From Databases to Dataspaces: A New Abstraction for Information Management

Michael Franklin
University of California, Berkeley

Alon Halevy
Google Inc. and U. Washington

David Maier
Portland State University

Abstract

The development of relational database management systems served to focus the data management community for decades, with spectacular results. In recent years, however, the rapidly-expanding demands of “data everywhere” have led to a field comprised of interesting and productive efforts, but without a central focus or coordinated agenda. The most acute information management challenges today stem from organizations (e.g., enterprises, government agencies, libraries, “smart” homes) relying on a large number of diverse, interrelated data sources, but having no way to manage their *dataspaces* in a convenient, integrated, or principled fashion. This paper proposes dataspaces and their support systems as a new agenda for data management. This agenda encompasses much of the work going on in data management today, while posing additional research objectives.

1. Introduction

A Database Management System (DBMS) is a generic repository for the storage and querying of structured data. A DBMS offers a suite of interrelated services and guarantees that enables developers to focus on the specific challenges of their applications, rather than on the recurring challenges involved in managing and accessing large amounts of data consistently and efficiently.

Unfortunately, in data management scenarios today it is rarely the case that all the data can be fit nicely into a conventional relational DBMS, or into any other single data model or system. Instead, developers are more often faced with a set of loosely connected data sources and thus must individually and repeatedly address low-level data management challenges across heterogeneous collections. These challenges include: providing search and query capability; enforcing rules, integrity constraints, naming conventions, etc.; tracking lineage; providing availability, recovery, and access control; and managing evolution of data and metadata.

Such challenges are ubiquitous – they arise in enterprises (large or small): within and across government agencies, large science-related collaborations, libraries (digital or otherwise), battlefields, in “smart” homes, and even on one’s PC desktop or other personal devices. In each of these scenarios, however, there is some identifiable scope and control across the data and underlying systems, and hence one can identify a space of data, which, if managed in a principled way, will offer significant benefits to the organization.

In this article we introduce dataspaces as a new abstraction for data management in such scenarios and we propose the design and development of DataSpace Support Platforms (DSSPs) as a key agenda item for the data management field. In a nutshell, a DSSP offers a suite of interrelated services and guarantees that enables developers to focus on the specific challenges of their applications, rather than on the recurring challenges involved in dealing consistently and efficiently with large amounts of interrelated but separately managed data. We begin our discussion of dataspaces and DSSPs by placing them in the context of existing systems.

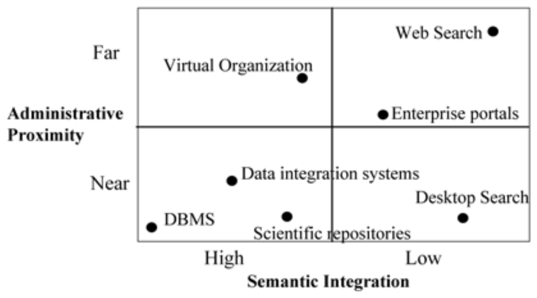

Figure 1. A space of data management solutions.

1.1 Data Management Architectures

Figure 1 shows a categorization of existing data management solutions along two dimensions. “Administrative Proximity” indicates how *close* the various data sources are in terms of administrative control. “Near” means that the sources are under the same or at least coordinated control, while “Far” indicates a looser coordination tending towards none at all. The closer the administrative control of a group of data sources, the stronger the guarantees (e.g., of consistency, permanence) that can be provided by the data management system.

“Semantic Integration” is a measure of how closely the schemas of the various data sources have been matched. In other words, how well the types, names, units, meanings, etc. of the data in the sources are matched up. At the high end of the spectrum, all data conforms to a single agreed-upon schema. At the low end, there is no schema information at all. In between lay various data integration solutions and approaches based on semi-structured data and controlled vocabularies. This dimension indicates the degree to which semantically rich query processing and data manipulation can be provided across a group of data sources, with higher degrees of integration providing richer functionality.

As shown in the figure, traditional DBMSs represent only one (albeit, an important one) point solution in today’s data management environment. DBMSs require all data to be under the control of a single administrative domain and to conform to a single schema. In return for these limitations, a DBMS is able to provide rich data manipulation and query processing with well-understood, strong semantics, as well as strict transactional guarantees for updates, concurrency, and persistence (the so-called “ACID” properties).

