. We will enumerate a few, since this will clarify the use of mediation.

3.1 Access paths

Browsing, or *netsurfing*, using the World-Wide-Web and MOSAIC tools is a common information access technology today. The knowledge about access paths is embedded in referencing linkages. A number of subsidiary services have sprung up that collect such linkages periodically, as: HARVEST, ALIWEB, NETCRAWLER, and FISH. The linkages are placed by subject into temporary databases, and represent a cached knowledge-base. Some systems also hold the actual text, or at least their abstracts, to reduce the work for successive retrievals. Today all these services are free. But lack of income means that few resources are available to monitor quality. A recent experiment showed that all of them were quite incomplete, and the listed linkages at most provided initial entrypoints for a dedicated researcher. Most of the documents obtained do not carry copyright. If they do, copyright may well be violated by the use of these tools.

3.2 Selection and review

Today we count on intermediaries to help locate high-quality and relevant information. Reviewers and editors filter much material prior to publication, and indexers, librarians and post-publication reviews provide additional services [Wiederhold:94L]. The added value of review, managed by publishers, is not easy to manage when much material is placed by the authors on the WWW. Subscribers to publications will be lost if the equivalent material is available to anyone by integrating documents according to a list of current contents, which is easy to place on the Internet, since most papers will be on the net as well.

Forward-looking publishers are trying to resolve some of these issues, but the conflict between free dissemination of information and reimbursement for value provided is great [Maurer:94]. An intermediary agent, taking the role of a publisher, sustained by a mediator, might be able to filter material with respect to quality and accessors with respect to privileges gained by subscription or payment of fees.

3.3 Views and objects

Designers of relational databases have used *views* to define subset of source databases that are relevant to a particular application. For a mediator, the view would be determined by the scope of the domain. Rather than creating a single, unnormalized relation representing the view, as is done by default in relational systems, it is better to create object classes corresponding to concepts relevant for the domain [Barsalou:90]. When the semantics of such objects are carefully managed, the view-update problem can be resolved effectively, permitting updating as well as retrieval of domain-specific information. Objects are always hierarchically structured, making them easy to navigate and understand.

3.4 Summarization

The volume of material on the World-Wide networks is huge and growing. A significant added-value service is provided by summarization. A recent example is the automated creation tables describing the status of publically traded companies based on the K-10 reports they submit to the regulators [Krishnan:94]. Seeking for exceptions is an even more effective summarization technique, since now only information that needs followup is transmitted to the user. An example here is the warnings issued by some tax-preparation programs that a person's tax deductions is outside of the normal range and likely to trigger an audit by the income tax people.

3.5 Knowledge-based computing

A mediating module carries out tasks to serve a user, and a desirable aspect of such a computer-based servant is that it has a model to *understand* what is needed, can be directed by the user, and, as a prerequisite for direction, can explain to the user the current information structure of the task to be performed. Database systems do that by exposing their schema; in mediation the application's hierarchical model and the matches made to the resources accessed have to be made visible. While understanding the intent of user in general is very hard, when the tasks are well organized and suitably limited, the presentation of the linkages is feasible. Such an understanding must be formalized into a machine-processable model. If well represented, the model can drive the computations required, and tasks as summarization become quite manageable [Chaudhuri:90].

Slide 8 goes here

Figure 3. Mediated Application View into the Resource Network

3.6 Ontologies and mediation

It is crucial to have a simple model of the user's application structure, and of the available resources in the domain. The linkage between the two is represented by the domain *ontology*. The model of the application, representing the current view of the user as it pertains to the mediator, consists of a listing of concepts and relationships among them: the task ontology. It must be a subset of the ontology of the domain of the mediator.

The resources also present their ontologies, sometimes only simple database schemas. Those resource ontologies must also be known in the mediators domain, so that appropriate matching can be carried out. The relationships found can be seen as candidate paths among the concepts, where the destinations nodes indicate the information that can be obtained from the resources. The linkages among the resources form a network, which changes over time.

Complementing the ontology are the processing paradigms used in that domain. The concepts are typically hierachically structured, following the common divide-and-conquer paradigm used for problem-solving. Resources are typically at the leaf nodes, and the processing can proceed in a well-structured manner. A mediator node will manage access and processing of data corresponding to the models it obtains from applications and sources, the method for processing will be appropriate for its domain.

