The Design and Implementation of a Corporate Householding Knowledge Processor to Improve Data Quality

STUART MADNICK, RICHARD WANG, AND XIANG XIAN

STUART MADNICK has been a faculty member at MIT since 1972. He is currently John Norris Maguire Professor of Information Technology at the Sloan School of Management and Professor of Engineering Systems at the School of Engineering. He has served as the head of MIT's Information Technologies Group for more than 20 years. During that time, the group has been consistently rated number 1 in the nation among business school information technology programs (e.g., by *Business Week*). Professor Madnick has also been an affiliate member of MIT's Laboratory for Computer Science, a member of the research advisory committee of the International Financial Services Research Center, and a member of the executive committee of the Center for Information Systems Research. He has been active in industry, making significant contributions as a key designer and developer of projects such as IBM's VM/370 operating system and Lockheed's DIALOG information retrieval system. He has served as a consultant to many major corporations, such as IBM, AT&T, and Citicorp.

RICHARD WANG is Director, MIT Information Quality Program and Co-Director for MIT Total Data Quality Management Program. Dr. Wang has put the term *information quality* on the intellectual map with myriad publications and conferences. He has served as a professor at MIT, Boston University, and the University of Arizona, Tucson. In 1996, Professor Wang cofounded the premier International Conference on Information Quality, which he has served as the general conference chair, and currently Chairman of the Board. He also cofounded the Workshop on Information Technologies and Systems (WITS) in 1991. His writings have appeared in leading journals such as the *Communications of the ACM, Management Science, Journal of Management Information Systems, Information & Management,* and *IEEE Transactions.* Wang coauthored *Information Technology in Action: Trends and Perspectives* (Prentice Hall, 1993), *Data Quality Systems* (CMI, 1995), *Quality Information and Knowledge* (Prentice Hall, 1999), *Data Quality* (Kluwer Academic Publishers, 2001), and *Journey to Data Quality* (MIT Press, forthcoming).

XIANG XIAN holds a B.S. in Computer Science and Engineering and an M.Eng. in Electrical Engineering and Computer Science from MIT. She is a member of the Electrical Engineering and Computer Science Honor Society, the National Engineering Honor Society, and the National Society of Collegiate Scholars. Miss Xian is currently working at the Oracle Corporation.

ABSTRACT: Advances in corporate householding are needed to address certain categories of data quality problems caused by data misinterpretation. In this paper, we first summarize some of these data quality problems and our more recent results from studying corporate householding applications and knowledge exploration. Then we outline a technical approach to a corporate householding knowledge processor (CHKP) to solve a particularly important type of corporate householding problem—entity

Journal of Management Information Systems / Winter 2003–4, Vol. 20, No. 3, pp. 41–69.
© 2004 M.E. Sharpe, Inc.
0742–1222 / 2004 $9.50 + 0.00.

aggregation. We illustrate the operation of the CHKP by using a motivational example in account consolidation. Our CHKP design and implementation uses and expands on the COntext INterchange (COIN) technology to manage and process corporate householding knowledge.

KEY WORDS AND PHRASES: context mediation, corporate household, corporate householding, database interoperability, data quality, enterprise knowledge management.

TODAY'S BUSINESS ENVIRONMENT EVOLVES RAPIDLY. Corporate group structures and the relationships between corporate entities are becoming increasingly complex and difficult to understand. Misunderstandings can result in incorrect use of the data, which can have serious consequences.

Previous research on organizations has mainly focused on organizational knowledge management [1, 8]. However, effective use of knowledge about corporate structures and relationships has come to be an important issue in designing corporations' strategies and performing business functions. Analogous to a family household, a *corporate household* is defined as "a group of business units united or regarded united with the corporation, such as suppliers and customers whose relationships with the corporation must be captured, managed, and applied for various purposes" [12]. We also define *corporate household knowledge* to be the actionable knowledge about organizations and related internal and external relationships. The process of capturing, analyzing, understanding, and managing corporate household knowledge is known as *corporate householding* [14].

Context plays a large role in deciding how corporate household knowledge should be understood. For example, in order to answer the question "How many employees does IBM have?" we have to consider the purpose of the question, in other words, the context in which the question is asked. Following common practices in their respective fields, an insurance company and IBM's internal staff may come up with completely different answers—but which is correct? Possibly both! This is because they have chosen to aggregate the employee counts from groups of entities within this giant corporation in different ways. If we can capture the corporate householding process for each context using a set of context-specific rules, we will be able to automate part of the process, and thus benefit organizations in cost reduction, more efficient operation, and elimination of data quality errors.

Categories of Corporate Householding Problems

CORPORATE HOUSEHOLDING DATA PROBLEMS that businesses encounter come in various forms. Most of these problems can be categorized into the following three types:

- *Entity identification.* Part of the complexity that is involved in understanding corporate household data results from the multiple representations of the same entity. For instance, the corporate entity "IBM" can also be represented as "Inter-

national Business Machines Corporation" or "I.B.M.," though they all refer to exactly the same entity. One corporate entity can appear to be multiple entities in different data sources, and therefore can be difficult to identify correctly and efficiently.

- *Entity aggregation.* After we have identified that "IBM," "International Business Machine Corporation," and "I.B.M" refer to the same entity, we need to determine what exactly that entity is. A large corporation's corporate structure is usually very complex, including entities such as subsidiaries, branches, divisions, and joint ventures, normally with multiple layers. In one context, some parts of the corporate structure tree need to be taken into account in order to get a complete and correct view of the corporation; in another context, other parts of the tree may be considered. For example, in considering "How many employees does IBM have?" should Lotus employees be viewed as part of IBM? When to aggregate which entities requires an understanding of the precise meaning of the task at hand.
- *Transparency of inter-entity relationships.* Relationships between corporate entities may involve multiple layers. For example, a seller can sell its products directly to its customers or through a broker. Knowing when the layers are important, and when they are not poses another type of problem for corporate householding, which also has to be addressed depending on the context.

In most cases, when a corporate householding question is asked, the above three aspects all need to be considered in order to reach a correct answer. For example, if MIT wants to know how much it bought from IBM in 2002, it has to first identify all the instances of the same entity "IBM" in its records; then it needs to make sure to include purchases from entities that may not be directly related to IBM in their names, but are in fact part of the corporation, such as its software subsidiaries Lotus and Rational; finally, besides direct purchases of IBM products from IBM, purchases through brokers such as a local computer store, for example, CompUSA, may also be considered. In the later sections of this paper, we will primarily focus on the second type of corporate householding problems, that is, entity aggregation, and present a technical solution to aggregate corporate entities efficiently and eliminate the data quality problems caused by incorrect entity aggregation.

Previous Research

Research on Data Quality

RESEARCH EFFORTS IN DATA QUALITY have been ongoing for many years. Organizations typically store a vast amount of data on distributed and heterogeneous systems for their internal and external activities. Therefore, well-managed and high-quality data is crucial to a company's success. Traditionally, "high quality" refers to the accuracy of data. In order to target the problems more precisely, research conducted at MIT's Total Data Quality Management (TDQM) program [15, 18] has shown data quality as a multidimensional concept, which includes dimensions such as accessibil-

ity, timeliness, believability, relevance, and accuracy of data. Methods, models, tools, and techniques for managing data quality using the information product approach have also been proposed [15, 16]. The approach includes a modeling technique to systematically represent the manufacture of an information product, methods to evaluate data quality, and capabilities to manage data quality. One research effort of the TDQM program is *corporate householding,* which aims at better understanding and utilizing corporate household data [11].

Research on Family Household

The conventional meaning of a household is "the people of a house collectively" [13]. The term *householding* has also been used increasingly in places such as an announcement sent out by the Securities and Exchange Commission (SEC), which states: "the Securities and Exchange Commission enacted a new rule that allows multiple shareowners residing at the same address the convenience of receiving a single copy of proxy and information statements, annual reports and prospectuses if they consent to do so. This is known as 'Householding'" [13]. Traditional householding issues are similar to corporate householding problems. The structure of a family evolves over time, and may sometimes be very complex. Single mother or father families, families in which a husband and a wife have different last names, and many other forms of families make it difficult to define and identify a family household. For instance, if a child goes to college in another city, will he or she be considered as part of the household? If two people live together but are not married, do they form a household? Similar to the corporate householding problems, these questions have no single "right" answers. We will need to consider the underlying purposes of the questions. In this paper, we will only focus on corporate households.

Commercial Approaches

In this section, we will explain some state-of-the-art practices used by some of the industry leaders, FirstLogic Inc. and Dun & Bradstreet (D&B), to solve certain types of corporate householding problems.