An important point in Figure 1 is “data integration systems”. In fact, traditionally, data integration and data exchange systems have aimed to offer many of the purported services of dataspace systems. The distinction is that data integration systems require *semantic integration* before any services can be provided. Hence, although there is not a single schema to which all the data conforms, the system knows the precise relationships between the terms used in each schema. As a result, significant upfront effort is required in order to set up a data integration system.

Dataspaces are not a data integration approach; rather, they are more of a *data co-existence* approach. The goal of dataspace support is to provide base functionality over all data sources, regardless of how integrated they are. For example, a DSSP can provide keyword search over all of its data sources, similar to that provided by existing desktop search systems. When more sophisticated operations are required, such as relational-style queries, data mining, or monitoring over certain sources, then additional effort can be applied to more closely integrate those sources in an incremental, "pay-as-you-go" fashion.

Similar flexibility exists along the administrative proximity dimension of Figure 1. If administrative autonomy is desired then the DSSP will not be able to provide certain guarantees in terms of consistency, durability of updates, etc. As stronger guarantees are desired, more effort can be put into making agreements among the various owners of data sources and opening up certain interfaces (e.g., for commit protocols).

To summarize, the distinguishing properties of dataspace systems are the following:

- A DSSP must deal with data and applications in a wide variety of formats accessible through many systems with different interfaces. A DSSP is required to support *all* the data in the dataspace rather than leaving some out, as with DBMSs.
- Although a DSSP offers an integrated means of searching, querying, updating, and administering the dataspace, often the same data may also be accessible and modifiable through an interface native to the system hosting the data. Thus, unlike a DBMS, a DSSP is not in full control of its data.
- Queries to a DSSP may offer varying levels of service, and in some cases may return *best-effort* or approximate answers. For example, when individual data sources are unavailable, a DSSP may be capable of producing the best results it can, using the data accessible to it at the time of the query.
- A DSSP must offer the tools to create tighter integration of data in the space as necessary.

1.2 A Dataspace Agenda

By all measures, the data management research community remains active, vibrant, and growing. The concern has been raised, however, that the community currently lacks a central focus — a "relational DBMS" equivalent for the new world of disparate decentralized data.[1]. Furthermore, there is a growing feeling among many, that the term "database research" is too restrictive for the breadth of topics being addressed by the community. While it may be possible that the field has simply grown too large to accommodate a single, succinct vision, this paper is intended as one proposal that could help further a discussion of the issues.

The database community has long had a process of self-assessment in which senior researchers meet periodically to survey the state of the field and to identify promising research areas for the future (the most recent of these are the 1998 Asilomar Report [BBC$^+$98] and the 2005 Lowell Self-Assessment [AAB$^+$05]). This paper builds on many of the goals and challenge problems identified in those earlier reports. In fact, much of the research in the data management community already falls squarely into the requirements of dataspaces and DSSPs, including areas such as schema mapping, data integration and model management, uniform search over multiple types of data; combining structured, semi-structured, and unstructured data, approximate query processing; managing and querying uncertain data and its lineage, and stream and sensor data management and processing. Thus, dataspaces can be viewed simply as an umbrella for these varied efforts. As we discuss later, however, we also believe that the holistic view taken by dataspaces and DSSPs can itself lead to a new set of research challenges.

The remainder of this paper is as follows. Section 2 motivates the need for dataspace systems with two prototypical examples. Section 3 describes the logical components of a dataspace and a first attempt at an architecture of a DSSP. Section 4 outlines several research challenges critical to building DSSP, and Section 5 discusses a few perspectives on the agenda. Section 6 concludes.

2. Examples

We begin by describing two dataspace scenarios.

Personal Information Management: The goal of Personal Information Management (PIM) is to offer easy access and manipulation of all of the information on a person's desktop, with possible extension to mobile devices, personal information on the Web, or even all the information accessed during a person's lifetime.

Recent desktop search tools are an important first step for PIM, but are limited to keyword queries. Our desktops typically contain some structured data (e.g., spreadsheets) and there are important associations between disparate items on the desktop. Hence, the next step for PIM is to allow the user to search the desktop in more meaningful ways. For example, "find the list of students who took my database course last quarter", or "compute the aggregate balance of my bank accounts". We would also like to search by association, e.g., "find the email that John sent me the day I came back from Hawaii", or "retrieve the experiment files associated with my SIGMOD paper this year". Finally, we would like to query *about* sources, e.g., "find all the papers where I acknowledged a particular grant", "find all the experiments run by a particular student", or "find all spreadsheets that have a variance column".

