

Report on the Workshop on Information Integration

DRAFT

April 5, 2007

EXECUTIVE SUMMARY

Background and Scope

The type and modality of information available in digital form is vast: it comprises image, video, text, audio, sensor and other forms of streaming data, as well as structured data such as databases and XML/HTML documents. Although the data may be geographically distributed, and collected and designed for specific uses and applications, it is often logically inter-related, and many important questions can be answered only by accessing it collectively. The problem of *information integration* (II) is therefore to provide uniform access to multiple, heterogeneous sources of information and enable discoveries that would not be possible by examining the data sources individually. Despite considerable research that has been performed over the past 20 years, II remains expensive and laborious, and hence in need of large technological and methodological improvements.

On October 26–27, 2006, the Workshop on Information Integration (II Workshop) was held in Philadelphia. The workshop was sponsored by ANRQ, EPA, NARA, NSF and ONR, along with the National Coordination Office for NITRD. More than fifty invited experts in II participated in the workshop, representing a mix of relevant stakeholders, including government and industry researchers and practitioners, academic researchers, and domain experts representing genomics, neuroscience, medical informatics, litigation, and national archives.

The purpose of the II Workshop was to direct national attention to the need for addressing the many remaining major challenges in information integration and for investing in the required research, engineering and development that will be needed to improve the state of the art and state of the practice in the near- and long-terms. This report, a tangible outcome of the workshop, prioritizes recommendations regarding research, engineering and development challenges using a roadmap to determine what, when, and how priorities should be addressed over identified time frames.

Purposes and Format of Workshop

The workshop provided a working forum for leaders and visionaries from industry, research laboratories and government concerned with II. The chief goal was to develop a roadmap for overcoming crucial issues and challenges facing the design and use of II technologies. Participants were selected to represent the breadth of issues to be considered.

Of particular importance to this workshop were issues associated with the integration of scientific and engineering data as well as governmental, medical, and other records; the use of ontologies in integration; combining text with structured data; data augmentation; mappings; global integrity; data exchange and update reconciliation; schema or instance matching and ontology alignment; imprecision and uncertainty in data and inferences; reliable query answering in the presence of uncertainty; tracking the origins of data; and “hands-off” integration. Technical issues associated

with heterogeneous sensitivities, differential access, and mandated sharing in large collections were also discussed.

The workshop focused primarily on information management technologies. Although user interfaces and visualization techniques are extremely important to the success of II technology, the workshop did not address these issues. There are also many ties with the artificial intelligence community, natural language processing, and machine learning that were not addressed in depth.

The workshop included keynote talks, technical talks, and breakout sessions. Keynote talks were selected to highlight an application domain (genomics) as well as the current state of industry (BEA Systems). Technical talks were selected on the basis of two-page position papers. The breakout sessions addressed the following four issues critical to II:

1. **Matching.** How can integration at the instance level be enabled? Matching – the problem of finding identical, similar or related data items or data groups across two or more data collections – takes place both at the property level and at the entity level, and relies crucially on the selection of features. Some of the technical problems include the use of context in matching, the scalability and generality of matching, characterizing the robustness of a matching process, re-use of matching components, “best-effort” matching, managing and reasoning with uncertainty in matching, duplicate detection, named entity resolution, and data cleaning.
2. **Mapping:** How are the schemas of underlying information systems connected semantically, and to what extent can the process of mapping schemas be automated? Schema mappings are important in solving a number of integration problems, such as data exchange, reformulating queries on a target schema to queries on a source schema, and translating updates between data sources. Some of the scientific problems include creating mappings, debugging and understanding mappings, visualizing mappings graphically, mapping composition, and mapping inversion.
3. **Integration Architectures:** What is the spectrum of technologies and architectures for II? The use of an II architecture for a particular problem will be based on factors such as the cost of integration, how comprehensive the integration effort must be, what types of data sources are involved (e.g. streaming data and unstructured data), and the longevity of the integration product. Traditionally, II research has focused on “engineered” integration, which is carefully specified and validated. However, to consider a broader spectrum of II needs, issues associated with “ad-hoc” integration must also be addressed, including development environments, declarative specifications, component reuse, incremental solutions, provenance, and designing data sources for integration.
4. **Contextual Issues:** What is the impact of the context in which II is used? People who use II tools to generate integrated data sets must be domain experts, but must also have some understanding of data models and how integration tools operate on them; they may also need assistance on finding and assessing the quality and benefit of information sources. II is also affected by security and privacy concerns, which may dictate not only what form of integration may take place, but what explanations may be offered for the provenance of an

integration product. Related issues include the ownership of the integrated information, and economic incentives for integration.

Each working group was asked to summarize the state of the art, practice, development and research in its area, and to identify R&D needs and challenges, along with a roadmap to address the needs and challenges. This report details the findings of the workshop, including an Executive Summary and the four working group summaries. All presentations, including keynote talks, technical talks, impromptu “gong-show” talks, along with submitted position statements of the participants, are available on the workshop Web site: <http://db.cis.upenn.edu/iworkshop>

Findings from the workshop are summarized below.

Driving Problems

The need for information integration arises in virtually every domain, including, for example, meteorology, ecology, archaeology, law, government, medicine, biology and business. Mergers between corporations force integration of not just information but of information management systems, a niche area that companies such as SAP have addressed. However, it is frequently in the sciences that the limits of today's technology are being tested due to the rapid increase of data modalities that are linked to different experimental techniques, the vast amount of information generated by high-throughput technologies, and the diversity of groups involved, leading to a lack of common methodologies, technologies, and vocabularies. Another issue in science is the continuous evolution of world view, hence of information models – the moving-target problem.

The benefits of information integration are enormous, and carry huge societal impact. For example, a large scale disaster brings together a diversity of people and organizations, and requires the discovery, modeling, integration, prioritization and dissemination of multiple, potentially complex, heterogeneous and rapidly evolving sources of data. While disasters have much in common, individual disasters can differ significantly: Disasters may vary based on the demographics of the affected people and the affected area – local, such as a hurricane; national, such as the 2005 earthquake; regional, such as the 2004 tsunami; or global, such as the 2003 SARS outbreak and the predicted H5N1 avian flu pandemic. The time dimensions are also quite different: An earthquake strikes within a few seconds, while an epidemic such as the avian flu spreads much more slowly. The cause of the disaster can be man-made, such as terrorist attacks, or they can have natural causes, or both, such as the 2005 flooding of New Orleans. Despite these widely varying characteristics, disasters lend themselves as a scenario where the benefits of information integration can have a significant positive impact, both in saving lives and improving the many tasks surrounding disaster management. The need for information integration here is that it be fast, facile, account for real-time data, and be accurate.

In another domain, governmental and social awareness problems frequently require extensive information integration. For example, Circle of Blue is a global journalism, communications and information management project designed to address the critical problem of a diminishing supply of clean, affordable fresh water. One of the goals of the project is to integrate information relevant to fresh water, such as meteorological data, water table data, maps, legislation, events and political

issues, to propel water awareness into the mainstream human dialogue and catalyze a meaningful and sustained response to the present freshwater crisis. The need for information integration here is frequently less time urgent and variable than the preceding scenario.

Modern biology also requires the integration of information across species as well as across experimental platforms and modalities. From the integrated data, expensive computational inferences are made. In this case, integrated data is frequently warehoused, since speed and accuracy are highly valued and the manner in which data is integrated (i.e., the workflow) does not change rapidly. Problems of data warehousing include the freshness of data: When one of the underlying data sources is updated the warehouse may still contain the older data and the integration process may need to be replayed. Another problem is that warehouses are difficult to construct when there is only a query interface to the underlying data sources and no access to the full data. This problem arises occasionally within the context of genomic applications, and more frequently in the context of Internet applications such as integrated travel or classified web applications.

Thus the issues and challenges of information integration depend critically on the context or domain in which it is being performed.

Current State of Affairs and R&D needs

The problem of electronic information integration is by no means new, and dates at least to the late 1960's with the development of database and networking technologies. By the 1980's, the first federated database research project appeared, called Multibase, which used a functional data model. Complementary approaches were explored in projects in the U.S. and abroad. Prototypes of distributed relational databases such as R* and distributed INGRES were developed, allowing access through the same interface to multiple relational databases. Much of the work on distributed database technologies later found its way into relational database products, and the integration products that emerged focused on integration among data in the same model.

The problem of integrating multiple *non-relational* data sources received increased attention in the early 1990s, with projects such as Tsimmis, Kleisli and Garlic. The goal of these projects was to provide a mediated schema defined as a set of views over the data sources, which could then be queried as a single source. Queries over the mediated schema were automatically rewritten as queries over the local sources, and results were combined to provide an answer to the original query. By the mid 1990's, the approach pursued in these projects became known as the *Global-As-View* (GAV) approach, in contrast with the *Local-As-View* (LAV) approach, which was proposed in the context of the Information Manifold project. In LAV, contents of data sources are described as views over the mediated schema (the converse of GAV). The advantages of each were later combined in the *Global-Local-As-View* (GLAV) approach, and the influence of peer-to-peer file sharing systems were felt in novel architectures for data sharing, such as Piazza.

By the late 1990s, several influential research projects such as as Clio and Model Management began to move work on data integration from the lab into the commercial arena (Enterprise Information Integration systems). Several of the solved problems in mapping made their way into solutions as commercial products, e.g., IBM's Rational Data Architect, which helps automate the

mapping-creation task, and IBM's Websphere Information Integration, Microsoft's ADO.NET and BEA's AquaLogic, which include execution environments for using mappings in query reformulation. The emergence and use of XML for data sharing also had an impact on II, and its adoption was influenced by standards and technology development for schemas and query languages. More recently, Oracle and rivals such as IBM, Microsoft and BEA Systems have woven "federated identity" into their infrastructure software to service-oriented architectures (SOA) and Web services.

Driven by the critical nature of the problem, there has been a recent resurgence of interest in information integration. For example, in 2006 alone there were over five workshops held on this topic (IIWeb, WWW 2006; Workshop on Information Integration in Healthcare Applications, EDBT 2006, Semantic Information Integration on Knowledge Discovery - SIIK 2006, ICS-FORTH Information Integration Workshop 2006, and the Workshop on Exchange and Integration of Data - EID2006). The topic was also a major focus of several of the major database conferences, such as VLDB 2006, in terms of invited keynote talks and selected papers.

However, despite the considerable research and development over the last 20 years, truly ad hoc II, where disparate information systems are accessed efficiently in real time in response to unanticipated information needs and data is combined to form reliable answers to queries, remains an elusive goal. In response, the concepts of *data mashups* and *dataspaces* have been introduced as architectures for future research. A very simple example of a data mashup is HousingMaps.com, which displays available houses in an area by combining a listing from Craigslist.com with a display map from Google. The idea behind data mashups is that there may be no means of performing integrated access in a managed or principled fashion, and that the integration needs may be immediate and unanticipated. In contrast, dataspace approaches strive for integrated access to all of an enterprises information, in whatever form it may be. As a consequence of the global scope of such an undertaking, it may be permissible to provide only the simplest of capabilities over the dataspace, such as cataloging and keyword search, with more sophisticated services, such as structured query and format conversion added incrementally.

Ad hoc II, both for individuals as well as for enterprises, remains an important future goal.