4. Ontologies

To permit information from distinct sources to be accessed by a mediator, there must be agreement on the terminology in the shared area. It is not sufficient to match information on the basis of words, because words differ in meaning in different contexts. For each domain an ontology must be defined: a vocabulary of terms and a specification of their relationships. With each term a definition is needed, both an informal annotation, perhaps with some examples, and a more formal specification of relationships, as *part-of*, *owned-by*, *referenced-by*, *subclass-of*, etc., which cite other terms in the domain. The entire structure forms a semantic net. Isolated terms should be avoided.

Slide 16 goes here

Figure 4. Establishing Ontologies

4.1 Sources

Initial sources for ontologies are defined vocabularies for a domain, perhaps represented by a textbook on the topic, thesauri that have been created to help indexers create bibliographies and help researchers locate references [Humphreys:92]. Smaller, but practical and verifiable ontologies can be obtained from database schemas. In the original design document for a database additional information may be available, as the relationships among represented entities and constraints imposed on the attributes.

Standardization efforts have contributed much effort towards ontologies, since the proper definition of a standard depends on getting all subsidiary terms precisely defined [HT:93]. Some large ontologies have been created for knowledge-based processing, since it becomes essential that the human understanding of the processes is faithfully transmitted to the computer [Lenat:90]

Only recently have the results of gathering terminologies and structuring their relationships been recognized as a valuable effort in its own right, deserving of attention. The work of Tom Gruber traces that development [Gruber:91–94]. Most current candidate ontologies are hidden, since they are part of the infrastructure for some effort in communicating among people, or between people and systems they share.

4.2 Ontology integration

Domain differences can make a once-and-for-all integration of distinct domain infeasible. People living in their domain cannot be forced to become consistent with the concepts of others. They cannot be kept from improving their view of the world. A capability to interoperate dynamically is essential if we wish to achieve associative access to multiple domains, since the transformations required to achieve optimization must maintain the correct semantics. We have experience with that issue in object-databases, since objects tend to represent a hierarchical view while the underlying world of data allows arbitrary composition via the relational algebra [Wiederhold:89]. For instance, the PENGUIN system constructs objects as needed out of relational databases, given a structural model of connecting references [Barsalou:91]. Linking relational algebras to object technology is showing good results now [RDCLPS:94].

There are several approaches to dealing with building composed ontologies from domains that have ontological differences:

1 Aggregate the terms from all relevant ontologies, give them to a committee, and ask them to prepare definitions that are acceptable to all. When the definitions are completely documented, release the document and expect that all participants will adjust their usage to conform to the definitions [HT:93].

2 Assume that terms match, and when mismatches are discovered, make the terms distinct, typically by prefixing them with some source or domain identifiers. This is the approach used by UMLS [Humphreys:92]; all the sources are labeled to make such distinctions easy, and by CYC, where micro theories can encapsulate differences Lenat:90]. Over time, the processes of sharing of information encouraged by the availability of the joint ontology will cause convergence of meanings, although coherence can never be assured.

3 Assume that terms from disjoint never mean the same thing unless one's system has been explicitly informed. Such information, encoded as *matching rules*, forms a knowledge-base to be managed by collaborators from two or more domains [Wiederhold:94N]. No restrictions are imposed on the evolution of local terms within a domain. Terms that are covered by matching rules form a new, second layer abstract ontology. Higher abstract layers can be defined recursively, leaving unneeded abstract terms local in their abstract layer.

We focus here on the third alternative.

4.3 An Example for Limited Domain Sharing

A multi-domain algebra needs the knowledge about the domains, specifically about the semantics of the intersecting terms.

We will illustrate the concept with a simple example:

S Domain **S** is of shoe stores, with objects as shoes to be sold, customers, their feet, sales people, business locations, and suppliers.

F Domain **F** is of shoe factories, with shoes being produced, lasts, materials as leather, glue, nails, and thread, suppliers for the material, employees, and production machinery.

In order to create an information system that combines data from both, it is not necessary to merge both the **S** and **F** ontologies completely. Only terms along their connections must be merged; we assume by default that terms do not match. Matching rules are required to match terms between our domains:

```
S:supplier.name = F:factory.name
S:shoe.size     = F:shoe.size
S:shoe.color    = color_table (F:shoe.color)
```

The color table provides the translation between the colors being attached to sales items, such as `pretty pink` and the color codes used in the factory, say, `13XF3`. Sometimes such relationships can be expressed as functions, say, conversions from `cm` to `inches`. Figure 5 illustrates the intersection and some matching rules.