FirstLogic uses the subject matter experts (SME) approach to identify entities correctly and efficiently. This approach helps to identify and build hierarchical structures in order to represent relationships between two households (either family household or corporate household).

Knowledgeable SMEs assist clients in establishing the business rules that identify the entities in their own family structure (referred to as the "internal view"), as well as entities in the family structure of their business targets (referred to as the "external view"). The involvement of SMEs makes it possible to perform householding across task domains. The FirstLogic tools then allow these rules to be applied across the company's database [13].

D&B has developed a representation of corporate structure to improve the understanding of their relationships. The Data Universal Numbering System (DUNS) number

is a unique nine-digit non-indicative identification number assigned to every business entity in D&B's databases. It is widely used for keeping track of millions of corporate group structures and their relationships worldwide. The D&B family tree is comprised of linkages and business relationships. It captures eight types of entities (single location subsidiary, headquarters, branch, division, subsidiary, parent, domestic ultimate, and global ultimate) and two types of relationships/linkages (branch to headquarters and subsidiary to parent). Each family member carries up to four DUNS numbers, including its own number, the number of its next highest member in the family, and its domestic ultimate's and global ultimate's numbers. D&B's approach captures a significant amount of useful information about a corporation, but there are some limitations to it. For example, the corporate householding applications of a company can be much broader than what the D&B family tree covers. For example, any subsidiaries that are less than 50 percent owned by parents are not listed in the parents' family trees. The DUNS numbers and the D&B family tree represent a major part of corporate structure data, but do not embed the corporate householding knowledge. Also, D&B's way of identifying corporations may not match the way data is organized in the corporations' internal databases.

Corporate Householding Application Areas

BUILDING ON THE DISCUSSION ABOUT PREVIOUS RESEARCH in the third section, we will further our understanding about corporate householding in this section by exploring a few common application areas with examples drawn from a review of the literature and interviews with subject experts. The issue of integrating information from multiple systems is a long-standing challenge [17]. Many examples are not industry-specific—any corporation may encounter similar problems in their business functions that relate to these areas.

Account Consolidation

Corporate householding is needed in the consolidation of financial statements. For example, consider a large organization such as IBM,[1] which has over 100 directly or indirectly owned subsidiaries—how should it prepare its financial statements? Should its financial statements be consolidated with those of Lotus, a company acquired by IBM? In the fifth section, we will use a simplified scenario of retrieving IBM's revenue to demonstrate a technical approach to corporate householding.

Financial Risks

Credit Risk

Since credit is a crucial consideration in many financial transactions, corporate householding in the area of credit risk [2] evaluation requires a significant amount of effort. For example, a firm planning to extend a large credit line to Hewlett Packard

Puerto Rico may find it useful to know that Hewlett Packard only has a rating of AA, though Hewlett Packard Puerto Rico has a credit rating of AAA. In other words, when evaluating the credit risk of a subsidiary, its parent and other related entities (if any) should be considered as well.

Bankruptcy Risk

Bankruptcy risk [4] is closely related to credit risk. When deciding whether or not to issue loans to a particular company, banks need to know who is responsible if the company bankrupts. For example, if a subsidiary goes bankrupt, how much liability (if any) does the parent company have? One concept that plays a significant role in the bankruptcy rules is affiliate. The definition of an "affiliate" covers parent corporations, subsidiaries of the debtor, and sister affiliates of the debtor, using a 20 percent stock ownership trigger. In addition, bankruptcy laws and regulations vary from country to country, increasing the need for corporate householding.

International Risk

As companies develop increasing global reach, risks caused by the differences in business protocols used in different parts of the world need to be considered. We name this risk "international risk." Consider a company that is located in Brazil but is also a division of a larger-sized American company. Or consider a company located in the United States, whose parent company is in Japan, such as a Toyota manufacturing plant in the United States. When should this plant be considered a "U.S. company" and when should it be considered simply a subsidiary of a Japanese firm?

Legal Sector

There are many types of corporate householding activities in the legal domain as well, such as issues of software and patent licensing. For example, if IBM licenses a patent from MIT, does Lotus automatically have use of that patent also? This problem can be further complicated by the consolidation of customers through mergers and acquisitions—especially for vendors of enterprise-wide solutions and those who sell enterprise-wide licenses.

Business Management and Operations

Corporate householding issues also exist in areas such as customer relationship management, supply chain management, sales and marketing, and business intelligence. A few examples follow:

- *Customer relationship management.* When the customer is a multinational corporation, the vendor usually has hundreds of unique contact records (individuals) in its information systems. Which individuals are relevant under which circumstances?

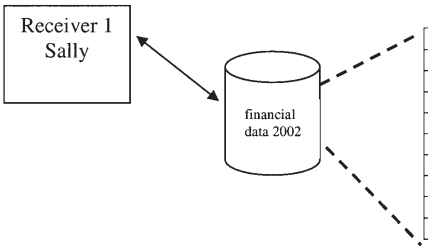
- *Sales and marketing.* There is a growing need for customer-identification systems that can provide integrated views of business-to-business customers to identify existing or high-potential customers, to assign resources to penetrate them, and to report on the performance of these efforts.
- *Supply chain management.* Identifying and maintaining relationship with material vendors is critical in order to achieve cost reduction. However, due to localized information systems, different manufacturing sites are highly likely to have different, independent relationships/contracts with the same vendor for the same material. The situation becomes even more complicated when a vendor has different relationships with different corporate function areas, such as manufacturing, financial, and accounting systems. Therefore, it becomes very hard to have a single and consistent view of a global vendor.

Corporate Householding Query Processor

ALTHOUGH CORPORATE HOUSEHOLDING APPLIES across all major business domains, most corporate householding problems can be classified into the three categories mentioned in the section second: (1) entity identification, (2) entity aggregation, and (3) transparency of inter-entity relationships. In this section, we propose a new approach for solving the second type of corporate householding problems—entity aggregation. This approach is based on an extension of the COntext INterchange (COIN) technology developed at MIT.

Motivational Example

Let us consider the following example. Suppose Sally is a financial analyst and she would like to find out what IBM's total revenue was in fiscal year 2002 (IBM's fiscal period ends on December 31). International Business Machines Corporation is a giant organization with about 100 years of history and numerous branches and offices around the globe with more than 100 subsidiaries[2] directly or indirectly owned by the company. Although these subsidiaries are legally independent organizations, according to the SEC's accounting rules, IBM should consolidate the revenues from all the majority-owned subsidiaries in its annual reports. Sally follows SEC's rules, but the database that she gets her data from represents IBM's revenue differently. The revenues of IBM and its subsidiaries are not consolidated. For illustration purposes, let us only consider a couple of subsidiaries of IBM–Lotus Development, IBM Far East Holdings B.V., International Information Products (Shenzhen) Co., Ltd., and IBM International Treasury Services. Lotus is directly and wholly owned by IBM; International Information Products is owned 80 percent by IBM Far East Holdings B.V., which is a wholly owned subsidiary of IBM; IBM International Treasury Services is owned by five different local branches of IBM in Europe, including IBM Germany and IBM France. The revenue number corresponding to "CorporateEntity = IBM" in the table "revenue1" in Figure 1 does not include the revenues of Lotus, IBM Far East Holdings, International Information Products, and IBM International Treasury Services.



Source Table: revenue1

CorporateEntity	Revenue
International Business Machines	77,966,000
IBM Global Services	36,360,000
Lotus Development	970,000
IBM Far East Holdings	550,000
International Information Products	1,200,000
IBM International Treasury Services	500,000
General Motors	177,828,100
Hughes Electronics	8,934,900
Electronic Data Systems	21,502,000

Figure 1. Motivational Example: Performing a Query for Total Revenue of IBM in 2002. *Notes:* The revenue data is directly extracted or estimated from the revenue/sales data of IBM, Lotus, GM, Hughes, and so on.

The source database also includes revenue data on other corporate entities related to IBM, such as one of its divisions—the company's consulting arm, IBM Global Services. However, because it is a division only (not an entity legally separated from IBM), its revenue is already consolidated in the revenue1 table and should not be double-counted. To illustrate Sally's accounting rules (SEC's rules) better, we summarize them in the decision tree in Figure 2.

Given the disparities between Sally (the "receiver") and the data source's accounting principles, if Sally issues a simple query (using SQL query notation):

Select CorporateEntity, Revenue from revenue1,

where CorporateEntity = "International Business Machines"

on the source database directly, she will get back:

CorporateEntity	Revenue
International Business Machines	77,966,000

But this result is not the one that she is looking for. The total revenue of IBM, from Sally's point of view, should include all the revenues of IBM's subsidiaries with majority ownership. For this simplified data source, the query should return the SUM of the revenues from International Business Machines, Lotus Development, IBM Far East Holdings, International Information Products, and IBM International Treasury Services. In other words, we should "aggregate" all these entities with their parent entity IBM, and hence, Sally's problem is an entity aggregation problem in corporate householding.