R&D Challenges and New Research Directions

The problem of information integration introduces research and development challenges in several different areas of computer science, with associated needs in sociological, educational and community-building efforts. The nature of these challenges is discussed below, as well as specific research directions that should be pursued to address these challenges.

Designing systems to be integratable

In an increasingly networked world, data sources should be designed with integration in mind rather than as the disconnected silos they have been in the past, used solely by one organization or known groups of individuals. For example, data sources are much easier to integrate if they have "hooks" to the outside world by using standardized terms and ontologies, and including cross-references to relevant external objects. Similarly, it is helpful if the data source provides a standard

naming scheme for external reference. It is also useful to provide protocols for data exchange, by agreeing (for example) on standard exchange formats and content. Interfaces to data sources should also be designed with access by external programs in mind: Although GUIs that help formulate queries and visualize results are extremely useful to people, they also often restrict access by programs. It is often pointed out that integration by clicking on a hyperlink to join data objects does not scale to hundreds of results.

In databases, the idea of design has been formalized in terms of classes of constraints (e.g., functional dependencies) and normalized schema forms using those constraints. The goal is to minimize redundancy and associated update anomalies. In programming languages, the idea of “design patterns” has been introduced to address recurring design problems that arise in specific design situations. They provide a common vocabulary and understanding for design principles, and support the construction of software with defined properties. No such ideas currently exist for designing data sources to be amenable to integration. What are the general principles? What are the best practices? Developing theoretical as well as practical guidelines and publicizing these “best practices” would considerably simplify the downstream problem of integration.

Enabling matching

A crucial part of II is *matching*, that is, finding identical, similar, or related data items or data groups across two or more data collections. Although matching may take place at the schema level, and is therefore one aspect of mapping, matching also takes place at the instance level. Somewhat different techniques tend to be employed at the two levels (for example, while term alignment using string matching and metadata structure matching can be applied at the schema level, the instance level generally calls more for large-scale statistical methods to identify shared characteristics). Therefore an important line of research and development for matching is the creation of tools and techniques using both rule-based and learning-based approaches, as well as methods to accomplish their combination. Studies are needed to investigate the treatment of uncertainty or probability in matching, and to provide a foundation for well-founded evidence integration.

Since information external to the data itself (e.g., schema, application domain, accompanying documentation, or other meta-data) may also aid in matching, many of the techniques that have been successfully developed are quite domain specific. For example, matching is important for applications such as comparison shopping, mailing list de-duplication, and data cleaning. Techniques that are being developed within those problem areas are therefore applicable to II. Furthermore, there is a need to develop techniques to bridge between different types of data, e.g., structured data (e.g., relational databases) and unstructured data (e.g., text, music, images, and voice).

Matching tools and techniques can be used to develop customizable matching components that cover a range of uses. These components could then be cataloged and reused. However, what is the framework in which these matching components should be placed? Given a particular matching problem, what are the features of that problem that should be used to determine the applicable set of components? It is clear that the features will not only include information about the domain (e.g., addresses, Chinese names, or European names) but also characteristics such as the

available technology, cost, time, quality, and completeness, as well as the task and its requirements. More, and more powerful, matching techniques are essential to help with the mapping process.

Enabling mappings

Mappings between data sources specify the semantic relationships between models of data. Writing such mappings, and maintaining them as the underlying data sources evolve, requires both database expertise (to express the mapping in a formal language), and business knowledge (to understand the meaning of the schemas being mapped). It is therefore a very difficult task to create mappings, especially between sizeable data sources (so-called “industrial strength schema mapping”).

There are several lines of research and development to pursue to enable the creation of mappings. First, there is a need for standardized, high level mapping languages that can be implemented efficiently. Ideally, this language – or language suite – should unify what has been done in Enterprise Information Integration (EII) and Enterprise Application Integration (EAI), which seeks to hook up applications to talk to each other to enable workflows. Second, there is a need to help create (and maintain) mappings in these languages, using AI-based approaches to automatically infer mappings, techniques for minimizing or directing human effort, schema visualization and summarization strategies, as well as debugging techniques. Part of the solution may also be to develop general-purpose design principles and patterns.

Third, there is a need to find and reuse mappings as well as human attention. Much can be learned from experience, e.g., existing mappings to or from a particular data source, and mappings between similar data sources. However, it is not always easy to find these mappings or understand them once they are found unless high level, declarative languages are used. The idea behind “reusing human attention” is that every time a human interacts with a collection of data sources, they are indirectly giving semantic clues about the data or about relationships between data sources. These semantic clues, or partial mappings, should be captured and used to create larger mappings.

Another issue associated with mappings is that of updates. In relational and object-oriented databases, the problems of propagating updates from the source data to a view of that data, as well as from a view down to the source data, have been well studied. While some of these techniques are useful in the context of II (e.g., when an integration result—or mapping—is buffered or stored), there are interesting problems that arise from the matching or copying of data that occurs between data sources and the ownership of data. For example, although data is routinely copied between biological data sources such as GenBank and UniProt, each data source controls how data is updated within its system; updates cannot simply be forced to UniProt from Genbank. Despite the fact that data can be matched between data sources (e.g., due to copying), which creates inter-source integrity constraints, these constraints cannot automatically be enforced as updates occur. Each site must be able to know of relevant updates that have occurred, but be able to develop their own policies by which they will accept or reject these updates. Developing notions of update policies which take into account the preferences of data sources, as well as notions of consistency with respect to these update policies, is an important open problem.

Managing imprecision and incompleteness

Managing uncertainty in matching is another important line of research and development. We do not currently know how to or have the means to characterize the uncertainty in a matching process. Some matching techniques are also probabilistic rather than exact. Furthermore, having captured this uncertainty or probabilistic information, it must be propagated appropriately through query language operations to provide uncertainty or probabilities in answers to queries over integrated information. Thus developing appropriate models of uncertain and probabilistic information (a particular form of *annotation*) that are complete (i.e., can represent all finite probabilistic databases) and closed under the relational algebra, together with appropriate language constructs for querying imprecise and probabilistic data is an important first step.

It would also be useful to calculate the effect of acquiring additional information, or using additional machine or human resources, in reducing uncertainty in a particular matching or step of the query process. This would allow the II designer to decide what to focus on to give the best reliability or cost-benefit to particular families of queries over the integrated information.

There is also a need for methods to conveying uncertainty in data and integration results to human and software clients.

Tracking and using provenance

Derived from the Latin word *provenire*, “to come forth”, provenance is a record of how the integrated information was derived. Provenance is especially critical in the context of II since data may undergo transformations when exchanged and be combined with information from other sources.

Two general approaches to provenance have been explored over the past few years. Workflow provenance involves recording not only the data sets and the transformations that were performed, but also the details of external computations that were used. Since workflow provenance entails a large amount of annotation, it is frequently coarse grained at the level of data. In contrast, for database (algebraic) manipulations, a fine-grained recording of information and reasoning can be performed that focuses on individual data elements, or even small components of the data elements (e.g., provenance information on XML data can be attached at various levels of nesting). Open research questions include how to efficiently store provenance, minimizing the size of provenance information while supporting efficient queries on it, and the development of provenance query languages. Combining annotation associated with coarse- and fine-grained reasoning has also not been explored.

Provenance has several uses in the context of II, the most obvious of which is to help users understand the meaning of an integration result. Conversely, provenance can be used to understand the meaning of a mapping between two data sources: For example, a schema mapping can be illustrated through the relationship between source data and restructured data. It would be useful if one could also reflect the changes that would be applied on the source data and restructured data when a schema mapping is modified. Such scenarios happen often during a debugging session, where the schema mapping may be modified. Understanding a schema mapping can also be achieved through “cause-and effect”. With a fixed schema mapping, one may change the contents

of the data sources, or even the restructured data, during debugging so that the cause-and-effect of those changes can help one understand the schema mapping. These problems correspond to the classical *view adaptation*, *view maintenance* and *view update* problems in existing literature and should be re-examined in the context of the data exchange framework.

Provenance corresponds to one form of annotation on data, and in this sense is closely related to incomplete and probabilistic databases in which the annotation represents a probability or condition under which the information is included. This connection needs to be explored and better understood.

Security and privacy

Information integration raises security and privacy issues that are currently dealt with in an ad hoc manner, if at all. For example, individual data sources may have privacy requirements that could be violated during II (data leakage). Connecting information between different data sources may also circumvent privacy requirements, such as arise in electronic health information with HIPPA. A first step towards managing privacy concerns in the context of II would be the ability to express one or more privacy policies in such a way that they can be automatically enforced, or at least checked, within particular phases of II. Within the matching phase, for example, such policies might prevent true matches from occurring even when they can be made. Such policies could be stated on both individual data sources as well as on the integration result. A related problem – and one that is not well understood – is how to propagate individual privacy requirements to an integration result.

Another approach to managing privacy and security concerns would be a trusted third party. The third party could aid in selected integration steps to shield sensitive elements from the originating parties. Such a service might rely on both the presence and absence of information in doing its work.

Complying with security and privacy policies might require logging enough information to make the processes auditable, and passing through annotation (e.g., provenance, uncertainty and probabilistic information) that pertains to privacy from the original information sources to the integration results. For example, suppose that as a result of some II connecting political figures with newspaper accounts, an individual was (incorrectly) associated with a past crime, leading to a potential case of libel. It would be important to be able to determine why that association was made (e.g., the individuals were different but had similar names), where the information components came from (so blame could be appropriately allocated), whether or not the information components were probabilistic, and what type of matching was used and what the certainty was. Demonstrating that this information was passed to consumers of the II product so that the consumers were aware of the uncertainty of the result would also be important. While it may be impossible to certify that stated privacy policies are being met within a particular integration, determining the minimal amount of required annotation to meet auditing requirements might be achievable. This information could also be used to debug an integration with respect to a privacy requirement, and iteratively refine the integration until policy thresholds are met.

On the other hand, security and privacy may also be used to *block* explanations and provenance. Aggregation of data is often a mechanism for ensuring privacy, which would be violated by giving details of individual elements of the aggregation.

Resources and infrastructure

For each of the challenges discussed above, there are associated needs for education, metrics for evaluation, corpora for testing, and pooling of best practices.

For example, the effectiveness of matching and mapping techniques that are being developed is hard to judge, since there is no common corpus for testing, nor any agreed-upon metrics. We need to develop test collections as well as synthetic data on which these techniques can be tested and metrics inferred.

However, there is a limit to how much can be learned from small or synthetic data sources, but larger collections of real data (such as patient records, customer accounts or search logs) raise privacy and business issues as seen in the recent publication (and withdrawal) of web-search data by AOL Labs. A possible alternative would be to have a trusted third party site control a copy of sensitive data, and have them measure and test submitted software. An obvious issue is how to report problems (e.g., false positives and negatives) back to the submitters without compromising privacy.

Finding, assessing and understanding what data sources, ontologies, and mappings are available for performing II is also a difficult problem. While designing data sources to be integrable is part of the solution, it would also be helpful to develop “Information Yellow pages” in a variety of different domains that would collect and organize information about data sources and mappings in each domain. The development of a Yellowpages would be enabled by developing standardized ways of documenting and reporting integration of information from a source, and has implications for annotation and provenance. These issues are currently being considered in the context of web services: For example, the Web Services Description Language (WSDL) specifies a way to describe the abstract functionalities of a service and, concretely, how and where to invoke it. Semantic Annotations for WSDL ([SAWSDL](#)) provides mechanisms by which concepts from the semantic models that are defined either within or outside the WSDL document can be referenced from within WSDL components as annotations; these semantics can then be used to disambiguate the description of Web services during automatic discovery and composition of the Web services. Generalizing these ideas to more general II scenarios would be useful.