Slide NPGS slides goes here

Figure 5. Example of Intersecting Ontologies

The income tax domain **I** will establish other connections between it and the sales and factory domains. A department store, incorporating many sales subdomains, will have more semantic connections, but can still avoid an unconstrained union of all its component ontologies.

4.4 Domain-specificity and domain interoperation

We achieve scalability of information systems in this approach by the ontological partitioning. We enable composition over the parts by having a knowledge-based algebra. The individuals chartered with defining and maintaining the knowledge need more breadth than those that maintain domain-specific ontologies, but do not need the same depth of knowledge for the shoe supply connection. No knowledge about manufacturing detail is needed, although the factory may provide an abstraction called `quality`.

Not included in the knowledge-base, and hence not composable, is the term `nail`, which in the store domain **S** is part of the customer's anatomy, and in the factory **F** designates part of the material used to make shoes. Similarly, the `employees` remain distinct, since the data collected for sales people differ from those in the factory. We show that independence also in Figure 5.

Constraining terms to a domain is useful, since most ambiguities disappear when the domain context is known. When a term is used outside of its context, it is labeled with its source domain, for example: *Carpentry.miter* versus *Religion.miter*. These domains are largely disjoint, and terminology can be maintained independently. It is actually likely that the domain *Religion* will be subdivided into hundreds of autonomous domains, since it is likely that no agreement can be reached for many crucial terms. The term *miter* need only occur in a few of those subdomains.

4.5 A DKB algebra

However, keeping all terms disjoint disables the domain interoperation we seek. Automatic reasoning about ontologies requires a formal approach to transformation and their equivalences, eventually enabling optimization of access. We have to develop a set of operations that permits us to match and merge ontologies: an *algebra* over ontologies. Only a few operations seem to be needed, namely Intersection, Union, Difference, and Map.

Given a formal *Domain-Knowledge-Base* model (DKB) containing matching rules that define sharable terms, the DKB-algebra should contain the following binary operations among domains:

Operation	symbol	semantics
DKB-Intersection	$\bigcap_{(DKB)}$	create a new subset ontology, comprised of sharable entries
DKB-Union	$\bigcup_{(DKB)}$	create a new joint ontology, labeling all but shared entries with their source
DKB-Difference	$-_{(DKB)}$	remove shared entries from an ontology, unique entries are retained

Simple negation is avoided, so that no infinite ontologies are created. We expand now on our expectations for these operations.

- $\bigcap$ Intersection ($\bigcap$) is the most critical operation, akin to the Join-operation in a relational algebra. It creates the list of terms that have been defined by mapping rules as belonging to both domains, and hence requires interpretation of the mapping rules. The interpretation may involve secondary terms if they are needed to define constraints on the primary terms. The objective remains to minimize the size of the intersection, since that will render the result more manageable. In our example, the terms *size* and *ShoeSales.color = f(ShoeFactory.colorcode)* are in the intersection, and the table expressing the mapping relationships f has to be maintained as well. The *Employees* do not intersect in our example. If there would be person, say a *Broker*, who is regarded as an *Employee* on both sides, then a rule of the form

  Include.$\bigcap$*: 'Employee' if 'ShoeSales.Employee.jobtitle' = 'Broker' and*
  if 'ShoeSales.Employee' = 'ShoeFactory.Employee'

  would be required, and the maintenance of the matching rules requires more breadth. Of course, if the broker hides the fact of receiving double pay, then no such rule would appear until an investigation discovers that fact.

- $\bigcup$ Taking the Union of ontologies ($\bigcup$) creates the merger of the source terms. For those terms that have a defined intersection rule the single result is given; the remainder are copied as they are. This operation creates a consistent new ontology, but little of value is added, since now similar prime terms continue to have their old domain identification. The Λ operation is helpful to search for similar terms that are unmatched, and consider placing them in the intersection. However, again, there has to be value to have them in the intersection $\bigcap$, since when trying to manage large-scale systems keeping the bases small is of crucial importance. The operation may be best used to combine the small ontologies that were combined by prior intersections; for instance, a department store may want to manage all its supplier relationships together, and hence requires an aggregated ontology.