Realizing that the simple query is not sufficient, what should Sally do to modify the query so that the desired result will be returned? In theory, she will need to consider each one of a few hundred entities in IBM's corporate family using the decision tree in Figure 2, and find out whether any particular entity should or should not be aggregated. In order to traverse the tree successfully, Sally has to rely on some auxiliary data sources that provide information on the entities within IBM's corporate group, including ownership percentages, controlling financial interest, fiscal periods, and other related information.

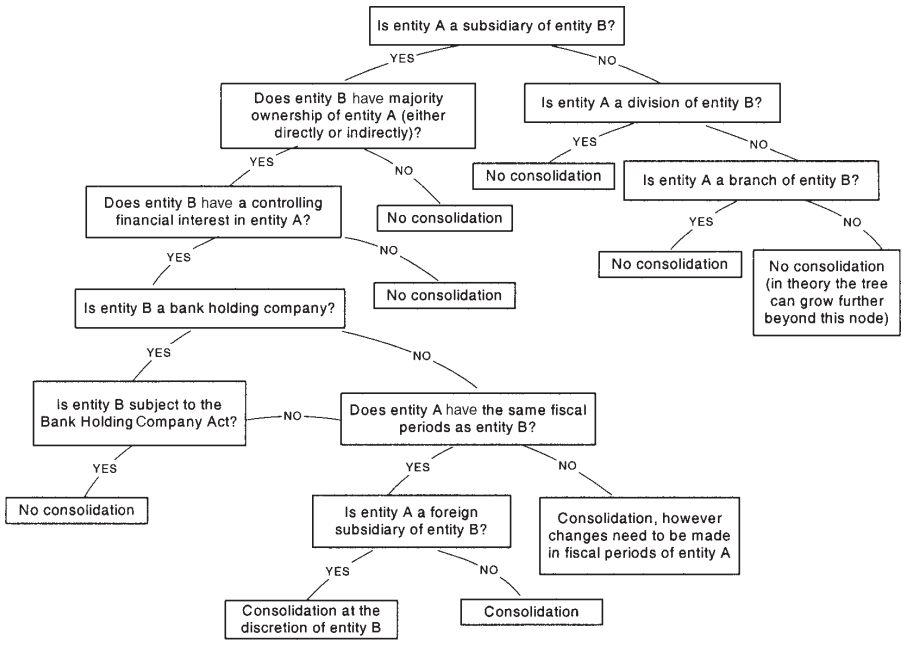


Figure 2. Decision Tree (simplified) That Represents the SEC's Rules for Calculating Total Revenue

For the simplified "revenue1" table in our example, entity A can take values such as Lotus Development, International Information Products and IBM Global Services; but entity B is reserved for International Business Machines. So Sally only needs to consider five pairs of entities. The steps of reasoning for {entity A = Lotus Development, entity B = International Business Machines} are illustrated in part (A) of Figure 3. Because Lotus Development Corporation is wholly owned by IBM, IBM has "majority ownership" of Lotus and has "a controlling financial interest" in Lotus. IBM is not a bank holding company,[3] and it has the same fiscal periods as Lotus. Both of them are incorporated in the United States. Based on the above information, Sally can conclude that Lotus's revenue should be consolidated with IBM's total revenue. Similarly, the revenues of IBM Far East Holdings, International Information Products, and IBM International Treasury Services should also be consolidated (assume IBM consolidates revenues from its foreign subsidiaries). On the other hand, when entity A = IBM Global Services and entity B = International Business Machines (steps of reasoning shown in part (B) of Figure 3), because IBM Global Services is a division of IBM, no consolidation should occur to avoid double counting.

After reasoning through the source data using her accounting rules, Sally knows that she should actually issue the following query on the database:

Select "IBM" as CorporateEntity, SUM(Revenue) as Revenue from revenue1

where CorporateEntity in ("International Business Machines," "Lotus Development," "IBM Far East Holdings," "International Information Products," "IBM International Treasury Services")

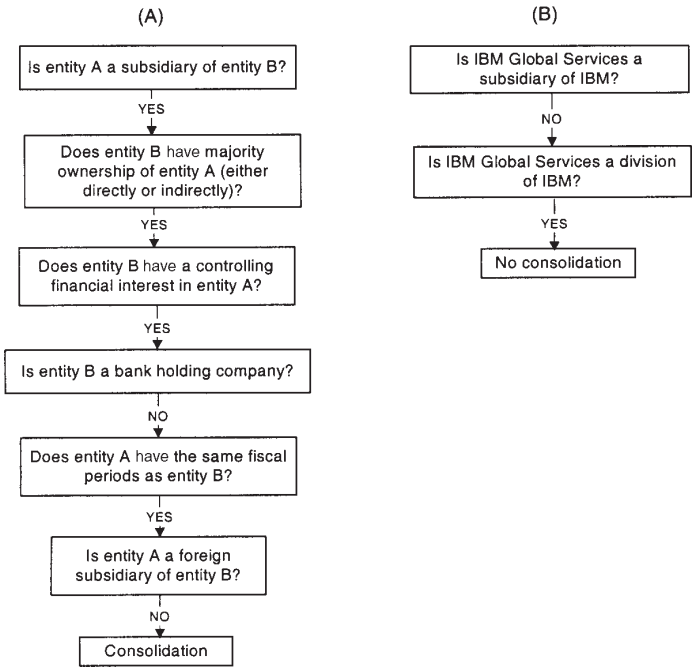


Figure 3. Steps of Reasoning Using the Decision Tree in Figure 2.
Notes: Entity A = Lotus Development in (A) and IBM Global Services in (B); Entity B = International Business Machines.

And she gets back

CorporateEntity	Revenue
IBM	81,186,000

This is the correct result because "77,966,000 + 970,000 + 550,000 + 1,200,000 + 500,000 = 81,186,000."

Now, let us suppose Sally's accounting rule has changed. According to the new rules, she consolidates only the revenues from the wholly owned subsidiaries, but not those that are only partially owned by IBM. Since International Information Products is only 80 percent (indirectly) owned by IBM, its revenue should not be consolidated with IBM's total revenue, whereas all the other subsidiaries in our example should be, because they are 100 percent owned by IBM. Therefore, Sally's query on the table revenue1 should look as follows:

Select "IBM" as CorporateEntity, SUM(Revenue) as Revenue from revenue1

where CorporateEntity in ("International Business Machines," "Lotus Development Corp," "IBM Far East Holdings," "IBM International Treasury Services").

The result she gets back is

CorporateEntity	Revenue
IBM	79,986,000

The above examples capture the essence of the entity aggregation problems. For any entity aggregation problem, whether its purpose is account consolidation or credit risk evaluation or sales and marketing, we want to find out what entities in the corporate family should be considered and should contribute to the final result, given the purpose. We perform corporate householding using the corporate group structure data, rules and regulations specific to the purpose, and other related information. The reasoning process described above can get very tedious and costly if the "revenue1" table Sally is querying on has all of the subsidiaries' data separated out from the parent, for she will have to consider the subsidiaries, divisions, and branches one by one. However, it is probably closer to reality than the simple table in Figure 1. It would be valuable to have a system that will capture the differences in aggregation rules between the source and the receiver of a query, test the entities recursively using these rules, and mediate the query according to the receiver's expectation to achieve the desired entity aggregation. That way Sally would not have to perform the corporate householding task manually when she searches for the total revenue of IBM. The COIN technology presents a unique approach to capture the differences between contexts (semantics of data sources and the receiver), to resolve those conflicts, and to output correct data in a fast and efficient way. The COIN system, with some extensions, can be used to store and process corporate householding knowledge. In particular, it can help to solve entity aggregation problems such as the challenges that the motivational example poses.