Having found the appropriate data sources, ontologies and mappings to use for a particular II problem, there are a variety of different architectures and approaches that could be used for its solution. However, we do not understand how to characterize the cost of integration, which is determined not only by the inherent complexity of the integration domain, but by how comprehensive the integration effort is to be. As a first step, it would be useful to develop a toolkit for each step of II – e.g., matching, mapping – along with metrics for their usefulness indexed by the context in which the II is to be performed.

Research Roadmap

Achieving this grand agenda will require research effort on the part of the database, information retrieval, machine learning and other computer science communities, engineering effort on the part of domains in which II will occur (e.g. genomics, medical informatics, geosciences, etc.), and coordination and support from industrial and government collaborators. Most importantly, it will require planning and support from the government agencies. Here we offer a sequence for the most important components of a research roadmap that addresses the research challenges identified earlier, broken into Infrastructure Achievements and Research Achievements.

Three-Year Roadmap:

The initial step in the roadmap is to gather experiences, tools, and testbeds for experimentation, and obtain initial research results in provenance, privacy and security, and imprecision and incompleteness.

Infrastructure Achievements:

- Matching testbed: Design of a standardized, nationally available corpus for testing matching technologies. Establishment of open competitions at conferences related to II, similar to NIST's TREC series of Information Retrieval competitions.
- Creation of "summer camps" that are focused on targeted sets of matching problems to develop experience on which techniques work and which do not work. The summer camps could also be used to enable cross-disciplinary linkages in the context of specific problems.
- Creation of web sites that act as centralized resources for matching and mapping technologies.

Research Achievements:

- Lightweight, best-effort matching: Development of better generic matching techniques and associated prototypes that can be used in lightweight integration scenarios.
- Design of models of provenance and annotation, in particular for retaining human-created annotations when manually matching or linking information items.
- Design of models of incompleteness, imprecision, and matching probability; understanding of their closure properties under mapping languages.
- Development of models and languages for expressing privacy policies that are amenable to verification with respect to mappings.

Five-Year Roadmap:

The second step in the roadmap is to develop standards and prototypes, as well as metrics for assessing the quality of results.

Infrastructure Achievements:

- Compendium of best practices with illustrative case studies.
- Syllabi developed for II, both as a section of a course and as a full course itself. Topics should include basic techniques for information integration (e.g., federation vs

consolidation), as well as best practices in matching, schema mapping languages, and design templates.

- Identification of “trusted third party” sites that could act as mini-integrators of sensitive information and produce information that ensures stated privacy guarantees.
- Development of Masters or training programs to produce “information integrators” skilled in using appropriate tools and expert in one or more application domains.
- Reference texts on mapping, matching, basic techniques for II, and guidance on when to use which technique; survey articles on provenance and security as it relates to II (such articles already exist for mapping and matching).
- Articulated “Grand Challenges” for developing experience in how well II technologies work.

Research Achievements:

- Identification of classes of features that tend to be important in matching, and their use in statistical, machine-learning and rule-based techniques to generate matching routines for specific domains and tasks. Investigation of probabilistic matching evidence combination.
- Characterizing incompleteness or uncertainty in matching processes (metrics), and propagating incompleteness through mapping languages.
- Development of metrics for assessing the quality and privacy guarantees of integrated information.
- Development of prototype provenance reasoning systems and data reduction techniques.
- Development of models for predicting the best combination of technologies to use for particular integration tasks, and embodiment of these models in tools to assist in integration.
- Development of techniques that enable research subsystems for matching, mapping, query answering, provenance, etc. to be hooked together (“end-to-end” integration).

Ten-Year Roadmap:

The third step in the roadmap is to provide “sandboxes” for experimenting with different II architectures, and transition research results to industrial products.

Infrastructure Achievements:

- Creation of a “Data Integration Curriculum” as a standard topic in computer science and for domain experts. The curriculum would embrace both integration (mapping and matching) tools as well as design principles for data sets that will be used by these tools.
- Development of a check list or audit procedure to assess how integrable a data source is.
- Standardized ways of documenting and reporting how information integration has been performed.
- Third-party escrow of sample data: techniques for making large collections of real data available for testing II techniques in a secure way (e.g. trusted third party control).

Research Achievements:

- Standardization of a common mapping specification language.
- Cost-based and policy-driven information integration in response to a non-procedural specification of the requirements of the task, with minimal need for additional human input.
- Calculating effect of gaining additional information to reduce uncertainty.
- Mature capabilities for annotation and provenance.

- Evaluation of II technologies using Grand Challenge metrics.
- Mature techniques for light-weight, ad-hoc integration.

The remaining sections of this report present the findings of the individual breakout groups on the following topics: Information Integration Taxonomy; Matching; Mapping; Contextual Issues; and Information Integration Systems.

Information Integration Taxonomy

Participants

Phil Bernstein, Howard Bilofsky, Michael Carey, Barbara Eckman, Todd Hughes, H. V. Jagadish (reporter), Anupam Joshi, Hilmar Lapp, Tom Lee (reporter), Bertram Ludaescher, Louiqa Raschid, and Arnie Rosenthal

Introduction

The purpose of this panel was to look at alternative architectures for data integration. What does information integration mean? What are the different ways in which people may attempt to do it? What are the tradeoffs between the options? The panel defined multiple axes of interest, and developed a taxonomy of information integration based on these axes.

This panel also looked at the big picture system issues for information integration, both in terms of system architecture and technology, and also in terms of the surrounding legal, social, and economic issues. The results from those deliberations are included in the last section (Information Integration Systems) of this report.

Technical Taxonomy

Information integration is required for many applications in many different contexts. To place these in perspective, we make an attempt here to characterize the spectrum of choices, and identify the issues and challenges specific to each.

Engineered versus *Ad Hoc*

Perhaps the most crucial distinction is between engineered integration and *ad hoc* (or exploratory) integration. In the former, we will have a carefully specified integration process. Usually, the integration process itself, if not the actual results of integration, are validated. In the latter, the integration is performed in a best-effort fashion when requested. Attributes that are typical of engineered mappings is that they are stable (or evolve slowly), developed up-front, and based on well-understood integration criteria. In contrast, *ad hoc* integration could be (but need not be) fast-changing, have little up-front development, or have vaguely understood objectives (which may be refined in an iterative fashion, using a “spiral” development methodology).

Traditionally, data integration research has focused on engineered integration, and this continues to remain important. However, there is new and growing interest in *ad hoc* integration, as suggested in the plenary talks at this workshop by David Maier (on-demand integration) and Bob Grossman (universal key).

Note that the actions performed on the results of the integration have nothing to do with how the integration is done. For example, a scientist may perform exploratory analyses on or issue *ad hoc* queries against an engineered integration of data from multiple sources. Similarly, the result of an exploratory integration process could be captured in a workflow that could be repeated at the push of a button for future similar needs.

There are several other dimensions of integration that have some correlation with this dimension (engineered vs. *ad hoc*). However, they are actually distinct axes, as we will illustrate through examples that bely this correlation.

Longevity: Long-lived versus one-time

Sometimes data integration is a one-shot process. We perform the integration once, use the result and move on. More often, the integration continues to remain of interest, and such long-lived integrations raise additional issues. If the result of integration is cached as data then it may be important to propagate updates of the source data to the cache. If the integration result can be defined as a view over the data sources, then the maintenance of a correct integrated result is simply that of incremental materialized view maintenance. However, information integration can sometimes involve complex matching and mapping functions, whose incremental maintenance can be considerably more complex than incremental maintenance of algebraic views.

Sometimes, the integration could be virtual in that there is no physically integrated data set, but rather just an integration process. Every query against the integrated data is suitably mapped and causes appropriate data to be retrieved from the sources and merged. The former style is typified by the data warehouse ETL (extract, transform, and load). The latter style is typified by enterprise information integration (EII). We note that in this case, usually, the queries are engineered, but the integration is dynamic. Even if no data result is cached, an integration can be long-lived because integration artifacts, such as mappings, are cached. If source schemas change over time, or if the details of the desired integration specification change, then the retained integration artifacts have to be updated suitably. Biologists, for example, might repeatedly use different queries over the same mapping specifications on different underlying data sources. Here it is the mapping specifications that are long-lived.

Typically, we might expect engineered mappings to correspond to long-lived integration. However, exceptions do exist. For example, in a legal/audit scenario, a firm might design a specification and engineer a mapping to satisfy a single round of litigation, never to be used again. This would be an engineered integration for a one-time use.

Deep versus Superficial integration

At one end of the spectrum, integration can mean assembling a simple union of queries applied to multiple sources, as some simple meta-search engines may do. At the other end, there could be a deep understanding of the semantics of the data being integrated, resulting in a true synthesis of information from multiple sources. While the quality of the integrated product may be superior at the latter end of the spectrum, the effort required is also likely to be much greater. There are many potentially useful trade-off points between integration effort and integration depth. There is ample

scope for research that moves this tradeoff curve, by decreasing effort for a given depth or increasing depth for a given effort level. One common such point, closer to the superficial end of the spectrum, is where only the main identifiers of objects of interest are matched, and the remaining data is simply unioned.

Homogeneous versus Heterogeneous Domains

A continuum can be defined over the similarity of the data sources being integrated. Sometimes, the multiple data sources being integrated all have data over the same domain (or similar domains), even if the exact sets of attributes do not match. We say such integration is over homogeneous domains. On the other hand, we could have multiple dissimilar data sources with data from different domains. In such a case our integration is over heterogeneous domains. The steps performed in the two types of integration are likely to differ considerably. We also note that these two types are not discrete, but are rather end-points in a spectrum.

Data Types

Traditional information integration has focused on structured alphanumeric data. However, other data types can be of importance, and can raise challenges specific to the type. Examples of such types include text, multimedia, data streams, event streams (messages, data value changes, ...), and so on. We also have a need for process integration, and this should be part of the information integration domain, just as much as data integration.

Precise versus Approximate Integration

There are many sources of uncertainty: we could have uncertainty in the data itself, in the specifications (as in a “keyword” query) leading to uncertain requirements, or in the mappings. Irrespective of the source of uncertainty, if it is present it has to be managed. Traditional integration is precise, and is appropriate where uncertainty is low.

The above collection of dimensions serves as a taxonomy of information integration. Every information integration problem can be evaluated and characterized in terms of these different axes. The nature of admissible solutions, and their cost, is determined by where a particular information integration problem lies with respect to these dimensions.

Several of these dimensions are frequently conflated in everyday discourse. Yet, except as noted, we were able to find examples of integration problems for every combination of values along the different axes.

Economic Dimensions

In addition to the technical dimensions discussed above, there are two additional, non-technical, dimensions of importance.

First, from a user perspective, a characterization of the cost of integration is crucial. One major determinant of this is the inherent complexity of the integration domain. For example, it is straightforward to perform a join on a consistently and completely populated identifier column. In contrast, it is challenging to determine identity using an expensive match function that takes a multiplicity of attributes into account. Similarly, mappings that require transformations between data and metadata tend to can be expensive to find and to apply. Note that the complexity of the integration domain has nothing to do with data set size. Our characterization of complexity is with regard to just two objects, one from each source. The number of sources involved, and the number of objects in each source, multiplicatively increase the integration cost.