$-$ The Difference operation ($-$) completes the algebra, and permits creating asymmetric subsets for further analysis. Its presence compensates for the absence of negation. An analyst may, for instance. want a list of all terms in an ontology that are not already covered by other ontologies.

$\mapsto$ To extend the capabilities for matching terms from disjoint ontologies, the Map operation ($\mapsto$) permits transformation within an ontology, so that more terms become candidates for matching and for the creation of useful intersections. Internal mapping assigns the responsibility for mapping rules to the owners of a specific domain, rather than to the managers of the mediators that access diverse domains. For instance, there is a relationship between the *ShoeSales.price* and the *ShoeSales.profi* which requires the *ShoeFactory.price*. We can define *ShoeSales.cost* $\mapsto$*(ShoeSales.price, ShoeSales.profit, ShoeSales.overhead* and now have a simpler attribute for computing the intersections.

An algebra as outlined above provide the capability for interrogating multiple databases which are semantically disjoint but where a shared knowledge-base has been established. This process mirrors the approach used in CARNOT, where a knowledge base is used to create *articulation axioms* for joining of data [Collet:91]. However, CARNOT uses the default assumption that everything matches. When CARNOT uses a large and broad CYC knowledge base, many irrelevant retrievals can occur, so that in practice CARNOT system applications limit the depth of search.

An abstract layer created by taking the union ($\bigcup_{(DKB)}$) of several prior intersections ($\bigcap_{(DKB)}$) should not contain so many terms that coherence is hard to achieve. The relative autonomy of the local source terms provides scalability. The layered structure actually adopts for information structuring the domain management strategy used by the INTERNET distributed naming conventions [Kahn:87].

With the conservative assumptions embedded in the *DKB-model*, the risk is that, because of having insufficiently many matching rules, too little information will be retrieved. By assigning the task of creating matching rules to multiple specific expert groups, we expect that high quality operations over data from distinct, but overlapping domains can be created at a reasonable cost. To evolve these systems effectively, feedback loops must exist that permit users to suggest new candidate matching rules, or to modify existing ones. Having small, distributed groups to maintain the partitioned *DKB-model*s will help ensure responsive maintenance of the domain knowledge.

Note that we stress maintenance throughout. It is important that new systems for our networks are designed with maintenance in mind, else we will follow the example of the traditional software industry where 60 to 95% of the overall cost is in maintenance, and yet the users complain that the services are always out-of-date. Keeping ontologies small enables semantic agreements among the people using them with little lag time. If maintenance takes too long, some terms will change in meaning during the time it takes to agree.

Support of convergence of mutual understanding, is a major motivation for formal management of ontologies. By enabling on-line interoperation people can exchange information with an intensity that other approaches cannot match. This interaction and interdependency will lead to convergence of terminology and semantics.

5. Conclusion

Information technology is serving us well in specific domains, although we have remained dependent on specialist model designers and programmers for the implementation. Object technology has lessened our dependence on specialists by being able to use an infrastructure which aggregates detail into meaningful units in many important domains.

Mediation is a technology that is intended to scale systems so that sources from many domains can contribute services and information to the end-user applications. The partitioning into domains creates a desirable autonomy, but also isolation. Careful management of intersections among disjoint domain ontologies can establish sharing according to defined matching rules. These matching rules are the responsibility of integration experts, typically the owners of the mediators. We propose a knowledge-based algebra, dealing initially with the ontological foundation, to help automate information integration using the knowledge embodied in these rules. The tasks of collecting and maintaining the matching rules to support such algebras can be naturally partitioned among specialists and collaborating integrators.

The entire mediation architecture focuses on maintenance. This focus distinguishes the mediated approach from many other proposals, which attempt to design optimal systems. In large systems the major costs are in integration and maintenance, rather than in achieving initial functionality and optimality. Integration in mediation can proceed at multiple levels of abstraction, avoiding the centralization that hinders progress in data exploitation of data from diverse sources.

Tools are needed to support such development, but to have effective tools a common formal structure is needed. We cited some early tools used to support the initial mediation projects. The emerging standards have been made public on the networks, to avoid proprietary dominance. Rapid development is possible by exploiting the same modern networks that make the architecture itself feasible. With network access to standards and the potential consumers, whoever build the best tools or provides the best services will gain the deserved advantage.