The principles of dataspaces in play in this example are that (1) a PIM tool must enable accessing *all* the information on the desktop, and not just an explicitly chosen subset, and (2) while PIM often involves integrating data from multiple sources, we cannot assume users will invest the time to integrate. Instead, most of the time the system will have to provide best-effort results, and tighter integrations will be created only in cases where the benefits will clearly outweigh the investment.

Scientific data management: Consider a scientific research group working on environmental observation and forecasting. They may be monitoring a coastal ecosystem through weather stations, shore- and buoy-mounted sensors and remote imagery. In addition they can be running atmospheric and fluid-dynamics models that simulate past, current and near-future conditions. The computations may require importing data and model outputs from other groups, such as river flows and ocean circulation forecasts. The observations and simulations are the inputs to programs that generate a wide range of data products, for use within the group and by others: comparison plots between observed and simulated data, images of surface-temperature distributions, animations of salt-water intrusion into an estuary.

Such a group can easily amass millions of data products in just a few years. While it may be that for each file, someone in the group knows where it is and what it means, no one person

[1] For instance, this issue was raised and discussed perhaps most publicly at the CIDR 2005 Conference

may know the entire holdings nor what every file means. People accessing this data, particularly from outside the group, would like to search a master inventory that had basic file attributes, such as time period covered, geographic region, height or depth, physical variable (salinity, temperature, wind speed), kind of data product (graph, isoline plot, animation), forecast or hindcast, and so forth. Once data products of interest are located, understanding the lineage is paramount in being able to analyze and compare products: What code version was used? Which finite element grid? How long was the simulation time step? Which atmospheric dataset was used as input?

Soon, such groups will need to federate with other groups to create scientific dataspaces of regional or national scope. They will need to easily export their data in standard scientific formats, and at granularities (sub-file or multiple file) that don't necessarily correspond to the partitions they use to store the data. Users of the federated dataspace may want to see collections of data that cut across the groups in the federation, such as all observations and data products related to water velocity, or all data related to a certain stretch of coastline for the past two months. Such collections may require local copies or additional indices for fast search.

This scenario illustrates several dataspace requirements, including (1) a dataspace-wide catalog, (2) support for data lineage and (3) creating collections and indexes beyond what any one participating source supplies.

3. Dataspaces

We now describe the logical components of a dataspace and the services we expect from a DSSP.

3.1 Logical Components of Dataspaces

A dataspace (see Figure 2) should contain all of the information relevant to a particular organization regardless of its format and location, and model a rich collection of relationships between data repositories. Hence, we model a dataspace as a set of *participants* and *relationships*.

The participants in a dataspace are the individual data sources: they can be relational databases, XML repositories, text databases, web services and software packages. They can be stored or streamed (managed locally by data stream systems), or even sensor deployments.

Some participants may support expressive query languages, while others are opaque and offer only limited interfaces for posing queries (e.g., structured files, web services, or other software packages). Participants vary from being very structured (e.g., relational databases) to semi-structured (XML, code collections) to completely unstructured. Some sources will support traditional updates, while others may be append-only (for archiving purposes), and still others may be immutable.

A dataspace should be able to model any kind of relationship between two (or more) participants. On the more traditional end, we should be able to model that one participant is a view or a replica of another, or to specify a schema mapping between two participants. We would, however, like to model a much broader set of relationships such as, that source A was manually curated from sources B and C, or that sources E and F were created independently, but reflect the same physical system (e.g., mouse DNA). Relationships may be even less specific, such as that two datasets came from the same source at the same time.

Dataspaces can be nested within each other (e.g., the dataspace of the CS department is nested within the dataspace of the university), and they may overlap (e.g., the dataspace of the CS department may share some participants with the EE department). Hence, a dataspace must include access rules between disparate dataspaces. In general, there will be cases where the boundaries of a dataspace may be fluid, but we expect that in most of the cases the boundaries will be natural to define.