Overview of the COntext INterchange Technology

COIN technology [3, 7] is a mediation approach for semantic integration of disparate (heterogeneous and distributed) information sources in order to achieve semantic interoperability and logical connectivity [10]. The COIN architecture consists of three major components: (1) client processes, such as applications that perform queries on multiple databases; (2) server processes, including database gateways and wrappers; and (3) the mediator process, which is the core of the entire system. The context mediator rewrites queries in the receiver's "context" into a set of mediated queries where all conflicts are automatically detected and explicitly resolved. The process is based on an abduction procedure that determines what data are needed to answer the query and how conflicts should be resolved by using the axioms associated with the contexts. Automatic identification and reconciliation of conflicts are made possible by general knowledge of the application domain and the implicit assumptions associated to the sources and receivers. The three components of the COIN architecture work together to enable efficient and meaningful use of heterogeneous data, when the data sources and potential receivers have semantic differences. The modularity of

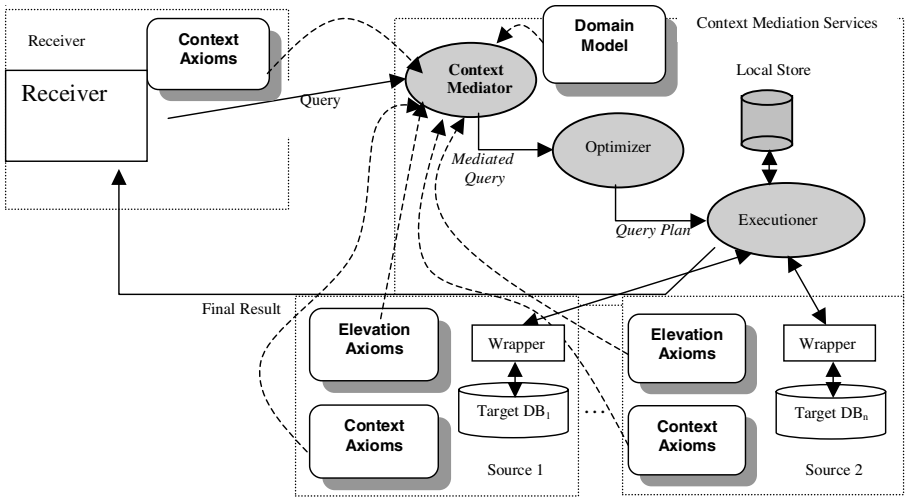


Figure 4. Architectural Overview of the COIN System

the design keeps the system scalable with increasing numbers of sources and receivers, extensible to local changes in the system, and accessible to end users.

Figure 4 illustrates the components of COIN. The COIN framework consists of a data model and a logical language, COINL, which are used to describe a Domain Model that can represent the sources and the receivers and the contexts associated with them. A Domain Model specifies the semantics of the "types" of information units, which constitute a vocabulary used in capturing the semantics of data in disparate sources. Three kinds of relationships are expressed: inheritance, attributes, and modifiers; the values of modifiers vary depending on the context. Together they define the ontology that will be used. The Elevation Axioms identify the correspondences between attributes in the sources and semantic types in the Domain Model. The Context Axioms define alternative interpretations of the semantic objects in different contexts. These three components enable the Context Mediator to generate the correct mediated query from the original user query. Besides the context mediator, the mediation services also include a query optimizer and a query executioner to enhance performance. The result of the query execution is reformulated into the receiver's context.

The research in context interchange and its framework has been ongoing for some time [3, 7], and a prototype system has been developed to validate the method. Applications that perform queries on disparate data sources (such as financial databases and online shopping sites) have been built and tested.

The extended COIN (ECOIN) system is an extension of the core COIN system, aimed to resolve *equational ontological conflicts,* which is defined as "the heterogeneity in the way data items are calculated from other data items in terms of definitional equations" [5, 6]. In ECOIN, modifiers are used to specify the definitional differences, and new constraints for basic mathematical operations, such as addition,

subtraction, and multiplication, are added. The implementation of corporate householding application presented in this paper utilizes the functionalities in the ECOIN system. For simplicity, we will refer to both versions as COIN in this paper.

Corporate householding problems, especially entity aggregation problems, are very similar to traditional COIN applications in the sense that entity aggregation also involves different source and receiver contexts. Under different contexts, an entity may or may not need to be aggregated. For instance, as discussed in the motivational example, for the purpose of account consolidation using SEC's regulations, Lotus Development should be considered as part of IBM. However, in the source context, Lotus's revenue is not consolidated with IBM's. These differences in contexts can be captured in the COIN system.

Corporate Householding Query Mediation Example

In this section, we will present a design of the corporate householding query processor, and explain how the COIN technology can help mediate corporate householding queries to better meet users' needs. We will also show the results of query mediation and execution from a live demo. The demo application follows the motivational example in the "Motivational Example" subsection, with some slight modifications. The purpose of this example is to demonstrate the working of the system, which is scalable and flexible enough to be extended to more complex cases.

Figure 5 illustrates how corporate householding knowledge is captured and elevated in COIN; Figure 6 illustrates the differences between contexts, and what happens when a sample query is asked in different receiver's contexts. The following sections will describe each of these components (e.g., ontology, elevations, contexts) in detail.

Ontology Framework

The first thing we need to do is to specify the domain model (ontology framework) for the sample corporate householding problem, as shown in the top part of Figures 5 and 7. The semantic types are divided into two categories—corporate structure-related and task-related. Corporate structure-related semantic types represent common concepts in corporate group structure and entity aggregation, and thus are useful in any entity aggregation problems; the task-related semantic types shown here are specific to the account consolidation example we are considering. This ontology can be extended easily to accommodate entity aggregation problems in other application areas by substituting the current task-related semantic types with a set of new task-related types and setting appropriate relationships across the two categories of semantic types.

Three kinds of arrows in Figure 7 represent the "inheritance," "attributes," and "modifiers" relationships, respectively.

- *Inheritance:* the classic type of "is-a" relationship. All semantic types root from one semantic type—"Basic," which includes system native types such as inte-