A second major determinant of cost is how comprehensive the integration effort is. To be able to get a first result quickly and provide quick return on investment, an incremental approach may be preferred, in which only some sources are integrated first. Other sources of interest can be integrated later, if at all. *Ad hoc* integration may also provide faster initial results, with iteration on the quality of result obtained rather than on the number of sources integrated.

Matching

Participants

Dave Maier (lead), Sergey Chaudhuri, Susan Davidson, Bob Grossman, Sylvia Spengler, Eduard Hovy, Doug Oard, Sally Howe, Chris Chute, Craig Knoblock, Dave Archer (scribe).

Nature of the Problem

Most generally, *matching* in the context of information integration is finding identical, similar or related data items or data groups across two or more data collections. Matching can take place at multiple levels of abstraction:

- Schema to schema
- Schema to ontology
- Data item to schema entity or ontology concept
- Data item to data item

This section deals chiefly with the last category, matching of data instances (though recognizing that schema and ontology matching are instance matching with meta-data).

We see data matching taking place both at the property level and at the entity level, with the former being the basis for accomplishing the latter. Property-level matching includes matching of single domain values (strings, numbers, dates, names), matching of contents of larger elements such as documents or images (generally involving extracting features, such as term frequencies, color histograms, frequency components), and matching of compound values (compose of several simpler values). Any of these kinds of matching might benefit from the context of a value. For example, the context of a name extracted from a web page might help determine if it is a company name or person name. [better example]. Ultimately, matching is usually trying to connect conceptual entities or objects: persons, events, genes, products. The output of a particular matching process might be just a Boolean yes-no, or a best-match (or top-k) match candidate, or a probability or similarity measure.

For any particular kind of match, there may be alternative procedures or operators for carrying it out.

What an appropriate technique for matching a given variety of property or entity is specific to the task or application need the match. An important facet of applying matching techniques is going from why you are matching to how you should be matching for that purpose. There are also implications going in the other direction. Limitations on matching methods, based on technology, cost or policy, can feed back to indicate places that the task or application will need to be reconsidered or modified.

It is helpful in discussing data matching to understand its potential uses in the context of information integration. Among the possible uses are

- Collation: Simply collecting similar items, possibly for further manipulation by manual or automated processes.
- Linking or annotations: Creating connections between matched items, or adding to a matched item, while keeping the items intact.
- Consolidation or removing duplicates: Coalescing or discarding matching items in order to end up with a single representative of a given entity or object.
- Ranking: Ordering items based on output of a matching process.
- Correction or update: Making changes to matched items in their original sources.

The adequacy of a matching process clearly depends on which of these uses it is intended to support.

Common Threads

Before we embark on a discussion of what currently is and isn't possible, and what might be possible in the future and how to achieve it, we call out certain topics threads that cut across these considerations.

The context of the target task or end application is critical in selecting what sources and items to match, how to match them and what are appropriate thresholds or parameters for the matching process. That context further helps in understanding the importance, consequences and criticality of matching performance in a particular situation. While people can certainly talk intelligently about the needs of a task or application, we are not aware of a rigorous framework for extracting and expressing salient aspects from an application context that is suitable for mechanical analysis and interpretation.

Examples abound of successful domain- and type-specific matching capabilities constructed in a one-off fashion. Comparison shopping sites rely on effective matching of product descriptions; mailing list de-duplication requires recognizing equivalent person-address pairs. Similar capabilities have been developed for particular flavors of scientific data. We expect some of these systems can be extracted in a reusable way and tailored for several similar uses. Perhaps some can be generalized into customizable components covering a range uses. However, we feel a generic, all-purpose matching module is not a foreseeable development.

There will always be situations where matching cannot be completely automated and human analysis and judgment will be required. Human attention is almost always the resource in shortest supply, so it is important to use it effectively. One aspect of effectiveness is reusing human effort that has already been expended. Too often, manual matching happens on data extracted from its original sources and manipulated in some local form, such as a spreadsheet. The expertise expended is unavailable for similar tasks in the future; most data sources aren't prepared to accept and store information about the connections of their entities to those in external sources. A second aspect of effective use of human attention is direction: If there is a limited amount of time to invest, where would it yield the most benefit. Directing attention requires some way to identify "problem spots" in a matching activity, and also an estimate of the potential gain if a problem is resolved or ameliorated.

Matching does not always take place at the same level of granularity in a data item. In some cases, it may be sufficient to identify equivalent items at the aggregate level. (For example, this record and that record talk about the same company.) In other cases, there is a need to align subcomponents. (For example, chromosome, locus and exons of a gene.)

Assessing what current capabilities are, and tracking what progress the field is making, will require some collection of generally accepted metrics. The challenge is the range of characteristics to qualify: cost, speed, quality, completeness, scale, usability and probably others. As metrics are proposed, their effect on task performance should be verified.

The previous section discussed lightweight versus heavyweight integration, and other axes of integration that are closely or loosely correlated with that dimension. These axes of integration show up in various forms in matching. Does an application require that matching be performed online, or in near-online time, or can it be an extended batch process? Must it be configured rapidly, or is there time for extended training and analysis. Does the matching process have unlimited access to the data sources being matched, or is it limited to specific interfaces or limited result sizes? Is it required to be an automated process, or is significant human assistance available? Understanding the area of matching requires better characterization of which techniques are best at which ends of each spectrum, and an articulation of the necessary tradeoffs between axes (such are time versus precision).

What Can We Do Well?

There are aspects and sub-problems of matching where there has been significant progress to date.

Database management systems can perform joins—matching records from two collections based on one or more component values—efficiently and in a highly scalable manner. However, the match conditions supported by the fastest join algorithms are limited: straight equality on domains that can be indexed, hashed or sorted, or on comparison conditions that are amenable to one of these access methods. Similarly, matching of a data element against a pre-supplied dictionary of values is feasible as long as equivalence only encompasses minor syntactic variations.

String-matching techniques are also well developed for a range of similarity criteria: exact match, longest common subsequence, and edit distance, to name a few. These techniques can take into account a priori symbol-level difference measures (for example, to reflect characters that are often transposed on a keyboard or proteins with similar codons). Large-scale approximate matching of a pattern string against a large sequence database has sub-linear techniques based on indexing. However, such methods generally depend on a priori probabilities and assumptions in order to estimate the significance of a given match. There are indications that as the databases are being filled in, some of these assumptions may no longer hold, such as the independence assumption for BLAST. Hence it may be time to re-validate likelihood estimations for such techniques. Also multi-string matching, such as is required for multiple alignments of biological sequences, is intractable for optimal solutions beyond a few dozen sequences.

Probabilistic methods and machine-learning techniques have also been effective in some domains, but may require significant amounts of training data.

Matching of data arising in different enterprises or communities often benefits from a reference controlled vocabulary or ontology for normalizing terms. Ontology-authoring tools have reached a reasonable degree of maturity. However, automated matching of a data source to an ontology, or of one ontology to another, is still an active area of research. Once a “crosswalk” between vocabularies or ontologies has been constructed, though, its use in matching is generally straightforward.

In the business domain, it is certainly common practice to build data warehouses within a single enterprise, and there are commercial offerings aimed at this task. The integration process can require heavy investment of personnel time, however. Constructing such a warehouse of course involves matching, and matching tools are available for specific domains (e.g., Soundex encoding for names). Integration solutions offered for scientific data appear to mainly offer federated access to multiple datasets, but without any specialized matching capabilities.

What Can't We Do Well?

The fundamental problem of matching is to determine whether items A and B are identical. However, there seems little likelihood of a general method of doing so, as what it means for them to be identical is unavoidably bound to the particular task at hand. At this point, we do not have good means to express precisely the requirements on identity on a particular task (even when the semantics of the data is well understood), in a way that they might be automatically implemented by a matching system, or even check the appropriateness of a matching method that was configured manually. A clear description of matching requirements would express whether the task is dealing with identity, similarity, proximity or some looser kind of association. It could also help decide the effectiveness of different attribute choices for matching where there are multiple options. (Which would be the most effective joins?)

A good theory of matching does not yet exist that copes with intra-domain differences in what an entity is. (For example, is a gene a chromosomal sequence, a particular transcription and splicing of that sequence, a product translated from such a transcript, or something else?) Instance matching is often conflated with some more-aggregated-level concept that needs to be connected. The way ahead may be to have very tight definitions of what constitutes a match at the component-entity level, combined with liberal linking rules for the higher-level concept.

There can be variations on how information should be matched and aggregated exist within single projects or organizations. There is only beginning to be work on software architectures that can support multiple, simultaneous matching over the same base data. Also, it is not clear that integration projects currently “learn from experience”—that the second time that a source is integrated is any quicker than the first time. A related issue is whether the n^{th} source can be integrated into a collection any more easily than the first source, that is, achieving sub-linear behavior in the number of sources added in terms of integration effort.

Many applications still require that some or all matching be done manually. However, there is generally not good support for capturing human attention in a way that it can be reused. For example, manual matching often proceeds by extracting elements from two or more data sources into a separate spreadsheet, where alignments between elements are made. However, now the information on element equivalence is isolated from the original sources, and not accessible to anyone using them. Most sources do not provide for accepting information about their connections to other sources, and there generally is not any separate repository for maintaining matching judgments.

Our ability to manage uncertainty in matching processes is limited. We currently do not have means to characterize the uncertainty in a matching process in general or of two instances in specific. Further, some matching techniques may be probabilistic, rather than exact. We do not do well at communicating such uncertainty or probabilities to users. We would like to predict the effect on acquiring additional information, or employing additional (machine or human) resources, would have on reducing uncertainty in a particular case, in order to make trade-off decisions on where to focus. However, any such decisions will also depend on understanding the consequences of the current level of uncertainty in a given application. It might be critical to reduce uncertainty about whether two patients are the same person, but less so to determine which of two brands of the same drug they were prescribed. In addition to deciding *where* to focus (which data instances or subsets), there is not much guidance *what* to focus on—what aspects of a given matching problem give the best evidence for reliability and effectiveness.

Most matching approaches appear to target one kind of data, structured or unstructured. There is limited capability to match between structured and unstructured sources, apart from throwing away structure (e.g., treating a database relation as a set of keywords) or building routines to try to impute structure to an unstructured source (e.g., information extraction). Unstructured sources should be understood here not to be just text documents, but also music, images, voice and so forth.

Information integration in general, and matching in particular, raise security and privacy issues that at the moment are generally dealt with in an ad hoc manner, if at all. Connecting information may circumvent privacy requirements, such as arise in electronic health information with HIPAA. We would like the ability to express one or more privacy policies that govern matching in a way that they can be automatically enforced (or at least checked). Such policies might need to prevent true matches, even when they can be made. Complying with policies might also require logging enough information to make the matching process auditable, and passing through provenance and annotations from original information sources that pertain to privacy directives of individuals.