3.2 Dataspace Services

Along with content heterogeneity comes the need to support multiple styles of access to the content. We envision that DSSPs will allow many different modes of interaction and we aspire to be as general as possible in allowing the application of different services to different types of content.

One of the most basic dataspace services is cataloging data elements from the participants. A catalog is an inventory of data resources, with the most basic information about each, such as source, name, location in source, size, creation date and owner, and so forth. The catalog is infrastructure for most of the other dataspace services, but can also support a basic browse interface across the dataspace for users.

Two of the main services that a DSSP will support are search and query. While DBMSs have excelled at providing support for querying, search has emerged as a primary mechanism for end users to deal with large collections of unfamiliar data. Search has the property that it is more forgiving than query, being based on similarity and providing ranked results to end users, and supporting interactive refinement so that users can explore a data set and incrementally improve their results. A DSSP should enable a user to specify a search query and iteratively refine it, when appropriate, to a database-style query. A key tenet of the dataspaces approach is that search should be applicable to all of the contents of a dataspace, regardless of their formats.

Universal search and query should extend to meta-data as well as data. Users should be able to discover relevant data sources and inquire about their completeness, correctness and freshness. In fact, a DSSP should also be aware of *gaps* in its coverage of the domain.

A DSSP will also support updating data. Obviously, the effects of updates will be determined by the mutability of the relevant data sources. A major research issue in dataspaces is the development and provision of guaranteed update semantics in a heterogeneous, highly-autonomous environment.

Other key DSSP services include monitoring, event detection, and support for complex workflows. For example, we may want to set up a computation to happen when a new piece of data arrives, and have the results of that computation distributed to a set of recipient data sources. Similarly, a DSSP should support various forms of data mining and analysis.

Not every participant in a dataspace will necessarily provide the interfaces necessary to support all DSSP functions. Thus, there will be the need to extend data sources in various ways. A source might not actually store its own metadata, so we may require an independent metadata repository for such sources. Information may need to be "externalized" from a source or its context. For example, a list of emergency services agencies from Washington might need to be explicitly labeled "Washington" in order to combine it with similar lists from Oregon and California. Or a scientific dataset might need a superimposed schema. The data elements in a source might be enhanced with annotations, ratings, links to elements in other sources. Monitoring support may need to be provided for participants that lack their own notification service.

3.3 Dataspace Systems

We now outline one possible set of components and architecture for a dataspace system. As depicted in Figure 2, a DSSP offers several interrelated services on the dataspace, some of which are generalizations of components provided by a traditional DBMS.

It is important to keep in mind that unlike a DBMS, a DSSP does not assume complete control over the data in the dataspace. Instead, a DSSP allows the data to be managed by the participant

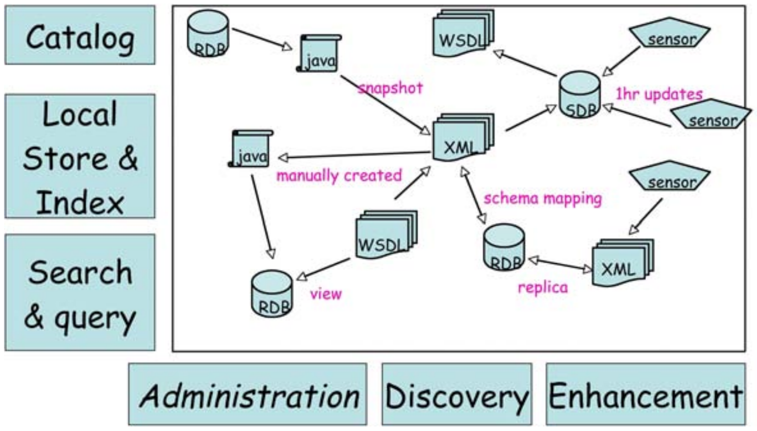

Figure 2. An example dataspace and the components of a dataspace system.

systems, but provides a new set of services over the aggregate of these systems, while remaining sensitive to the autonomy needs of the systems. Furthermore, we may have several DSSPs serving the same dataspace – in a sense, a DSSP can be a personal view on a particular dataspace.

Catalog and Browse: The catalog contains information about all the participants in the dataspace and the relationships among them. The catalog must be able to accommodate a large variety of sources and support differing levels of information about their structure and capabilities. In particular, for each participant, the catalog should include the schema of the source, statistics, rates of change, accuracy, completeness, query answering capabilities, ownership, and access and privacy policies. Relationships may be stored as query transformations, dependency graphs, or sometimes even textual descriptions.