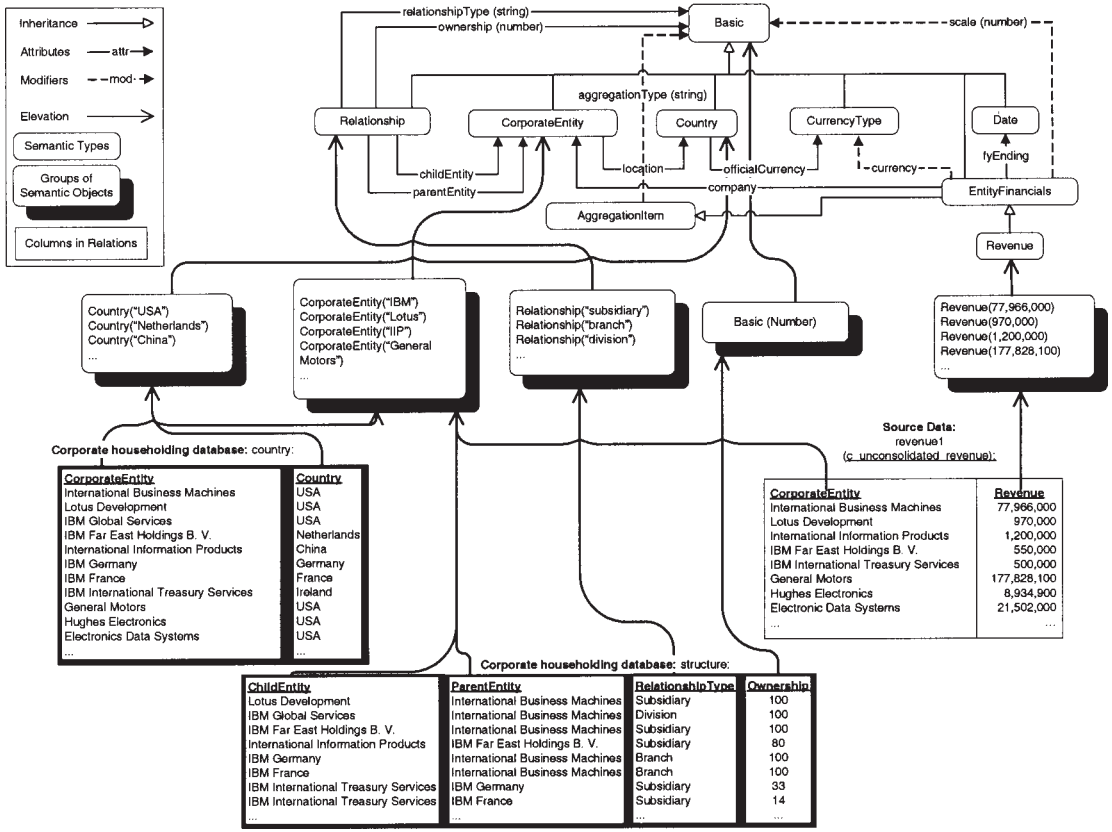


Figure 5. Summary of Ontology, Relations, and Elevations

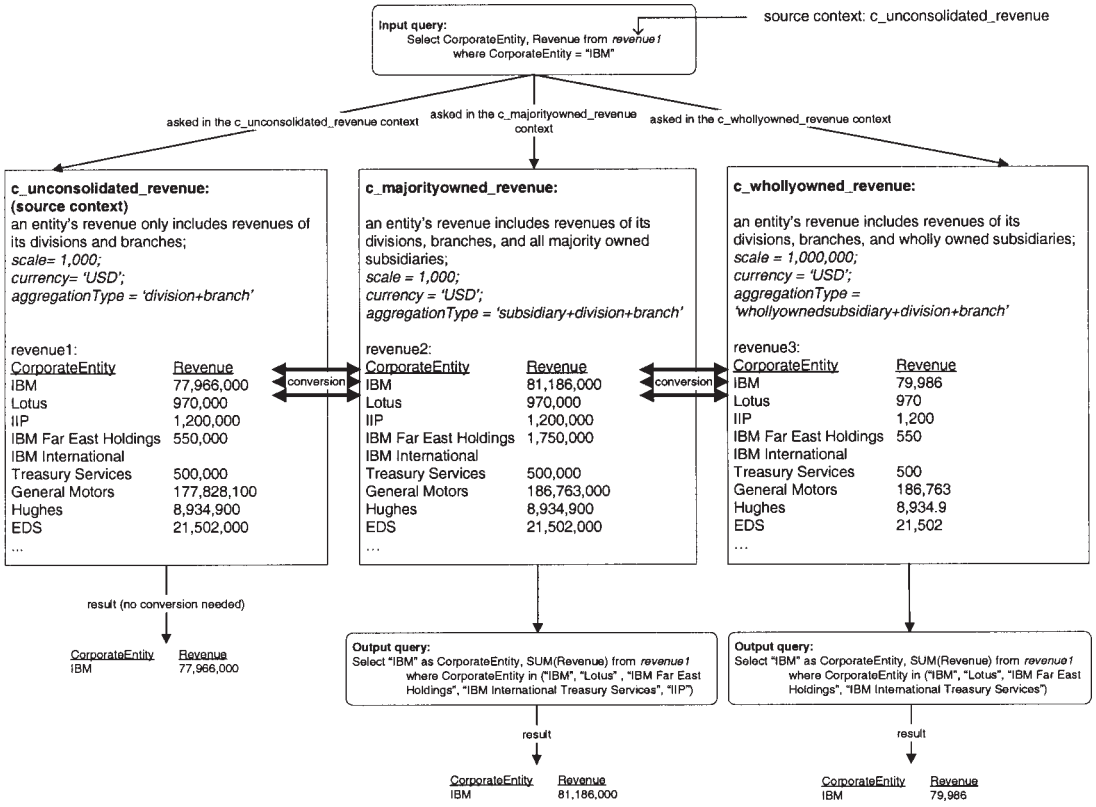


Figure 6. Summary of Contexts and Queries

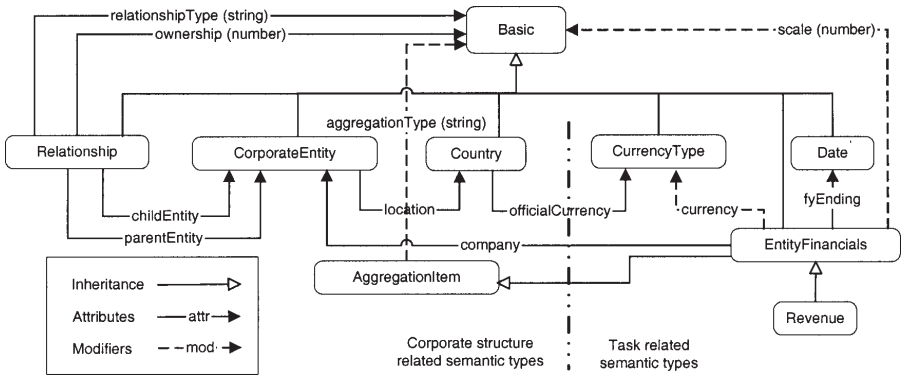


Figure 7. Ontology for the Motivational Example on Account Consolidation

gers, strings, and real numbers. If type B inherits from type A, B is a subtype of A and inherits all A's properties and attributes. For example, in Figure 7, "Revenue" inherits from "EntityFinancials"; thus it automatically has modifiers, such as "currency" (which represents which currency the financial data is in) and attributes, such as "fyEnding" (which represents the ending date of the fiscal period associated with this piece of financial data).

- *Attributes:* used to represent the structural properties of semantic types. In other words, they define relationships between objects of corresponding semantic types. For example, the semantic type "CorporateEntity" has an attribute called "location" of type country. This attribute represents the country of incorporation of a corporate entity. The semantic type "Relationship" has attributes "parentEntity," "childEntity," "relationshipType," and "ownership." The "parentEntity" (of type CorporateEntity) owns the "childEntity" (also of type CorporateEntity) with ownership percentage equal to the value of "ownership." The types of relationship between child and parent entities include subsidiary, branch, and division, and they are captured by the attribute "relationshipType."
- *Modifiers:* special attributes whose values vary depending on the context and whose values determine the interpretations of data. Modifiers are used in conflict detection during query mediation. For instance, the modifier "currency" has value "USD" in a U.S.-based context, and value "GBP" in a U.K.-based context. The modifier "aggregationType" has value "division+branch" in an unconsolidated revenue context, and value "subsidiary+division+branch" in a consolidated revenue context based on the SEC rules.

Corporate Structure-Related Semantic Types and Data

Semantic Types. The semantic types in this category are closely associated with representations of entity aggregation, corporate group structure, and relationships between corporate entities:

- CorporateEntity: inherits from Basic. This semantic type has the attribute *location* of type Country, which specifies the corporate entity's country of incorporation or its location. Some sample values that CorporateEntity may take are "Johnson & Johnson" and "Citibank Canada"; some sample values for "Country" are "USA" and "Canada."
- AggregationItem: inherits from Basic. It is a super-type of any specific item that is being aggregated. These specific items are semantic types in the task-related domain, such as EntityFinancials in the current ontology. Other subtypes of AggregationItem may include Employee, Customer, or CreditRisk, depending on the task at hand. The modifier *aggregationType* specifies how the items should be aggregated. Suppose that the aggregation rule in the context concerned is to aggregate all divisions, branches and wholly owned subsidiaries with their parents. The value of aggregationType here is therefore "whollyownedsubsidiary+ division+branch."
- Country: inherits from Basic and has an attribute *officialCurrency* of type CurrencyType, which captures the official currency type of the country concerned.
- Relationship: inherits from Basic and has attributes *relationshipType* of type Basic, *ownership* of type Basic, and *parentEntity* and *childEntity* of type CorporateEntity. For instance, we know that Lotus is a subsidiary 100 percent owned by IBM. This relationship can be represented as "childEntity = Lotus, parentEntity = IBM, relationshipType = Subsidiary, ownership = 100."

Data. In order to perform corporate householding, we need information about corporate structures. Part of the desired relation ("structure")[4] is shown in Table 1.

The columns in Table 1 are self-explanatory. For example, IBM Far East Holdings B.V. is a wholly owned subsidiary of IBM, and International Information Products is 80 percent owned by IBM Far East Holdings. The "ownership" column describes the percentage ownership of the "ParentEntity" on the "ChildEntity," and it can take values up to 100 (wholly owned subsidiaries or divisions or branches). For demonstration purposes, we have implemented the structure table as an inline database table, that is, a set of rule statements in the abduction code. Only a minor change in the code is needed if database calls are implemented and added in the future. A sample statement that defines one row of Table 1 follows:

```
rule(structure("Lotus Development," "International Business Machines,"
"Subsidiary," 100), (true)).
```

Another piece of information we need is the country of incorporation or location table. We name this relation "country," and part of this relation is as shown in Table 2.