We observed in the previous section that businesses do routinely build integrated data warehouses, which nearly always entails matching on at least some of the attributes involved. However, such systems almost always involve replicating data and assembling it in one place. Such replication allows modification of copies for data cleaning and to add additional linking or disambiguation information. Matching and integration “in place” is not commonly realized. Also, the existing technology to support warehouse construction do not always keep up with the burgeoning amount of data and number of formats, sometimes leaving information behind in the original data sources. For scientific endeavors, such technology can also be too expensive and unable to cope with the moving target as domain models and data representations evolve. There needs to be complimentary

technology for such uses, where it can be more important to quickly and cheaply do a rough match of data for a quick browse, rather than obtaining fully resolved matching for a full integration.

What Could We Do Better? (or Why Can't We Declare Victory?)

Some of the deficiencies listed in the previous section may be inevitable—problems that are just too hard or ambiguous to expect effective, automated solutions. In this section we call out areas where we believe progress is possible, given research that is underway today or might be anticipated in the coming years. We first comment on limitations of current solutions that we do not think are inherent—that is, where there is a reasonable expectation that progress can be made.

- Current matching technologies seem very domain-specific and brittle. They do not generalize easily. Lightweight integration scenarios, at least, should permit more-generic solutions. For scenarios with stronger requirements, we do not have a good way to characterize when a solution developed in another domain is appropriate to apply.
- Matching solutions do not always scale with the number of instances, attributes and sources.
- It is hard to judge progress in the area at the moment, as there is no common corpus for testing, nor any agreed-upon metrics.
- The process for identifying features that can be used for matching decisions is not well understood. Evidence for equivalence comes from diverse sources, and it might need a wide variety of statistical, machine-learning and rule-based techniques to interpret.

The matching group briefly touched on areas where there is no apparent hope. It does not seem feasible to convert all information to structured form prior to matching – the cost is too expensive. We also do not expect to see the day when every information source (or even most of them) has complete semantic markup.

For areas where progress seems feasible, we categorized them by where on the timeline they seem obtainable.

Two-Year Roadmap:

- Construct Giigel (Generic Information InteGration Engine – Lite): A lightweight, best-effort matching system that operates with minimal configuration.
- Develop a framework for retaining human-created annotations when manually matching or linking information items.
- Make progress on particular matching problems, possibly through intensive “summer camps” focused on a targeted set of problems.
- Produce a set of test problems for matching techniques, in order to evaluate progress. Such a corpus should be accompanied by a range of metrics, for accuracy, performance and other relevant criteria.

Five-Year Roadmap: :

- Figure out how to harness contributions to matching from a community (e.g., domain scientists) while being able to regulate quality and ensure appropriate trust in the results.
- Build an integration center, similar to current supercomputer centers, where scientists and others can bring their data and have access to a wide array of matching and integration tools.
- Adapt and develop machine-learning techniques that generate matching routines for a specific domain and task.

R & D Challenges and Hard Problems

The matching group briefly discussed possible “Grand Challenge” problems. However, it seems that such problems would not be for matching in isolation, but rather for information integration more generally.

Some seeds for challenge problems:

- Insurance risk estimation for properties in advance of an impending natural disaster (hurricane, flood) combining weather information and specifics on other geolocated information (flood plains, building types).
- FEMA scenario construction based on storm surge and other predictive data, existing resource data (e.g., private generator availability), survivability data for structures.
- Personal health record construction from sources at multiple providers, supporting different views of that record for use with different care providers.

Specific hard problems identified by the group:

- Purging all information on an entity from an enterprise, which might be required for privacy or legal reasons. Not only will a wide variety of matching techniques be required, but we might not even be aware of all the sources to match against. This problem is a dual to the usual matching situation, where entities are being matched between a few sources. Here there is one entity, and matching happens across a potentially large number of sources.
- Defining appropriate inference limits for users. There may be cases where matching is possible, but should be suppressed for privacy, security or business reasons. Expressing policies for limits on matching in an understandable and efficiently enforceable way is an open problem.
- Characterizing robustness of matching processes. We can test a matching technique on particular sources to get a single-point measure of accuracy, precision and efficiency. However, we do not have good protocols for assessing how matching performance degrades in the presence of confounding factors, such as errors or increasing numbers of similar but non-identical items.
- Dealing with intentional interference, either to mask true matches, or to create incorrect matches. Are there any independent, systematic “sanity checks” on matching to reveal possible gaming of the process? Such checks might be holistic, such as overall matching

patterns that are inconsistent with known statistics, or specific, such as a matched item being of questionable authenticity (for example, a user account with a non-existent zip code).

Research Approaches & Road Map

Reward being integrated: Matching techniques can go only so far. Some integration tasks will have limited room for improvement unless information sources are improved. The research tasks here are to devise ways of rewarding information authors when their information is integrated, and to promulgate guidelines to source builders about how to make their information more easily integrated. A start to the former task is to develop a standardized way of documenting and reporting integration of information from a source, which will likely require mature capabilities for annotation and provenance. The latter task might commence with a survey of domains where information is being successfully integrated to extract the salient characteristics that enable integration.

Third-party escrow of sample data: There is a limit to how much can be learned about matching from small test collections or synthetic data. However, making large collections of real data (such as patient records, customer accounts or search logs) will undoubtedly raise privacy and business issues. (Recall, for example, the recent publication (and withdrawal) of web-search data by AOL Labs.) A possible alternative to a downloadable corpus for matching tests is having a trusted third party control a single copy of the data. Researchers could install matching software at the third party site, and have its performance measured there. An obvious issue is how to report problems (false positives and negatives) back to the information without compromising the security of data. Perhaps some form of “post-match” anonymization can be developed.

Intensive, short-term research collaborations: The example of the CLSP summer workshops at Johns Hopkins was cited as a model for how promising but under-investigated problems in a domain can make substantial progress through a targeted research workshop. Such workshops often make progress because they bring together cross-disciplinary teams or assemble unique combinations of software and data resources.

Take better advantage of progress elsewhere: The need for information matching is ubiquitous, and it is likely a target of research and development in most countries with activities in the computer and information science domains. However, the group felt that the awareness of foreign efforts was low, in general. Hence, research workshops with an international focus could be a valuable means to make advances elsewhere more visible in the US.

Maintain a moderated list of Grand Challenge problems: Each problem statement should give sufficient details on the problem to understand what the requirements are on matching, the characteristics of a good solution, and what the payoff is for solving the problem. Perhaps there should be an annual designation of the best solution to each problem to date.

Establish a benchmark set of data and problems with performance and accuracy metrics: One approach is to start with a common core, then extend into specialty corners. There might also

be different versions of a benchmark, such as batch versus on-line varieties. Some suggested sources of data or problems were the tobacco corpus, Genbank data, patents (national and international), high-volume matching for watch lists, and privacy-preserving matching.

Mapping

Participants

AnHai Doan, Michael Fitzmaurice, Howard Ho, Grigoris Karvounarakis, Sergey Melnik, Renee J. Miller (lead), Dan Suciu, Val Tannen

Introduction: the Nature of the Problem

A **schema mapping** is a specification of the semantic relationship between models of data. We will refer to the models of data as schemas, and schemas may be represented using the relational model, XML, an object-relational model, or more generally in a language for specifying data.¹

Mappings are often based on a more primitive notion called a matching. A **matching** is a representation of the correspondence or association between individual data values or individual elements within a schema. Matchings may be discovered or constructed using statistical inference or other forms of inference including ontology alignment. A matching can be used to help derive a mapping, which relates the possible instances the schema.

Schema mappings are important in solving a number of integration problems. The following is a list of such problems that arise commonly, but is not meant to be complete.

- *Data Exchange* – data exchange is the problem of taking data under a source schema and creating an instance of a target schema that reflects the source data as described by a mapping. Data exchange is sometimes referred to as data translation and is in fact a generalization of data warehousing (where the mappings may be simple view definitions).
- *Query Reformulation* – consists of taking queries on a target schema and translating them into queries on a source schema using a mapping. The source queries are executed and the results are returned as results to the target queries. Sometimes the results need to be further translated using the mapping (e.g., they may need to be combined when the source is in fact a vector of sources). In the research literature, this problem is also referred to as (virtual) data integration because data is translated at query time (on-demand) as opposed to data exchange where data is translated and materialized into a target (and can then be queried directly).

¹ This terminology is used by database practitioners but it is at variance with the terminology used in logic. There, a model is what is called in database an instance and a database schema corresponds to a logical signature. From a logical perspective a schema mapping between two schemas is an assertion expressed using both signatures corresponding to the schemas. The instances of the two schemas are expected to jointly satisfy the assertion (the mapping).

- *Update Propagation* – where updates specified on a source schema instance must be translated into updates on a target schema instance so that the mapping continues to be valid.

The use of mappings in data integration depends on the satisfactory solution of the following scientific problems:

- *Mapping Creation* – development of tools and techniques to help in the design and creation of schema mappings.
- *Mapping Debugging* and *Mapping Understanding* – development of tools and techniques to help design, understand and refine schema mappings. A closely related problem is that of visualizing mappings.
- *Mapping Composition* – given a mapping from schema S_1 to schema S_2 and a mapping from schema S_2 to schema S_3 , define a mapping from schema S_1 to S_3 with the corresponding validity property relating instances of S_1 and S_3 via some instance of S_2 .
- *Mapping Inversion* – given a mapping from schema S_1 to schema S_2 , create a mapping from schema S_2 to schema S_1 with the corresponding instance validity property. In both inversion and composition, it is important to understand the detailed validity properties of the resulting mapping.
- *Specification Formalism* – used to express mappings. We need to understand the expressive power of different formalisms, understand the properties of different formalisms, and understand their suitability for use in different integration tasks.

What Can We Do Well?

State of the Art

In the last decade, there has been significant work on understanding how to factor the problem space in schema mapping research. The problems listed above are largely a result of this study.

There has been significant work on *ontology alignment*, that is, creating mappings between ontologies. Mostly this work has focused on matchings hence they led to relatively simple mappings. These mappings are represented in logical formalisms that depend on the ontology language.

For structured and semistructured data (including relational and nested data such as XML data), there has been considerable work on developing a *formal semantics* for schema mappings. Most work has focused on the use of GLAV (global-and-local-as-view) mappings. A GLAV mapping, informally, is specified by a containment or an equivalence relation between a conjunctive query over a source schema and a conjunctive query over a target schema. GLAV mappings are a generalization of views (named queries) that permit the representation of incompleteness in two

ways: one is by using containment instead of equivalence, the other is by using conjunctive queries (with projections) instead of the actual source or target relations. Logic (first order logic and to some extent second order logic) has also been used as a specification formalism for mappings. Tuple generating dependencies (tgds) are the most commonly used logical mapping formalism. Like GLAV mappings, tgds permit the specification of incompleteness in two ways: one is by using implication instead of logical equality, and the other by using existentials within the mapping.

Most work has focused on the use of mappings in enterprise environments. These are relatively stable environments where data sources are known and where it is important that mappings be well-designed and accurate (for example, it may be important that data is not lost by a mapping). Mappings may be used to represent the transformation (exchange) of data into a data warehouse. They are also used in federations to represent the relationship between data sources and a virtual integrated view. Some (limited) work has considered the use of schema mappings in more autonomous networks of peers.

We have a solid foundation of work on *semi-automatic techniques* for creating mappings, maintaining mappings (as schemas change), and using mappings in both data exchange and query reformulation. This work considers mappings for both relational and nested relational models such as XML.

We have some (limited) work on understanding and debugging mappings (for relational mappings) and on propagating updates through mappings (again for relational mappings).