Wherever possible, the catalog should contain a basic inventory of the data elements at each participant: identifier, type, creation date and so forth. It can then support a basic browse capability over the combined inventory of all participants. While not a very scalable interface, it can at least be used to answer questions about the presence or absence of a data element, or determine which participants hold documents of a particular type. Simple scripts run over the participants can extend the capabilities of this interface. For example, computing and storing an MD5 hash of all data elements can help identify duplicated holdings between participants.

On top of the catalog, the DSSP should support a model-management environment that allows creating new relationships and manipulate existing ones (e.g., mapping composition and inversion, merging of schemas and creating unified views of multiple sources).

Search and Query: The component should offer the following capabilities:

(1) Query everything: Users should be able to query any data item regardless of its format or data model. Initially, the DSSP should support keyword queries on any participant. As we gain more information about a participant, we should be able to gradually support more sophisticated queries. The system should support graceful transition between keyword querying, browsing and structured querying. In particular, when answers are given to a keyword (or structured) query, additional query interfaces should be proposed that enable the user to refine the query.

(2) Structured query: Database-like queries should be supported on common interfaces (i.e., mediated schemas) that provide access to multiple sources, or can be posed on a specific data source (using its own schema) with the intention that answers will also be obtained from other sources (as in peer-data management systems). Queries can be posed in a variety of languages (and underlying data models) and should be reformulated into other data models and schemas as best possible, leveraging exact and approximate semantic mappings.

(3) Meta-data queries: The system should support a wide spectrum of meta-data queries. These include (a) including the source of an answer or how it was derived or computed, (b) providing timestamps on the data items that participated in the computation of an answer, (c) specifying which other data items in the dataspace may depend on a particular data item and being able to support hypothetical queries (i.e., *What would change if I removed data item X?*), and (d) querying the sources and degree of uncertainty about the answers.

A DSSP should also support queries locating data, where the answers are data sources rather than specific data items. For example, the system should be able to answer a query such as: *Where can I find data about IBM?*, or *What sources have a* salary *attribute?* Similarly, given an XML document, one should be able to query for XML documents with similar structures, and XML transformations that involve them. Finally, given a fragment of a schema or a web-service description, it should be possible to find similar ones in the dataspace.

(4) Monitoring: All of the above Search and Query services should also be supported in an incremental form that can be applied in real-time to streaming or modified data sources. Monitoring can be done either as a stateless process, in which data items are considered individually, or as a stateful process, where multiple data items are considered. For example, message filtering is a stateless process, whereas windowed aggregate computation is stateful. Complex event detection and alerting are additional functionalities that can be provided as part of an incremental monitoring service.

Local store and index: A DSSP will have a storage and indexing component for the following goals: (1) to create efficiently queryable associations between data objects in different participants, (2) to improve accesses to data sources that have limited access patterns, (3) to enable answering certain queries without accessing the actual data source, and (4) to support high availability and recovery.

The index needs to be highly adaptive to heterogeneous environments. It should take as input any token appearing in the dataspace and return the locations at which the token appears and the roles of each occurrence (e.g., a string in a text file, element in file path, a value in a database, element in a schema or tag in XML file). Two important aspects of the index are that (1) it identifies information *across* participants when certain tokens appear in multiple ones (in a sense, a generalization of join index). Typically, we may want to build special indexes for this purpose for a certain set of tokens, and (2) it is robust in the face of multiple references to real-world objects, e.g., different ways to refer to a company or person.

We may want to cache certain dataspace fragments (vertical or horizontal) for several purposes including: (1) to build additional indexes on them for supporting more efficient access, (2) to increase availability of data that is stored in participants that may not be reliable, and (3) to reduce the query load on participants that cannot allow ad-hoc external queries.

The Discovery Component: The goal of this component is to locate participants in a dataspace, create relationships between them, and help administrators to refine and tighten these relationships.

Location of participants can take several forms, such as starting a traversal from the root of a directory structure, or trying to locate all the databases on an enterprise network. The component should perform an initial classification according to participant type and content.