The above two relations—"structure" and "country"—are generic across all contexts; in other words, no matter what the purpose of the query is, the data from these two tables will be used and they will not change. We may also need other data sources to derive answers to the questions asked in the decision rules that are context-specific. For example, in the context of "c_majorityowned_revenue," according to the decision tree in the "Motivational Example" section, we will need to know informa-

Table 1. Relation “Structure”: Pairs of Related Corporate Entities and Details About Their Relationship

ChildEntity	ParentEntity	RelatioshipType	Ownership
IBM Credit Corp.	International Business Machines	Subsidiary	100
Lotus Development	International Business Machines	Subsidiary	100
IBM Far East Holdings B.V.	International Business Machines	Subsidiary	100
International Information Products	IBM Far East Holdings B.V.	Subsidiary	80
IBM Global Services	International Business Machines	Division	100
IBM Germany	International Business Machines	Branch	100
IBM France	International Business Machines	Branch	100
IBM Finland	International Business Machines	Branch	100
IBM Denmark	International Business Machines	Branch	100
IBM Switzerland	International Business Machines	Branch	100
IBM International Treasury Services	IBM Germany	Subsidiary	33
IBM International Treasury Services	IBM France	Subsidiary	14
IBM International Treasury Services	IBM Finland	Subsidiary	10
IBM International Treasury Services	IBM Denmark	Subsidiary	18
IBM International Treasury Services	IBM Switzerland	Subsidiary	25
Hughes Electronics	General Motors	Subsidiary	100

Table 2. Relation "Country": Country of Incorporated or Location of Corporate Entities

CorporateEntity	Country
International Business Machines	United States
Lotus Development	United States
IBM Far East Holdings B.V.	Netherlands
International Information Products	China
IBM Germany	Germany
IBM France	France
IBM Finland	Finland
IBM Denmark	Denmark
IBM Switzerland	Switzerland
IBM International Treasury Services	Ireland
General Motors	United States
Hughes Electronics	United States

tion on (1) a company's controlling financial interest on the other, (2) if a company is a bank holding company and if it is subject to the Bank Holding Company Act, and (3) if two entities have the same fiscal period. Here, we have simply assumed that IBM has controlling financial interest on its subsidiaries, it is not a bank holding company, and they share the same fiscal period end date. Therefore, the types of relationships and ownership percentages are what determine the aggregation between entities.

Task-Related Semantic Types and Data

Semantic Types. The task-related part of the ontology includes the semantic types related to a specific task, that is, one of the corporate householding application areas. Different kinds of task-related components could be added onto the current ontology model when different problem domains are considered. Here, we include only some of the semantic types in the Company Financials domain, those that are closely related to our "total revenue" example.

- EntityFinancials: inherits from AggregationItem, and encapsulates the representations of a corporate entity's financial information. It has attribute *fyEnding,* as well as modifiers *company, currency,* and *scale.* For example, the fact that Entity A's fiscal year ends on December 31 and its financial data is in thousands USD is represented by "entity = Entity A, fyEnding = 12/31, currency = USD, scale = 1000." Because EntityFinancial is a subtype of AggregationItem, it inherits the modifier *aggregationType.*
- Revenue: inherits from EntityFinancials, and thus inherits all its attributes and modifiers by default. It captures a corporate entity's revenue data.
- CurrencyType and Date: inherit from Basic. They help to define EntityFinancials, and could be shared by other problem domains.

In the following sections, we will explain how the COIN technology and the ontology framework for the corporate householding problem described in previous sections can help mediate (i.e., rewrite) the query to better meet users' needs. The purpose of this example is to demonstrate the working of the system, which we believe is scalable and flexible enough to be extended to more complex cases with a reasonable amount of add-ons and more specification.

Data. We have already presented the key needed task relation in Figure 1, that is, the "revenue1" (in the c_unconsolidated_revenue context) table. In reality, this table could be the result of a join on multiple data sources. We assume: (1) the "Revenue" corresponding to "CorporateEntity = IBM" does not include the revenue from any of IBM's subsidiaries, such as Lotus Development, International Information Products, IBM Far East Holdings B.V., and IBM International Treasury Services, but includes revenues from all divisions and branches; (2) all the entities have the same fiscal periods ending on December 31 and the data is for the year 2002; and (3) IBM consolidates the revenues from its foreign subsidiaries.

The Contexts and Rules

Recall the example described in the "Motivational Example" section. Sally wants to find out IBM's total revenue in fiscal year 2002. We name the context of the data source Sally uses "c_unconsolidated_revenue," and Sally's own context "c_majorityowned_revenue" (because Sally uses accounting rules from the SEC, which require consolidation of majority-owned subsidiaries). Another possible context, that we will consider, is "c_whollyowned_revenue," which requires account consolidation of only wholly owned subsidiaries. The three boxes in the center section of Figure 6 summarize the three contexts. Table 3 compares the differences and similarities among these contexts.

Similar to the modifier values as shown above, rules (decision trees) are defined per context as well. For instance, if Sally is in the c_majorityowned_revenue context, the purpose of Sally's query is to find out the "total revenue" of a company using the SEC's consolidation rules, and the tree that represents the rules in this context is shown in Figure 2. Here, the example is simplified such that the value of the modifier "aggregationType" captures the rules in different contexts.

Modifiers

Table 3 summarizes the values that the modifiers take in the three contexts. For example, the modifier *scale* is 1,000 in the first two contexts, but is 1 million in the c_whollyowned_revenue context. This means that the actual revenue figures in different contexts may differ by a factor of 1,000. The definitions of these modifiers in COINL[5] look like follows (using the c_unconsolidated_revenue context as an example):

Table 3. Context Comparison in the Motivational Example

Context Name	Revenue (of any corporate entity)	Scale	Currency	Aggregation Type
c_unconsolidated_revenue	Includes revenues of its divisions and branches only.	1,000	USD	division+branch
c_majorityowned_revenue	Includes revenues of its divisions, branches, and majority-owned subsidiaries+division+branch.	1,000	USD	subsidiary
c_whollyowned_revenue	Includes revenues of its divisions, branches, and wholly owned subsidiaries.	1,000,000	USD	whollyownedsubsidiary+ division+branch

modifier(
'EntityFinancials,' Object, aggregationType, c_unconsolidated_revenue, Modifier), (cste(basic, Modifier, c_unconsolidated_revenue, "division+branch"));

modifier(
'EntityFinancials,' Object, currency, c_unconsolidated_revenue, Modifier), (cste(CurrencyType, Modifier, c_unconsolidated_revenue, "USD"));

modifier(
'EntityFinancials,' Object, scale, c_unconsolidated_revenue, Modifier), (cste(basic, Modifier, c_unconsolidated_revenue, 1000)).

Every modifier corresponds to a potential conflict that may occur between the context "c_unconsolidated_revenue" and some other context. For example, the above clause states that the modifier "scale" for the object *Object* of type EntityFinancials in the "c_unconsolidated_revenue" context is the object *Modifier,* where *Modifier* is a constant (cste) of type Basic and value 1,000 in this context.

Conversion Functions

Conversion functions define how modifier values change between different contexts. In most cases, they are defined independent of any specific source or receiver context. During query mediation, the context mediator decides whether or not a conversion should be used. For example, the following is the internal representation of the conversion function between scales in different contexts:

```
cvt (EntityFinancials, _O, scale, Ctxt, Mvs, Vs, Mvt, Vt):
        Ratio is Mvs / Mvt,
        Vt is Vs * Ratio,
```

where scale is a modifier of semantic type EntityFinancials, and has value *Mvs* in the source context and value *Mvt* in the target receiver context. The value of *scale* for an object *_O* of type EntityFinancials in the receiver context (*Vt*) is equal to the value of scale for _O in the source context (*Vs*) multiplied by the *Ratio* of the modifier value in the source context to the modifier value in the receiver context. In our example, the ratio might be 1,000/1,000; 1,000,000/1,000; or 1,000/1,000,000—depending upon the contexts being considered.

The conversion functions that take care of the modifier *aggregationType* encapsulate the reasoning part of the aggregation process according to relationship types and ownership percentages. The conversion function makes calls to many helper functions in the abduction engine. Nevertheless, the reasoning steps can be described in words: first, the ownership percentages (directly or indirectly, regardless) of all the subsidiaries of the corporate entity concerned are calculated, through some recursive helper functions that are defined in the abduction engine; then, the function filters out those subsidiaries that are not majority-owned; finally, it specifies that the revenue in the receiver's context (i.e., c_majorityowned_revenue) should be the sum of the rev-

enue of the corporate entity in the source context (i.e., c_unconsolidated_revenue) and the discounted revenue of the majority-owned subsidiaries. There is a subtlety here. To illustrate a useful capability of the COIN system in the demo to follow, we assume that when adding the numbers together, our user Sally first discounts them using IBM's ownership percentages on these subsidiaries. For example, because International Information Products is only 80 percent owned by IBM, Sally would multiply International Information Product's revenue number by 80 percent before adding it to the total revenue of IBM. This is slightly different from what has been presented in the motivational example, but, nevertheless, it is another interesting and reasonable way of consolidating revenues. Using this slightly modified aggregation rule, Sally will get 80,946,000 as the total revenue of IBM in the context of c_majorityowned_revenue.

Elevation

The elevation axioms map the data and data relationships from the sources to the domain model. There are three steps involved in an elevation process:

1. define a virtual semantic relation corresponding to each relation in the previous section;
2. assign values to each semantic object according to the context of the source; and
3. map the semantic objects in the semantic relation to semantic types defined in the domain model.