Both software and artificial intelligence technologies has been hard to scale when domains grew large or became diverse. The technology we described can provide the needed formalism for scaling, by building on concepts demonstrated in relational algebras, formal management of semantics, and the incorporation of ontological concepts as a foundation for the management of the required knowledge bases.

Acknowledgement
This research direction would not have been feasible without the support of many colleagues in the ARPA Intelligent Integration of Information (I3) program, and, of course, from ARPA itself. Recent workshops sponsored by Bob Neches and David Gunning of ARPA, and chaired by Bill Mark of Lockheed, Tom Gruber of EITech, and others have contributed valuable ideas. Work in progress on these topics can be located via the World-Wide Web node http://www-ksl.stanford.edu/knowledge-sharing . Voy Wiederhold and Waqar Hasan provided feedback and corrections for this paper.

References
Since much of this work is recent, the list provides handles to the World-Wide-Web, where much of the current information appears, especially when it deals with applying the network's capabilities to today's systems.

[Barsalou:91]T. Barsalou, N. Siambela, A. Keller[A, and G. Wiederhold: "Updating Relational Databases through Object-Based Views"; *ACM SIGMOD Conf. on the Management of Data 91*, Boulder CO, May 1991.

[Chaudhuri:90] Surajit Chaudhuri: "Generalization and a Framework for Query Modification"; *Proc. IEEE CS Intl. Conf. on Data Engineering 6*, Feb. 1990.

[Collet:91]C. Collet, M. Huhns, and W-M. Shen: "Resource Integration Using a Large Knowledge Base in CARNOT"; *IEEE Computer*, Vol.24 No.12, Dec.1991.

[Cox:94] Brad Cox: The Coalition On Electronic Markets; http://www.site.gmu.edu/ bcox/CEM/00CEM.html

[FFMM:94] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire: "KQML as an Agent Communication Language"; to appear in *The Proceedings of the Third International Conference on Information and Knowledge Management* (CIKM'94), ACM Press, November 1994.

[GK:94] Michael Genesereth and Steven Ketchpel: "Software Agents"; *Comm. ACM*, Vol.37 No.7, July 1994, pp.48-53,147.

[Gravano:94] Luis Gravano, Hector Garcia-Molina, and Anthony Tomasic: "The Effectiveness of GLOSS for the Text Database Discovery Problem"; *Proc. of the 1994 ACM SIGMOD*, ACM Sigmod Record, Vol.23 No.2 May 1994, pp.126–137.

[Gruber:91] T.R.— Gruber: " The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases"; in Allen, Fikes, Sandewall (eds): *Principles of Knowledge Representation and Reasoning*: Morgan Kaufmann, 1991.

[Gruber:92O] T.R. Gruber: ONTOLINGUA*: A Mechanism to Support Portable Ontologies*; Knowledge Systems Laboratory, KSL-91-66, November 1992.

[Gruber:92P] T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing"; in Gurino (ed) *International Workshop on Formal Ontology*, Padova, Italy, 1992.

[Gruber:93]Thomas R. Gruber: "A Translation Approach to Portable Ontology Specifications"; *Knowledge Acquisition*, Vol.5 No. 2, pp.199–220, 1993

[Gruber:94]Thomas R. Gruber and Gregory Olsen: "An Ontology for Engineering Mathematics"; in Doyle, Torasso, and Sandewall (eds.): *Fourth International Conference* *on Principles of Knowledge Representation and Reasoning*, Gustav Stresemann Institut, Bonn, Germany; Morgan Kaufmann, May 1994.

[Haddock:94]G. Haddock and K. Harbison: "From Scenarios to Domain Models: Processes and Representations"; *Proceedings of the onference on Knowledge-based Artificial Intelligence Systems in Aerospace and Industry*, SPIE, April 1994.

[Humphreys:92]B.L. Humphreys and D.A.B. Lindberg: "The Unified Medical Language Project: A Distributed Experiment in Improving Access to Biomedical Information"; *MEDINFO 92*, North-Holland, 1992, pp.1496–1500.

[HT:93] Bob Hodges and Craig Thompson (eds.): *Proceedings of the Workshop on Application Integration Architectures*; organized by Texas Instruments, published by NIST, Feb.1993.