Once the participants are discovered, the system should provide an environment for semi-automatically creating relationships and improving and maintaining existing relationships between participants. This involves both finding which pairs of participants are likely to be related to each other, and then proposing relationships (e.g., schema mappings, replicas, containment relationships) that are then verified and refined by a human. Finally, it is important that the discovery component monitor the contents of the dataspace to propose additional relationships over time.

The Source Extension Component: Certain participants may lack significant data management functions. A participant might be no more than a departmental document repository, perhaps with no service other than weekly backups. A DSSP should be able to imbue such a participant with additional capabilities, such as a schema, a catalog, keyword search and update monitoring. Note that it may be necessary to provide these extensions "in-situ", as there can be existing applications or workflows that assume the current formats or directory structures.

This component also supports "value-added" information held by the DSSP, but not present in of the initial participants. Such information can include "lexical crosswalks" between vocabularies, translation tables for coded values, classifications and ratings of documents, and annotations or links attached dataset or document contents. Such information must be able to span participants. For example, in the desktop database a significant amount of effort is put into building associations between items in different applications (e.g., storing connections between presentations, papers and programs that all relate to the same project).

While we imagine that a "full service" DSSP contains all these components, we point out that many of them could be used on their own, to achieve certain price-benefit tradeoffs. For example, a large university initially may only be able to afford a Catalog and Browse service for the campus-wide dataspace, but that could be an improvement over the existing opaqueness of resources. Later, keyword query capabilities might be added, campus-wide or in selected sub-dataspaces. It is important that DSSPs can yield incremental payoff for incremental investment, and not exist only as monolithic solutions. Finally, though we do not describe these in detail, we expect a DSSP to have an administration component and some module that supports "soft" recovery.

4. Research Challenges

This section identifies some of the new challenges that arise in building DSSPs.

4.1 Data models and querying in DSSPs

Data modeling and basic querying: Unlike a DBMS, a DSSP needs to support multiple data models at its core so it accommodates as many types of participants as possible in a natural way.

The data models supported by a DSSP will fall into a hierarchy of expressive power. Every participant in the dataspace supports some data model and some query language appropriate for that model. For example, at the very top (most general) level of the hierarchy are collections of named resources, possibly with basic properties, such as size, creation date and type (e.g., JPEG image, MySQL database). "Query" against this data model corresponds to what a file system typically supports for its directories: name match, find in date range, sort by file size, and so forth. Below the top level, a DSSP should support the bag-of-words data model, implying that we should be able to pose keyword queries on any participant in the dataspace, and hence gain some visibility into the participants in a dataspace.

The semi-structured labeled-graph data model can come one level below the bag-of-words model in the hierarchy. Whenever a participant supports some structure, we should be able to pose simple path or containment queries, or possibly more complex queries based on the semi-structured data model. The goal should be that whenever there is a way of naturally interpreting a path query on a participant, the query processor should attempt to follow such an interpretation.

There will be other data models in the hierarchy, including the relational model, XML with schema, RDF, OWL (the Web Ontology Language). Given an environment, a key challenge is to find methods for interpreting queries in various languages on participants that support certain models. Specifically, how do we reformulate a query posed in a complex language on a source that supports a weaker data model, and conversely, how do we reformulate a query in a simple language on a source that supports a more expressive model and query language (e.g., keyword query on a relational database).

A broader view of querying: To adequately address the needs of dataspace application scenarios and users, a DSSP needs to support a broader approach to querying. Due mostly to the WWW and the ensuing revolution in how people can access information, people have recognized search to be a first-class activity. Computer users realize that a significant portion of their computer-aided activities can be divided into two parts: searching for relevant information, and acting on the found information. Search can come in many flavors, some reminiscent of database querying, such as finding flights for a trip, checking bank balances online, and others closer to keyword search, such as finding appropriate documents within an enterprise and looking for waffle recipes.

Hence, offering intuitive search and query on everything is a key challenge. In fact, from a user's perspective, the distinction between search and query should disappear. Users should start searching in the simplest way and then be directed as appropriate to more specialized search and query interfaces. The system should provide useful suggestions to the user as to what other searches or topics may be of interest given the query. Intuitive visualizations of results also need to be developed to guide users in the right directions.