The upward arrows in Figure 5 indicate how each column in the relations is elevated through semantic objects to semantic types in the ontology. Recall that the "revenue1" relation has two columns, CorporateEntity and Revenue. The elevated relation corresponding to "revenue1" for the context c_unconsolidated_revenue looks as follows (in the internal COINL).

```
revenue_p(
        skolem('CorporateEntity,' C1, c_unconsolidated_revenue, 1,
        revenue1(C1, C2)),
        skolem('Revenue,' C2, c_unconsolidated_revenue, 2, revenue1(C1, C2))).
```

The semantic relation "revenue_p" is defined on the semantic objects in the corresponding relation attributes. The columns in relation "revenue" are mapped to semantic objects, which have a unique object-id: the first column is mapped to 'CorporateEntity' and the second column is mapped to 'Revenue.' Similarly, we define other elevation axioms in COINL:

```
structure_p(6
        skolem('CorporateEntity,' C1, Ctxt, 1, structure(C1, C2, C3, C4)),
        skolem('CorporateEntity,' C2, Ctxt, 2, structure(C1, C2, C3, C4)),
        skolem('Relationship,' C3, Ctxt, 3, structure(C1, C2, C3, C4)),
        skolem(basic, C4, Ctxt, 4, structure(C1, C2, C3, C4)));
```

Table 4. Summary of Elevations from Relations to the Domain Model

Column	Semantic Type
revenue1.CorporateEntity	CorporateEntity
revenue1.Revenue	Revenue
structure.childEntity	CorporateEntity
structure.parentEntity	CorporateEntity
country.CorporateEntity	CorporateEntity
country.Country	Country

country_p(
skolem('CorporateEntity,' C1, Ctxt, 1, country(C1, C2)),
skolem('Country,' C2, Ctxt, 2, country(C1, C2))).

The elevation is summarized in Table 4.

Query Mediation

In this section, we will go through the steps in query mediation and execution using a demo application derived from the motivational example. Recall that the source context (the context that the data source "revenue1" uses) is c_unconsolidated_revenue, and the receiver context (the context that Sally is in) is c_majorityowned_revenue. Because Sally would like to find out what IBM's total revenue is according to her consolidation rules, she issues the following query on "revenue1":

Select CorporateEntity, Revenue from revenue1

where CorporateEntity = "International Business Machines."
[context= c_majorityowned_revenue],

the COIN system must convert this into the query:

Select "IBM" as CorporateEntity, SUM(Revenue) as Revenue from revenue1

where CorporateEntity in ("International Business Machines," "Lotus Development," "IBM Far East Holdings," "International Information Products," "IBM International Treasury Services").

Here, since ownership percentages are used to discount the revenue numbers in this example, the final result should be 80,946,000. As the first step after an SQL query is fed in, the COIN system generates an internal datalog query[7] as follows:

answer("International Business Machines," 'V1'):
revenue1("International Business Machines," 'V1').

Then, a context-sensitive datalog query is produced, using elevation axioms and contexts defined in above sections. This query ascertains that the result returned to the user has to be in the c_majorityowned_revenue context:

```
answer (‘V4,’ ‘V3’):-
        revenue1_p(‘V2,’ ‘V1’),
        value(‘V2,’ c_majorityowned_revenue, ‘V4’),
        ‘V4’ = “International Business Machines,”
        value(‘V1,’ c_majorityowned_revenue, ‘V3’).
```

The above unmediated query is then fed to the mediation engine, where conflicts are detected and resolved. The mediation process is based on an abduction engine, which takes the datalog query and the domain model axioms (such as the conversion function presented in the “Conversion Functions” section), and computes a set of abducted queries that have considered all the possible cases of conflicts. Modifier values in the source and receiver contexts, as well as the conversion functions between these two contexts are discovered.

The mediated datalog query produced by the context mediator is shown below and in Figure 8:

```
answer(“International Business Machines,” ‘V25’):-
        revenue1(“International Business Machines,” ‘V24’),
        revenue1(“Lotus Development,” ‘V23’),
        revenue1(“IBM Far East Holdings B.V.,” ‘V22’),
        revenue1(“International Information Products,” ‘V21’),
        ‘V20’ is ‘V21’ * 80, ‘V19’ is ‘V20’ / 100, ‘V18’ is ‘V19’ + ‘V22,’
        ‘V17’ is ‘V23’ * 100, ‘V16’ is ‘V18’ * 100, ‘V15’ is ‘V17’ + ‘V16,’
        ‘V14’ is ‘V15’ / 100,
        revenue1(“IBM International Treasury Services,” ‘V13’), ‘V12’ is
        100 * ‘V13,’ ‘V11’ is ‘V12’ * 33,
        ‘V10’ is ‘V12’ * 14, ‘V9’ is ‘V12’ * 10, ‘V8’ is ‘V12’ * 18, ‘V7’ is
        ‘V12’ * 25, ‘V6’ is ‘V11’ + ‘V10,’
        ‘V5’ is ‘V6’ + ‘V9,’ ‘V4’ is ‘V5’ + ‘V8,’ ‘V3’ is ‘V4’ + ‘V7,’ ‘V2’ is
        ‘V3’ / 10000, ‘V1’ is ‘V14’ + ‘V2,’
        ‘V25’ is ‘V1’ + ‘V24.’
```

If the above mediated datalog query is expanded by substituting values (e.g., “V31,” “V32”) with more meaningful notations such as Revenue(“Lotus Development”) and Revenue(“International Business Machines”), we get the following equation:

```
Revenue of IBM      = R(IBM)
(in c_majorityowned + R(Lotus)*100%
_revenue context)   + R(International Information Products)*100%*80%
                    + R(IBM Far East holdings)*100%
                    + R(IBM International Treasury Services)*100%*33%
                    + R(IBM International Treasury Services)*100%*14%
                    + R(IBM International Treasury Services)*100%*10%
                    + R(IBM International Treasury Services)*100%*18%
                    + R(IBM International Treasury Services)*100%*25%,
```

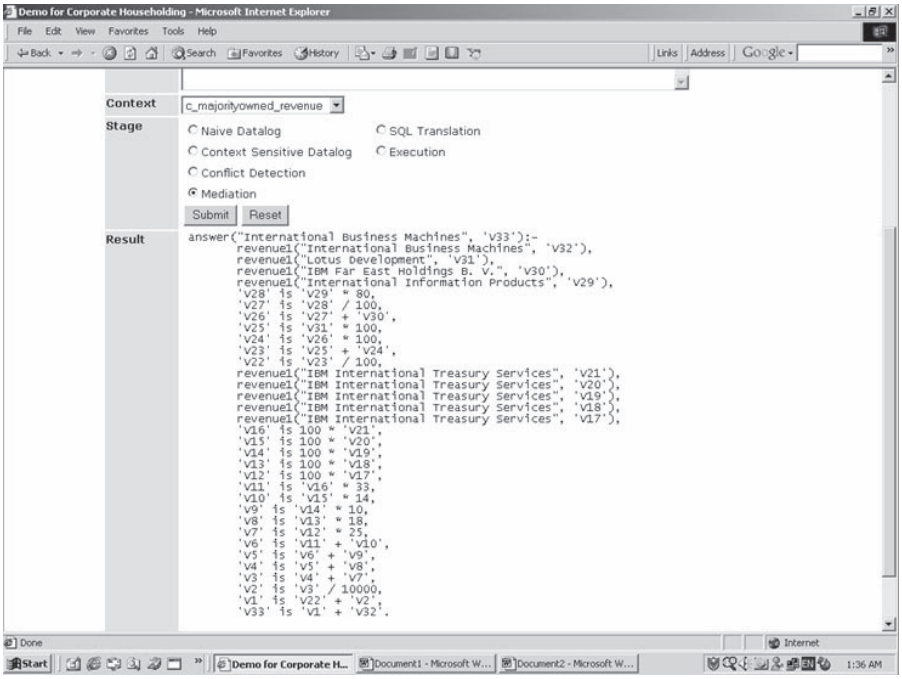


Figure 8. Demo (1)—The Mediated Datalog Query

where R(X) denotes the revenue of entity X in the revenue1 table (i.e., in the c_unconsolidated_revenue context). This equation verifies that the mediated query does give the desired sum of revenues, discounted by their ownership percentages.

After the mediated datalog query is generated, it is translated to an SQL statement (shown in Figure 9) through a query planner and optimizer. This SQL statement, unlike the original input query, takes into account the differences between source and receiver contexts and will return a result in the receiver's context.

Finally, this SQL query is performed on the data source "revenue1," and IBM's correct total revenue is returned as shown in Figure 10.