We also have a strong theoretical foundation in mapping composition and inversion, but little systems work in these areas.

State of the Practice

The specification languages used for schema mappings in practice (that is, in commercial systems) include queries or views. Such mappings have the benefit of being declarative and data independent. However, they cannot represent incompleteness in the specification (as in GLAV mappings).

More commonly products use procedural mapping languages, including scripting languages such as XSLT and object-relational extensions to model mappings. The problem with using more expressive languages for specifications is that reasoning about mappings, including query reformulation, mapping composition and mapping inversion becomes less tractable, even uncomputable.

Many products use proprietary solutions without any formal semantics. This includes specialized schema annotation frameworks and proprietary ETL workflow scripts.

There are many visual tools for helping in mapping creation, but most of the work must still be done manually. At best, these tools provide a library of type-conversion and value transformation functions to use in writing a schema mapping. Debugging and maintaining mappings is largely

manual. However, these tools may have good execution engines for evaluating data transformations.

Some of the research mentioned above has been transferred into practice, particularly in the products of a number of large data service and product providers. IBM's Rational Data Architect incorporates research on helping to automate the mapping creation task and supports both views and general GLAV mappings. IBM's Websphere Information Integration, Microsoft's ADO.NET, and BEA's AquaLogic include execution environments for using mappings in query reformulation. Many tools optimize the execution of data transformation programs and can execute such programs on very large data sources. These products work mostly with relational and XML data.

Why Can't We Declare Victory?

The field of data integration and sharing is indeed just beginning. We are still facing a number of unsolved technical application challenges. We list below several challenges that were identified by this group (others may well exist). Among these challenges, some the technical problems constitute continuing barriers to the more widespread acceptance of mapping techniques while for practical purposes most of the domain and community challenges will have to be overcome.

R&D Challenges

Technical Challenges

- Formal mapping specification
 - Probabilistic/approximate mappings
 - Interaction with integrity constraints, repairs and incompleteness
 - For data models other than relational and XML, e.g., unstructured data
 - More general logics, e.g., negation, aggregation, recursion, higher-order
 - Formalisms for ETL
 - Mappings between Web service specifications, EAI
 - Streaming, sensor data
- Practical mapping specification
 - Standardization
 - Common visual specifications
- Unifying theories of mapping usage
 - interchange = integration + exchange
 - sharing = integration + update propagation
- Shared datasets and benchmarks (need to learn from ontology alignment contests)
 - User studies
 - Measures of success
- Tools that scale with schema, specification, and network complexity
 - Usability, low cost of "ownership" for mappings

- Design theory for mappings
 - “Understanding” mappings
 - General purpose design principles and patterns
 - Debugging processes
 - Incremental implementation/change
- Formalize the bridge from matching and ontology alignment to mapping
- Mapping/query execution
 - Optimization in execution engines to exploit redundancy
 - Runtime environments (synchronization, exception handling)
 - Integration of runtime and design time
- Interaction between mappings and security, privacy, access control, authentication, and authority
- Integrate the *tools* for integrating data sources, tools for ontology and for schema specification with those for mapping specification and construction
- Maintaining data provenance through mappings
- Mappings between rapidly (even continuously) changing schemas as in information extraction from text, video, etc.
- Interaction between mappings and queries with both structured and search-like components
- Create a standard mapping language
 - Unification of EII and ETL by defining high-level declarative mapping language for ETL and by refining the EII mapping semantics. This will enable the transformation from an EII mapping instance to an ETL job and vice versa.
- Best-effort mappings
 - Use of underspecified or incomplete mappings
- Discovery and reuse of ontologies, mappings
 - Here the focus is not on global standards but is on better tools at the small scale of e.g. between individual databases.
- Mapping in non-enterprise environments
 - Dynamic web or networked environments

Challenge Domains

Schema mappings can be considered to be plumbing. They are ‘under the hood’ (the benefits, while considerable, are indirect to end-users, in general). As a result, for-profit companies are unlikely to fund the necessary research and development. Companies are convinced of the need for schema mapping in enterprise information, but are unlikely to fund work in the following areas where the business model is less clear.

- Health Care
- Life Sciences
- Ecology
- Homeland Security
- Scientific Discovery
- Web User Life

Community Challenges, Resources

The standardization of a common mapping specification language would greatly facilitate tool development and research and it would also enhance the quick transfer of research into products.

Scaling up the mapping problem via *incentives*: how can we entice individuals (or the masses) to create mappings? For Health Care, for example, we first need data about patient outcomes v. procedures/technologies used (which requires a solution to the basic challenge of defining a common terminology and then aggregating the numbers from across the nation). Then, incentives could be established in terms of relative performance, conformance, etc. for healthcare companies and individual providers. (ICD9 is a codeset in common use, but it does not have sufficient granularity. Proprietary extensions have locked customers into vendors and prevent interoperability.)

What are the business models of mapping manufacture and usage? Costs, licensing, etc. We are in dire need of education and training in schema mapping. Schema mapping is not a common topic in undergraduate programs and is only sporadically covered in graduate programs. We lack a good reference text on schema mapping.

Research Roadmap

3 year horizon

- a satisfactory theory of uncertainty in mappings
- a first draft of a design theory for mappings
- good ideas for integration-integration, that is, ETL, EII, and EAI integration
- development of mapping benchmarks
- tools for data provenance propagation through mappings
- schema and mapping theory for streaming data

5 year horizon

- more expressive formalisms for mapping specification capturing negation, aggregation, recursion, higher-order features; good theory of relationship between formalisms
- research tools for ETL, EII, and EAI integration
- highly usable tools for mapping design with debugging facilities
- security/privacy components to any mapping
- extensive use of benchmarks
- research tools for the integration of streaming data
- unification of theories of mapping usage

7 year horizon

- integration of schema, data and ontology mapping paradigms
- robust mapping software systems

- support maintainability, adaptability and evolution
 - low cost of ownership for mapping tools
- widespread commercialization
 - emergence of commercial tools for ETL, EII, and EAI integration

10 year horizon

- new directions in mapping research
 - commoditization of mappings will permit use of mappings in new, unforeseen metadata tasks
- tools for ontology mapping
 - emergence depends on wide spread use of ontologies (at the level of current day use of databases)
- integration of process translation, transformation and mapping tools with tools for data, schema and ontology mapping

Conclusions

Data integration systems are software systems that permit the transformation, integration, and exchange of data that has been designed and developed independently. The often subtle and complex interdependencies within data can make the creation, maintenance, and use of such systems quite challenging. Mappings are a fundamental abstraction for reasoning about and achieving the reconciliation of semantic differences in schemas and data. While there is a growing maturity in our knowledge of how to create and use mappings in such tasks as query answering, data exchange, data integration, and data sharing, much work remains to be done. Understanding the foundations of mappings and the intricacies of using them in real systems is an important problem that deserves more attention by researchers in our increasingly networked world. Success in this area is of paramount importance to business, governments

Information Integration in Context

Participants

Barbara Blaustein, Peter Buneman (reporter), Sarah Cohen-Boulakia (reporter), Robert Chaddock, Yi Chen, Laura Haas, Tim Finin, James French, Yannis Ioannidis, Subbarao Kambhampati, Deborah McGuinness, Victor Markowitz, Wang-Chiew Tan, Kenneth Thibodeau and Gary Walter

Introduction

The purpose of this panel was to look at the larger picture surrounding data integration. Although huge strides have been made in information integration technology, the holy grail of system that will integrate data at the push of a button remains as elusive as ever. The problem lies not with the integration technology itself but with contextual issues: How does one find and select the relevant data sets? How much technical knowledge is needed to understand these sources? What guarantees does one have on the quality of data? How do we know what trust to put in the integrated data set? Does data integration compromise security or intellectual property issues? The list is almost endless because data integration impacts on almost every other aspect of databases. However, some of these topics are crucial to the success of data integration.

One of the findings of the panel was that not only further research but also the development of appropriate training in both computer science and the relevant sciences/domains is necessary for the practical advancement of information integration.

What can we do well?

The first observation is that anything involving databases has to be done efficiently. Doing things fast is one of the main thrusts of database research; it is what made the subject in the first place; and it continues to provide the necessary basis for effective integration. In addition to this there are a number of technologies that are becoming increasingly mature.

- Languages that will allow efficient access to a variety of data models, data formats and physical storage layers.
- Representation of partial knowledge, which is engendered by information integration.
- Emergence of a common data format, XML, which deals with – at least – low level aspects of data transmission and parsing.
- Schema mapping
- Data and schema matching

The last two of these were specific topics of other panels.

Why can't we declare victory?

We cannot declare victory because so many of the topics mentioned in the introduction are essential to the complete task of data integration. To take two obvious examples: one cannot integrate anything without first identifying the sources, and one cannot effectively integrate dirty data with anything. More importantly we need to set up a *culture* in which information integration is understood as part of the process of scholarship and scientific research. For the purposes of this discussion it is useful to divide users of integration technology into two classes: one is the *brokers*, the people who generate integrated data sets, the other is the *consumers*, the end users of these data sets. They are not always distinct: there are cases of someone doing “one-off” integration for the sole purpose of answering a single question.

No-one believes that there is a silver bullet for information integration. We shall always need brokers, but what kind of skills should they have? We expect that data integration tools will become sufficiently simple that they will not need a PhD in Computer Science to use them. However, no-one thinks that they can be used without some kind of understanding of data models and how integration tools operate on them. More importantly we expect the brokers to be domain experts. So will these people, for example, be biologists with some knowledge of CS will they be computer scientists who have learnt some biology or will we require both? Our current belief is that we need both, moreover special training should be introduced into the relevant curricula in data integration, and that understanding the semantics of data models/schemas/ontologies should be an important part of this training.

Equally important to training is the process of data design. New scientific databases are coming on-line at an extraordinary rate. However, most of these are designed to serve the needs of a specific community of users. The people who generate new databases to serve such communities also need to be aware that, if their data is of any value, it will also be the object of integration technology, and they should prepare their data for integration.

Specific R&D challenges

Forms of integration

The current view of integration is to combine n sources into one, but within this general prescription there is a wide variety of forms of integration. If we assume (usually incorrectly) that the sources are static there is the issue of whether we are trying to produce a comprehensive integration of all the relevant sources, as is done in some scientific databases or are we trying to build something that will answer a narrow, focused set of queries. Are the demands on the brokers that they produce something as quickly as possible (best effort) or do they have the time to do a thorough job. Another issue that has not been properly addressed in integration tools is that of time. In demographic and health data integration of several data sets from different periods in time is often used to get a “longitudinal” study of how data such as population or the incidence of a disease

changes over time. Synchrony is also important in ordinary integration. There are many stories of disasters that have resulted from integration with stale data.

Security and privacy

Does integration compromise security? This goes both ways. Sometimes integration is used to ensure privacy. For example joining on identifiers and then erasing those identifiers and other personal data to obtain aggregate data for medical research is common practice. Conversely trying to combine anonymized data is used -- sometimes benignly -- to identify people with medical conditions. Security can also be an issue in blocking explanations and provenance (see below).