4.2 Dataspace discovery

A crucial component to building a dataspace is to discover its participants and the relationships between them. A very common problem in today's large enterprises is that they don't even know which data sources they have throughout the organization. The ultimate goal of dataspace discovery is locate participants in the dataspace, create the relationships between them, and improve the fidelity of the existing relationships between participants. The main components of a dataspace discovery system are (1) locating the participants in the organization, (2) a semi-automatic tool for clustering and finding relationships between participants, and (3) a tool for creating more precise relationships between participants (at the extreme, these are schema mappings).

4.3 Reusing human attention

One of the key properties of dataspaces is that semantic integration evolves over time and only where needed. The most scarce resource available for semantic integration is human attention. Hence, it is crucial that DSSPs know how to reuse human attention, generalize from it, and reuse it for other tasks. The community has already developed methods for reusing human work in creating semantic mappings between data sources, but this capability is only a first step. Other examples of human work that can be reused include annotations (e.g., someone manually relating data items from two different sources), temporary collections of data that are created for a particular task (known as digital workspaces), queries written on the data (which imply certain relationships that may not be known otherwise), and operations on the data (e.g., cutting and pasting values from one column in a spreadsheet into a column in a different spreadsheet). The goal is that previous work should be recorded in the system and leveraged when we try to create additional relationships between participants in the dataspace or try to answer queries over it. We expect Machine Learning techniques to be useful here.

4.4 Dataspace storage and indexing

The key challenges involved in building the local store and indexing component of a DSSP have to do with the heterogeneity of the index. The index should uniformly index all possible data items, whether they are words appearing in text, values appearing in a database, or a schema element in one of the sources. In addition, the index needs to consider multiple ways of referring to the same real-world object. (Note that so far, research on reference reconciliation has focused on detecting when multiple references are about the same object).

Keeping the index up to date will be tricky, especially for participants that do not have mechanisms to notify it of updates. In addition, deciding which portions to cache in the local store and which indexes to build raises several interesting challenges in automated tuning.

4.5 Correctness guarantees

A core benefit of using a DSSP to access disparate data sources is the ability to do so with some confidence in the quality of the answers provided to queries and the effects and permanence of updates. Given the wide variance in administrative proximity and semantic integration (see Section 1.1) of the data sources in a dataspace, traditional DBMS guarantees for query answers and transactional updates will often be simply unobtainable. The research question then, is how to define realizable, practical, and meaningful levels of service guarantees that can be provided in a range of dataspaces. This challenge will require a rethinking of many fundamental data management principles, and the introduction of new abstractions. Tools to help designers and users understand the inherent tradeoffs in terms of quality, performance, and control will also be needed.

4.6 Theoretical foundations

There are several questions regarding the theoretical underpinnings of dataspaces. Clearly, there is need for a formal understanding of the different data models, relationships and answering queries in a dataspace. Digging deeper, in a traditional database theory, one of the main questions of interest is the expressive power of a query language. In the context of dataspaces, the analogous question would be the expressive power of a query language over a set of participants with certain properties on the relationships that are specified amongst them, i.e., what queries are expressible over a dataspace? Similarly, how can we detect semantically equivalent but syntactically different ways of answering queries?

5. Perspectives

To round out our discussion, we briefly discuss several important perspectives on dataspaces.

5.1 Relationship to Other Fields

Designing DSSPs builds on the traditional strengths our field and will involve significant extensions of data management techniques, but it will be crucial to leverage techniques from several other fields. We mention a few here. Recent developments in the field of knowledge representation (and the Semantic Web) offer two main benefits as we try to make sense of heterogeneous collections of data in a dataspace: simple but useful formalisms for representing ontologies, and the concept of URI (uniform resource identifiers) as a mechanism for referring to global constants on which there exists some agreement among multiple data providers. Similarly, as discussed earlier, several operations on dataspaces inherently involve some degree of uncertainty about the data, its lineage, correctness and completeness. The Uncertainty in AI Community had developed several formalisms for modeling uncertainty, but these tend to be very expressive. The challenge is to find models that are useful yet simple, understandable, and scalable.

Naturally, much of the data in a dataspace will be unstructured text. Hence, incorporating techniques from Information Retrieval will play a crucial role in building DSSP. Importantly, in a complex dataspace, users do not know exactly what they are looking for or how to interpret the results. Hence, it is important that they be able to effectively visualize results of searches and queries to better guide their exploration. Recent techniques from Information Visualization will be valuable here.