[ISX:94] ISX Corporation: Intelligent Integration of Information (I3); http://isx.com/pub/I3, Westlake Village CA, 1994

[Kahn:87]Robert E. Kahn: "Networks for Advanced Computing"; *Scientific American*, Vol 257 No.5; Oct.1987, pp.136-143.

[Krishnan:94] R. Krishnan and D.H. Steier: "Adaptive access to a digital library of corporate information"; *Digital Libraries Workshop*, Rutgers Un., Newark NJ, May 20-21, 1994. http://www.cs.umbc.edu/conferences/dl/abstracts/krishnan.html

[Lenat:90]D. Lenat, R.V. Guha, et al.: "CYC: towards programs with common sense"; *Communications of the ACM*, Vol.33 No.8, Aug.1990.

[Loomis:87]Mary E.S. Loomis: *The Database Book*; MacMillan, 1987.

[Maurer:94] Herrman Maurer: "Advancing the Ideas of World Wide Web: Hyper-G"; *Proc. Distributed Multimedia Systems and Applications*, Honolulu, 1994, pp.201–203; http://iicm.tu-graz.ac.at/pub/Hyper-G.

[NFFGPSS:93] R. Neches, R. Fikes, T. Finin, T.R. Gruber, R. Patil, T. Senator, and W.R. Swartout: "Enabling Technology for Knowledge Sharing"; *AI Magazine*, Vol.12 No.3, pp.37–56, 1993.

[PFPMFGR:92] R.S. Patil, R.E. Fikes, P.F. Patel-Schneider, D. Mckay, T. Finin, T.R. Gruber, and R. Neches: The DARPA Knowledge Sharing Effort: Progress Report; in Nebel, Rich and Swartout, eds., *Principles of Knowledge Representation and Reasoning*; pp.777–788, Morgan Kaufmann, 1992.

[RDCLPS:94] B. Reinwald, S. Dessloch, M. Carey, T. Lehman, H. Pirahesh and V. Srinivasan: "Making Real Data Persistent: Initial Experiences with SMRC"; *Proc. Int'l Workshop on Persistent Object Systems*, Tarascon, France, pp.194–208, Sept.1994.

[Rose:83]Robert F. Rose: "A 'Data Engine' Using SAS and INQUIRE"; *Journal of Medical Systems*, Vol.7 No.3, 1983, pp.257–266.

[Roussopoulos:91] Nick Roussopoulos: "An Incremental Access Method For Viewcache: Concept, Algorithm, and Cost Analysis; *ACM Transactions on Database Systems*, Sep. 1991, Vol.16 No.3.

[Salton:90] Gerard Salton: "Full Text Information Processing Using the Smart System"; *IEEE CS Database Engineering Bulletin*, March 1990, Vol.13 No.1.

[SDKLPSS:94] Michael Stonebraker, Robert Devine, Marcel Kornacker, Witold Litwin, Avi Pfeffer, Adam Sah, and Carl Staelin: "An Economic Paradigm for Query Processing and Data Migration in Mariposa"; *3rd Intl. Conf. on Parallel and Distributed Information Systems*, (PDIS) Austin, Texas, September, 1994.

[WCC:94] Gio Wiederhold, Stephen Cross, Charles Channell: *Information Integration*; IEEE Educational Videotape, 2 hours, October 1994, Robert Kahrman, sponsor. IEEE, Picataway NJ.

[Wiederhold:89] Gio Wiederhold: "Views, Objects, and Databases"; *IEEE Computer*; Vol.19 No.12, December 1986, pp.37-44.

[Wiederhold:92C] Wiederhold, Gio:"Mediators in the Architecture of Future Information Systems"; IEEE Computer, March 1992, pp.38-49.

[Wiederhold:92I]Gio Wiederhold: "'The Roles of Artficial Intelligence In Information Systems"; *Journal of Intelligent Information Systems*, Vol.1 No.1, 1992, pp.35–56.

[Wiederhold:92N]Gio Wiederhold, Peter Wegner, and Stefano Ceri: "Towards Megaprogramming"; *Comm. ACM*, November 1992, pp.89-99.

[Wiederhold:94L] Gio Wiederhold: Digital Libraries, and Productivity; submitted for publication, Oct.1994 .

[Wiederhold:94N] Gio Wiederhold: "An Ontology Algebra"; *Proceedings of the Monterey Workshop on Formal Methods*, Luqi (ed.), U.S. Naval Post Graduate School, Sept. 1994.