Source selection and discovery

Finding and assessing the relevant sources is not always straightforward, and once the sources have been identified, it is often the case that several sources will provide the "same" data so that the broker is faced with the problem of source selection. Here is a list of some of the factors that may govern source selection:

- authorship -- who generated the data in the case that it was a person?
- provenance -- where did the data come from, in the case that it was copied from somewhere?
- completeness -- does the source provide comprehensive coverage of the data of interest?
- performance -- how efficiently can the tools interact with the source
- format -- integration tools are, of course designed to deal with different formats, but there are always issues of how easily the format can be understood (for example, is further parsing of text fields or conversion of numeric fields needed.)
- cost
- cleanliness -- does the data conform to constraints that may be assumed by the schema used in the integration rules?
- freshness -- how up-to-date is the source?

Many of these topics fall under the general heading of *data quality* or *trust* -- a topic that requires further investment of research effort.

Source understanding

How do brokers understand other people's data? This is always a problem in data integration. The overarching requirement is that the sources should be documented at all levels: the description of the units used in some numeric field is as important as the description of the schema; moreover (see below) it is important that this information is carried through integration. Schemas do not always provide adequate documentation on how databases are structured. And they typically do not provide adequate information about the semantics of types and the relevant operations.

Several people have noted that looking at the data is often a better entry into understanding a data source than looking at the schema. In fact some mapping tools have a facility for doing this. Many databases are not adequately documented.

The general principle that is needed here is that databases should be designed on the assumption that they will be used by integration tools.

Databases or Ontologies?

Ontologies have recently emerged as an alternative to databases for certain kinds of knowledge representation. Ontologies may permit a more flexible approach to designing data sets -- especially in their initial evolution -- while databases still provide the basic technology needed for efficient access to large shared data sets. Is one more appropriate than the other for integration? There is a close relationship between database schemas and ontologies, so the likelihood is that integration tools developed for one of these will be transferable to the others. Nearly all the issues discussed in this section on context are applicable to both.

Data consumers

Most interfaces for consumers are engineered -- as they should be -- to present the data in a form that is most understandable. Typical examples include the sequence displays one finds in genetic databases and the pictorial displays of astronomical data. Unfortunately the data display is far too often the *only* end product of the integration. There are two other important extensions that are needed.

First the product of any integration service is likely, if it is of sufficient quality, to be needed as input to some further analysis tool or some other integration service. Therefore it is also essential that the data is made available in some clean understandable format with full documentation. Many integration tools do this, but given the further demands we are about to place upon data integration tools, this is an entirely non-trivial task.

The second extension is that of *explanation*. Consumers need more than a well engineered presentation of the data. They need the ability to "drill down" or expand parts of the data in order

to understand why it is there. This is a sufficiently important addition to current data integration tools that it requires special attention.

Explanation

There are cases in which the results of some integration task were dismissed by the consumers, simply because they did not know how the integration had been performed. One financial institution reports that up to 50% of an integration task is the process of identifying the best sources. Worse, the same users then attempted to re-integrate the relevant sources in order to be sure that they knew what the integrated data meant. Some form of explanation of *how* the integration was performed and the relevant contextual information is essential. Two components of explanation are particularly important.

Provenance

In the context of information integration, provenance is a record of how the integrated data was derived. Two general approaches to this general topic have emerged. The first is *workflow* provenance in which one keeps a complete record of how the resulting data set evolved. Workflow provenance involves recording not only the data sets and the transformations that were performed, but also the details of external computations that may have been used. For example Blast searches are frequently used in biological data integration. The second *fine-grain* approach focuses on individual data elements in (or small components of) the integrated data set and asks for the provenance of that element. It may be that providing the provenance of small pieces is simpler than recording an entire workflow. This is certainly true in the case that the integration tools simply caused copying of the data.

Annotation and context

Important contextual information is often lost in integration. An integration tool may do a perfectly good job of producing accurate data, but may lose important descriptive data, such as dimensional information that is only available in a comment in the source schema. It is essential that such documentation is carried through, as some form of annotation into the integrated result. This is again a non-trivial task and may require an understanding of provenance in understanding how the documentation should be carried through. Even determining what documentation in the source schema is relevant to the extracted data is non-trivial and usually requires human judgment. Finally there are external factors. Knowing the purpose for which a data set was constructed or why an integration task was performed is essential in assessing its use for other purposes.

Data Cleaning

It was noted by people who had been engaged in any practical attempt at data integration or transformation that they often spent more time cleaning the data than applying the transformation

or integration tools. Moreover the impact of their efforts is often limited. The corrections – for a variety of reasons – seldom propagate back to the sources, nor do they survive to the next round of integration that is needed after the sources have been updated.

Much data cleaning is done manually, or is assisted by some basic automatic tool such as a generalized spelling-checker. Much more needs to be done especially in situations in which database constraints come into play. At the same time we need to improve the practical use of view update techniques, which is what is needed to propagate corrections back to the source data.

Compositionality

To return to our earlier point concerning the requirement that data integration tools should produce, in addition to good user interfaces, well-organized data sets that can be used by other systems, perhaps by other integration tools. The key idea here is compositionality. Not only should integration be composable, but so should the systems that convey documentation, context and provenance. This is one of the key challenges in bringing integration technology to market.

Curated Databases

A substantial number of databases are produced not by integration tools but by humans who constantly use their judgment in the selection and classification of data. This is true of many of the 800 or more molecular biology databases and it is also true of many on-line reference manuals, gazetteers, business compendia, etc. While automated integration is less relevant to the “curators” of these databases, nearly all the issues that we have discussed in this section on context are equally important to curated databases, and they are equally challenging.

Research strategies and roadmap (3, 5, 7 and 10 years)

It is difficult to give time precise predictions for times needed to bring some of the topics to conclusion. In fact, in cases such as data cleaning we can only expect progress to be incremental. However there are some cases, which do not involve extensive research, in which we can provide estimates of time involved.

3 years

We have stressed that education is one of the most important factors in effective data integration. We need scientists – the “domain experts” -- to be educated in data models. Without an understanding of the representation of data, integration is likely to produce erroneous or incomplete results. Educational programs in this need to be developed.

However, to make this possible, the database community needs to “clean up its act” on the topic of data models. Although most database design is undertaken using some form of entity-relationship (E-R) model, it is very difficult to find an agreed and precise semantics for these. Open any two

database textbooks and you will almost certainly find that they disagree on the details of E-R diagrams and are fuzzy on the semantics. Bringing these into line is not difficult and is almost certainly regarded as a tedious task by database researchers, but it is essential.

The same needs to be done in ontologies. A simple “core” of RDF is not far from E-R modeling, but this also needs to be expressed simply and with a clear semantics – a task that database researchers have tried to initiate.

5-7 years

We have already started to see progress in research in some of the topics mentioned above, notably data models and model management, provenance, annotation, data cleaning and topics related to compositionality. Much of the research is in its initial stages and has not yet come to market. Depending on the encouragement and funding that the research is given, we may expect to see the fruits of this research coming to market in this time-frame.

Beyond 10 years

In order to make information integration really effective, it is not sufficient to develop additional tools. We need to change the existing environment. For example, curated databases are frequently constructed by a curator copying from some data found on the web to a forms interface for a local database. Unfortunately the ubiquitous control-C – control-V sequence loses both contextual and provenance information. Improving operating environments to take care of this is a major undertaking. A similar story could be told about the automatic generation of SQL programs, which are also used widely in data integration. Improving the operating systems and the user interfaces will be the real challenge in data integration.

Conclusions

We have argued that information integration is not an isolated problem that will be solved by the development of the right integration tools. While such tools are certainly essential, effective integration will depend on a number of other technologies that are currently in their infancy. It will also depend on having a community of appropriately educated experts who are familiar with the variety of ways in which structure is imposed on data.

Information Integration Systems

Participants

Phil Bernstein, Howard Bilofsky, Michael Carey, Barbara Eckman, Todd Hughes, H. V. Jagadish (reporter), Anupam Joshi, Hilmar Lapp, Tom Lee (reporter), Bertram Ludaescher, Louiqa Raschid, and Arnie Rosenthal

Introduction

The panel that developed a taxonomy of information integration (presented earlier), also considered big picture system issues, in terms of system architecture and technology, and also in terms of the surrounding legal, social, and economic issues. The results from those deliberations are presented here.

What Can We Do Well?

Information integration has been studied for several years already, and we have progressively become better. At the most basic level, we had early systems that were able to perform format conversion and ingest data from remote sources. With the advent of XML, and open connectivity standards in the computing industry, we no longer need to worry about low level connectivity, access and format conversion. Today, we can be sure that we can get at information from multiple sources – the question is whether we can put it together with other information we may have and make sense of it.

Companies have recognized the value of integrating their information systems, and there is considerable effort being devoted today to enterprise information integration. For a carefully specified set of well-defined corporate systems, we are able to develop comprehensive integrated solutions that can render the specific database locus transparent for any piece of information of interest.

Why Can We Not Declare Victory?

Even though corporate information integration is possible today, it is enormously expensive, often requiring thousands of staff months for a large organization. It is also very brittle – individual units have to give up autonomy and toe the global corporate line so that their data can be utilized by others. This makes change difficult, and individual units either have to sacrifice agility, or break the information integration when circumstances demand that they change their schema. What we would like is quick, cheap, and accurate information integration on poorly defined and autonomous sources – this vision is not likely to be achieved in its entirety even in the next decade. However, much that is valuable can be attained at intermediate milestones, which are possible in the next several years. We next attempt to devise a research agenda toward this end.

A Research Agenda

Given the wide variety of information integration problems possible, it is clear that one size does not fit all. Do we need many information technologies, specialized for different purposes, or does? The answer to this question is fairly easy – there is need for a broad range of technologies if the possible variety of problems is to be tackled. Perhaps more important, it was felt that systems researchers should be looking at the convergence of all these technologies. We all need to agree on an overall framework of major components that will be there, even if there are many alternatives for each component. While there was broad agreement that this was a central issue that should be addressed with high priority, it was also felt that this may be hard to do unless the integration context/domain is well-defined. E.g. streaming integration may be important, but architectural pieces for streaming may be different than for traditional databases.

Service-oriented architectures (SOA) have become quite popular and provide an excellent framework for interoperability of application code, just as XML provides a common data format for data sharing. SOA is fine as far as it goes, but it does **not** solve the data problem, just as using XML does not solve the information integration problem.

Thirty-five years of experience with database technology has clearly demonstrated the value of declarative specifications. This is just as true for information integration as for querying. The keynote talk by Mike Carey made this point very well. This is true for the entire range of integration problems. For instance, consider streaming data, which is quite different from stored data – yet declarative subscriptions are useful for such data.

It was felt that there was a pressing need for a “CAD framework” – or a development environment for data architects. This would incorporate many of the topics discussed above, including an architecture with well-defined pieces of the integration process, which can then be mixed and matched; a declarative specification and query language, permitting creative optimizations under the covers; and tools to manipulate artifacts of integration, including

- Means to explain integration results (and transformations applied) to the user,
- Means to debug integration artifacts to fix errors, and
- Means to modify them to manage change (to data sources or integration needs).

Existing data modeling tools are inadequate: UML does not have everything we need.

Workflow systems are an important special case of data and process integration.

Workflows provide a sequential model of data integration (e.g. a framework for defining how to integrate). They are the “upper ware” of cyber-infrastructure. An ordinary query can be considered a simple one-step workflow. Process “integration” is a poor man’s way of doing data integration when enough data integration is not provided. But data integration can also be a piece of a larger workflow. Business process integration is a high level view of integration, involving both data and process.