5.2 Teaching Dataspaces

An interesting litmus test for the concept of dataspaces is whether a course can be designed around it. Naturally, the foundations of dataspaces will evolve significantly as the research progresses, but we believe there is already sufficient material for a course. In addition to a review of basic data models and query languages, some of the topics that would be covered are: the challenges of heterogeneity and its different sources, architectures for data integration and data exchange, queries as a mechanism for data translation, algorithms for semi-automatic schema matching, different notions of QoS, supporting best-effort querying, and integrating structured and unstructured data querying. An important component of such a course would be to use and analyze successful examples of dataspaces (e.g., the Sloan Digital Sky Survey).

5.3 The Industrial Perspective

The concept of dataspaces is inspired in large part by challenges faced by industry today. In fact, there are many examples where industry is already making steps in this direction, but these steps are isolated from each other and there is clear need for a broader view that will yield a cleaner system abstraction and set of techniques.

For example, Enterprise Information Integration is starting to gain traction. The companies in this space are building systems to query multiple data sources within an organization. There are several examples of products that create indexes across multiple data sources for the purposes we mentioned above (e.g., Master Data Management, a component of NetWeaver of SAP). There are projects attempting to discover data sources within an enterprise, and there are quite a few companies looking at various aspects of enterprise meta-data management. Interestingly, the desktop search tools are also extending into the enterprise, coming from a completely different industry sector.

6. Conclusion

The most acute information management challenges within organizations today stem from the organizations' many diverse but often interrelated data sources. In this paper we have proposed the idea of dataspaces and the development of DataSpace Support Platforms (DSSP), as a means of addressing these challenges. DSSPs are intended to free application developers from having to continually re-implement basic data management functionality when dealing with complex, diverse, interrelated data sources, much in the same way that traditional DBMSs provide such leverage over structured relational databases. Unlike a DBMS, however, a DSSP does not assume complete control over the data in the dataspace. Instead, a DSSP allows the data to be managed by the participant systems, but provides a new set of services over the aggregate of the systems, while remaining sensitive to their requirements for autonomy.

Dataspaces can be seen as an umbrella for much of the research that is already being actively pursued in the database community; in fact this was one of our original goals. We have also, however, tried to outline several new research opportunities that arise from taking a more holistic view of emerging "data everywhere" challenges. These are challenges that the database research community is uniquely qualified to address, and we look forward to continued progress in extending the applicability of data management technology.

Acknowledgments

The authors were inspired (in different ways) by events at the CIDR 2005 Conference and the SIGMOD 2005 Post-PC Symposium to have the discussions that ultimately led to this document. Jennifer Widom played a key role in the original development of the dataspace concept. Serge Abiteboul, Phil Bernstein, Mike Carey, David DeWitt, Laura Haas, Zack Ives, Donald Kossmann, Mike Stonebraker, Dan Suciu, Jeff Ullman, Gerhard Weikum, and Stan Zdonik gave us useful advice (some of which we followed) on earlier instantiations of these ideas. We'd also like to thank the many colleagues who've read earlier (much longer) versions of this document.

References

[AAB+05] Serge Abiteboul, Rakesh Agrawal, Phil Bernstein, Mike Carey, Stefano Ceri, Bruce Croft, David DeWitt, Mike Franklin, Hector Garcia Molina, Dieter Gawlick, Jim Gray, Laura Haas, Alon Halevy, Joe Hellerstein, Yannis Ioannidis, Martin Kersten, Michael Pazzani, Mike Lesk, David Maier, Jeff Naughton, Hans Schek, Timos Sellis, Avi Silberschatz, Mike Stonebraker, Rick Snodgrass, Jeff Ullman, Gerhard Weikum, Jennifer Widom, and Stan Zdonik. The lowell database research self-assessment. *Commun. ACM*, 48(5):111–118, 2005.

[BBC+98] Phil Bernstein, Michael Brodie, Stefano Ceri, David DeWitt, Mike Franklin, Hector Garcia-Molina, Jim Gray, Jerry Held, Joe Hellerstein, H V Jagadish, Michael Lesk, Dave Maier, Jeff Naughton, Hamid Pirahesh, Mike Stonebraker, and Jeff Ullman. The asilomar report on database research. *ACM SIGMOD Record*, 27(4):74–80, 1998.