Artifacts of Integration

Integration has hitherto been treated as a unique process. As with any other process, there can be significant benefits, in both cost and quality, from reuse. If we “share” or re-use, this begs the

question what precisely are we re-using? What is a (shareable and reusable) integration artifact? The answer should not be just the integrated data result. An integration artifact could be a workflow. It could be relationship knowledge that is critical to getting some mapping correct. We need to develop a model for describing integration artifacts.

We also need to develop a model for reuse specification and reusable components of integration (such as composing/merging/mapping schemas). We could bundle such artifacts into an “appliance”. We can then think of in integration appliance as a model for imprecise integration. This may lead to easier deployment and adoption, and hence a lower price for a deployed information integration system.

Versioning and Identity

We need globally unique identifiers not just for data objects, but also for vocabulary, ontology, and other artifacts of integration. To be able to do this, all participants must agree on a language to preclude ambiguity in the subject of discussion. There are tricky socio-technical issues in determining who is responsible for assigning identifiers, and for maintaining them as changes occur over time. Real-world objects evolve, often requiring creation of new identifiers or merging of old ones. To understand this, consider an example of a purchasing history database maintained at a household level. If two individuals, initially living alone, get married, we merge two households into one. If they then get divorced, we have to split it back into two. However, reality may not always be so simple. What if they remain married but purchase a second home? Is that a new household? What if the husband spends most of his time in the second home whereas the wife is in the first? Reasonable people can disagree on where to draw the line and call it two households rather than one. In technical terms, objects in a database are identified based on a set of “key” attributes. Choosing this set of attributes is non-trivial in real life. This difficulty is compounded when values are not available for all attributes. Moreover, objects may change value for some key attributes, yet remain the “same” object – that is, the identifier has not changed. (E.g. our married couple may move to a new home, thereby changing their home address, an attribute that is part of the key, and yet not become a “different” household, with a new identifier, in consequence).

In the preceding paragraph we have focused on changing data. Difficulties are compounded because even the integration specifications evolve over time, as do the schema (of both source and target). Thus we have to consider both versioning of data and versioning of transformations. This gives rise to some particularly hard problems for correctly defining and maintaining provenance.

Multi-Step Integration

Real integration problems can often be very large. It may not be feasible, or at least not affordable, to integrate all at once. Rather, we may perform integration in multiple steps: first building small islands of integration, then bridging islands together. This requires that we think about a “federation of federations”. The result of one integration process may well be an input to the next integration process. Data management experts do naturally think about such closure properties. However, the theory of integration is still in its infancy, and most integration today is only single-level, so that such issues have not been sufficiently studied.

Multi-step integration can be with regard to the integration functions applied, rather than with regard to the data sources integrated. We can begin with a crude integration function, and refine it incrementally. For example, consider a workflow as an integration process: workflow evolution can result from iterative refinement of the integration process.

Provenance is crucial to maintain with information integration. As the integration process evolves, we have to consider (at least) two distinct notions of provenance:

- How did you transform the data
- How did you arrive at this particular sequence of transformations

Provenance issues more broadly are crucial to information integration and have been discussed previously in this report, in Chapter xxx.

Error Management

Source data often has errors in it. It is important that information integration techniques be robust to errors in source data, and handle these gracefully. Many integration processes begin with a data cleaning step to fix such errors. More generally, integration software makes assumptions regarding source data. It must also deal with exceptions to these assumptions when the need arises. We can model these assumptions as explicit constraints on the source data. Violations of these constraints can then be recorded in an “exceptions table.”

In addition to errors in data sources, many other types of errors must also be managed. We could have errors in the integration specification or software, physical faults during the integration process, and so on. If processes should be able to tolerate such faults as well. Addressing these challenges will require research to define the problem scope and possibly some solution techniques, followed by considerable engineering to work out all the details.

Access Control

There often are access restrictions on source data. These restrictions should be propagated appropriately to integration views. This is not easy to do. In fact, even understanding the access implications of view creation remains a challenge. Developing a framework for effective access control with integration is a worthwhile and difficult research challenge.

Standards

Standards, for schema and for terminology, can be very helpful for data integration. However, we note that it is unlikely there will ever be a single global information representation standard. Rather, there are likely to be several domain-specific data standards. There also will typically be overlapping communities of interest, each with its own needs and priorities. As such, a more realistic standardization goal should be to minimize, rather than eliminate, diversity. Furthermore the degree of compliance with respect to the standard can vary, whether there is one standard or many. Declarative specifications can be of value since one may be able to standardize interfaces without having to standardize implementations.

Standards can be valuable to many aspects of data integration. Whereas there are few standards, or even standardization efforts, with respect to most aspects of II, one area that has received a great deal of attention is that of standardizing terminology, or more generally, ontologies. Ontologies provide the same sorts of benefits as standards, but also provide similar drawbacks. A typical database creator has to ask: “Given that there are many ontologies that (vaguely) relate, how do I choose? What do I do?” A suggestion was made to develop a “universal” ontology with user/domain specific views and ontology/schema summaries for user understandability. However, there was no consensus on whether such an objective was feasible.

If all aspects of information representation are standardized, then the integration task becomes trivial. Of course, attaining this sort of standardization is not easy. In this regard, it is fruitful to think of standardization as one possible “large-scale” integration approach with significant up-front cost in return for future benefits. One can then perform an economic trade-off: ROI on integration versus cost of standardization and adoption. Such evaluations can lead to guidelines for creating communities to manage proposed standard specifications. Organization studies may reveal institutional barriers, suggest a maximum effective size of community, and so forth.

Benchmarks

Research in information integration could really benefit from having common schemas and mappings that could be used to validate and benchmark work. An excellent repository of large schemas is JDXM (<http://justicexml.gtri.gatech.edu/>) from the Justice department. This, or other such complex schema should be used as a basis to create a benchmark. Such a benchmark should include not just schemas and mappings, but also sample integration problems with fully-specified sample solutions.

A related issue is defining metrics for integration, that is, measuring how much progress has been made in meeting user needs. This frequently needs an expert evaluation – at least for deep semantics. In consequence, validation has typically been through a community of users. This often makes the validation anecdotal and not objective. To be able to measure progress, we should develop synthetic data sets with schema perturbations to enable internal evaluation. For example, we could define challenge tasks such as “In x hours how many entities/attributes can you get across to your partner, starting from scratch.”

Legal, Social, and Economic Issues

Finally, there are legal, social, and economic issues related to Information Integration, which it is important not to ignore. Even though we do not have the expertise ourselves to address (or even fully define) these issues, we felt it important at least to sketch the main points, and to understand the technical implications of these. We consider legal, social, and economic concerns in turn below. But before that, we discuss privacy, which has both legal and social roots, and is important enough for Information Integration to discuss as an issue category of its own.

PRIVACY ISSUES

The primary concern is that there is opportunity for leakage of data through integration. Detecting (and preventing) this is hard. Even a simple security audit is hard over integrated data. To know whether data from some source has been compromised, we may need to determine the provenance of every query that has been answered. Even after this, we would only be able to obtain a conservative bound (at best indicating that a privacy loss may have occurred) rather than an actual answer.

This difficulty is further compounded by the fact that even specification of privacy policies is really hard in real-life scenarios with complex combinations of data. This is particularly true as we attempt to create a fine grain specification. Once we have a privacy policy clearly specified, enforcement is a second challenge. While it is conceptually more tractable than the specification problem, there remain many technical challenges to efficient implementation.

In addition to privacy requirements on the source data, we can often have interesting policy specifications on the integration result or process. For instance, we may choose to allow temporary leakage during an *ad hoc* integration process, perhaps because the chances of leakage during that ephemeral period are assumed to be low. Another interesting requirement can be for “double blind” integration where the result of the integration can be made public but the “join ID” is to be kept private, perhaps because it carries personally identifying information, whereas the other fields do not.

Many corporations today have privacy policies that they attempt to follow. Given that information within a company is manipulated in various ways, and many integration products of information are created, it is our opinion that it is unlikely most corporations can certify that they are complying with their stated privacy policies. They may be able to show good faith attempts to comply with these policies, but that is a much weaker standard. Technical advances are required to enable solid certification.

LEGAL ISSUES

Information integration raises interesting questions regarding both ownership and liability. For each data source, let us say that there is a well-defined owner who enjoys both ownership rights and bears liability. How does this translate to integration products? Can we have data that is multiply owned when put together? Who bears liability for errors and incompleteness in integration? What are the rights and responsibilities of the integrator, as opposed to the source data owners.

SOCIAL ISSUES

When a composition of multiple pieces of information is obtained by integrating information from multiple sources, it is often important to give credit to the source providing the information. Automatic tracking of provenance is important, and integration systems today are beginning to track some provenance. However, there remains work to be done in terms of provenance extraction and representation, particularly if work is used at small granularity.

We need to define policies for archiving of derived products. It is difficult to reproduce the result of information integration, since it can change if any of the input data sources change. It could also change if any of the integration processes change. Therefore, for reproducibility, we must maintain the precise versions of each source as well as the integration process used. Such questions must be carefully considered if we define policies for archiving of integration products.

Often the “owners” of a database would prefer to maintain their autonomy, and would worry about ceding control of their data or schema to an integration process. Even if the integration process requires no changes of the data providers, it may still impose costs on future changes. These sorts of turf issues are not within the realm of technology, but must be settled before we can attain success in information integration.

ECONOMIC ISSUES/INCENTIVES

Aligning Costs and Benefits to Participants

While there are many places where information integration can be of value, we do not always see enthusiasm for it from all the players involved. This is because the people reaping the benefits from the integration are often not the ones paying the cost of integration. This problem is better understood for software reuse (but remains unsolved even in that case). Building a software component for reuse requires considerably more effort than building it for one-time use, so developers tend not to do it. Yet, from an organizational/societal perspective, reuse is a very good thing and can greatly reduce total cost.

For II to succeed, it is important to clearly identify who will do what, and make sure that they have the economic incentive to do it. E.g. if someone has to create and provide metadata, they should get some return for doing so. For example, we could make sure they get some technical benefits directly, independent of the downstream benefits from integration.

Some organizational incentives for II include audit pressures that force financial integration, and insurance requirements to protect company officers from data breaches.

Estimating Return on Investment

Another issue with Information Integration is that the economic value is often hard to quantify, and a significant upfront investment is required, at least for traditional engineered integration processes. Appropriate ROI arguments need to be made if the investment is to be supported. In this context, incremental integration can help reduce investment barriers, and decrease the risk of failure.

Fame is an important non-monetary incentive, and should not be forgotten. In today’s world, we are seeing more and more examples of people doing work for free in order to earn fame. For instance, people may spend a very large amount of time writing a blog, or maintaining an informational web site, which they make available for free to everyone. Once the blog or website

becomes popular, the authors can usually find some way to monetize it, for instance, by charging for advertisement on the site. Similar benefits can accrue from data integration.

Education

Information integration is central to a broad range of computing-related activity. Yet, topics such as integration and workflow are typically treated as specialized topics in most university curricula today, discussed in graduate seminars if at all. In contrast, other topics of similar importance, such as concurrency or encapsulation, are usually planned into the curriculum with care. Given the vital importance to many graduates when they enter the workforce, we recommend that they be made part of a standard undergraduate education in computer science.