Conclusions

IN THIS PAPER, WE BRIEFLY EXPLAINED the importance of improving data quality by addressing the challenge of corporate householding. We described categories of corporate householding problems and illustrated a few applications areas with examples derived from corporate householding knowledge research. Then we presented a motivational example in account consolidation. Following that, we described the COntext INterchange technology that performs mediated data access among heterogeneous data sources. By extending the COIN model, we developed a technical solution to an important type of corporate householding problem—entity aggregation—and demonstrated the concept by going through the design and implementation of an applica-

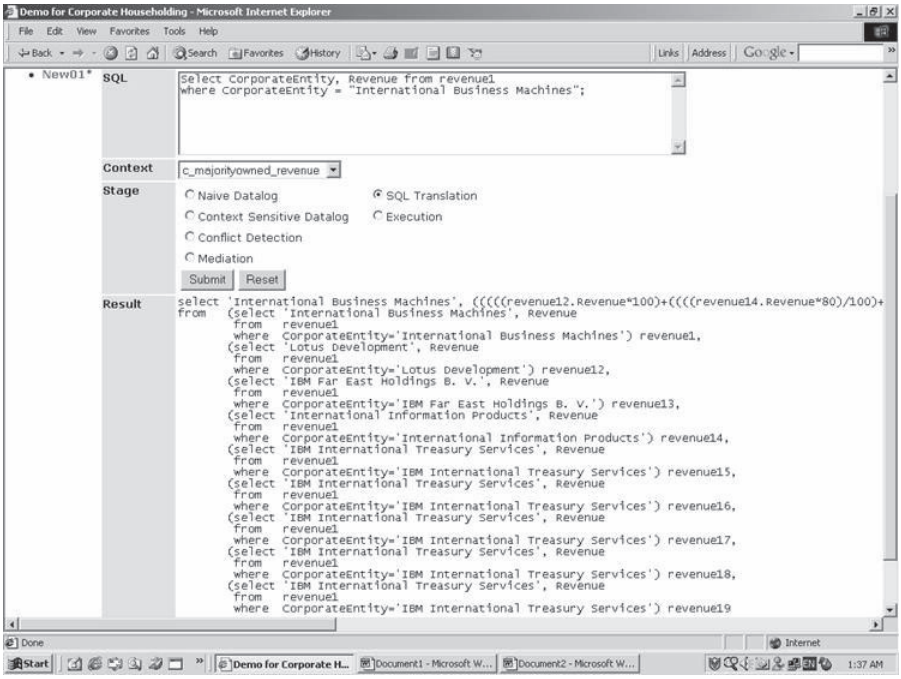


Figure 9. Demo (2)—Result of SQL Translation

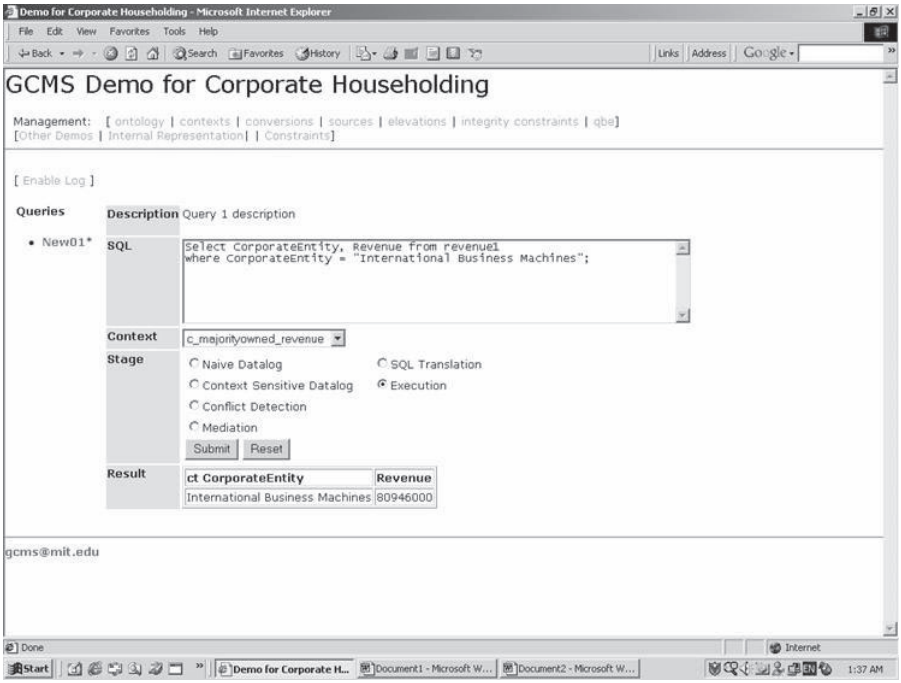


Figure 10. Demo (3)—Result of Execution

tion derived from the motivational example. The corporate householding query processor needs to be further improved and extended to serve more areas of application—but the feasibility of this approach has been demonstrated.

Acknowledgments: Work reported herein has been supported, in part, by Cambridge Research Group (CRG), D&B, FirstLogic, Naval Inventory Control Point (NAVICP), Singapore-MIT Alliance (SMA), and Total Data Quality Management (TDQM) Program. Helpful suggestions from Xing Ping Chen, Krishna Chettayar, Frank Dravis, James Funk, Raissa Kats-Haas, Chris Haywood, Cindy Lee, Yang Lee, Pat McCoy, Ahmad Shuja, and Wei Zhang are greatly appreciated.

NOTES

1. For a complete list of IBM's subsidiaries, please refer to IBM's annual report on SEC filings, available at www.sec.gov/Archives/edgar/data/51143/000104746903008194/a2102367zex-21.htm.

2. For a complete list of IBM's subsidiaries, see note 1.

3. According to the Bank Holding Company Act, a bank holding company is "any company [that] has control over any bank or over any company that is or becomes a bank holding company by virtue of this Act." Any company has control over a bank or over any company if: (A) the company directly or indirectly or acting through one or more other persons owns, controls, or has power to vote 25 per centum or more of any class of voting securities of the bank or company; (B) the company controls in any manner the election of a majority of the directors or trustees of the bank or company.

4. The data in this table is extracted from Exhibit 20.01 in IBM's annual report for the year ending December 31, 2002. See note 1.

5. COINL is a logical programming language based on F-logic, which is based on Prolog. Although we show the underlying COINL representations, there is a user-friendly tool [9] that automatically generates the COINL code shown.

6. As noted before, for demonstration purposes, "structure" is coded as facts, so the database table and its elevation are not used in the current implementation.

7. Datalog query representation is used internally in COIN.

REFERENCES

1. Becerra-Fernandez, I., and Sabherwal, R. Organizational knowledge management: A contingency perspective. *Journal of Management Information Systems, 18,* 1 (Summer 2001), 23–56.

2. Bielecki, T.R., and Rutkowski, M. *Credit Risk: Modeling, Valuation and Hedging.* Berlin: Springer-Verlag, 2002.

3. Bressen, S.; Goh, C.H.; Levina, N.; Shah, A.; Madnick, S.; and Siegel, M. Context knowledge representation and reasoning in the context interchange system. *Applied Intelligence: The International Journal of Artificial Intelligence, Neutral Networks, and Complex Problem-Solving Technologies, 12,* 2 (2000),165–179.

4. Epstein, D.G. *Bankruptcy and Related Law in a Nutshell.* St. Paul, MN: West Group, 2002.

5. Firat, A.; Madnick, S.; and Grosof, B. Knowledge integration to overcome ontological heterogeneity: Challenges from financial information systems. Paper presented at the International Conference on Information Systems (ICIS). Barcelona, Spain, December 2002.

6. Firat, A.; Madnick, S.; and Grosof, B. Financial information integration in the presence of equational ontological conflicts. Paper presented at the Workshop on Information Technology and Systems (WITS). Barcelona, Spain, December 2002.

7. Goh, C.H.; Bressan, S.; Madnick, S.; and Siegel, M. Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transactions on Office Information Systems, 17,* 3 (July 1999), 270–293.

8. Gold, A.H.; Malhotra, A.; and Segars, A.H. Knowledge management: An organizational capabilities perspective. *Journal of Management Information Systems, 18,* 1 (Summer 2001), 185–214.

9. Lee, P. Metadata representation and management for context mediation. Composite Information Systems Laboratory Working Paper No. 2003–01, MIT, Cambridge, MA, 2003.

10. Madnick, S. Metadata Jones and the Tower of Babel: The challenge of large-scale semantic heterogeneity. In R. Coyne, M. Jones, and B. Kobler (eds.), *Proceedings of the 1999 IEEE Meta-Data Conference.* Los Alamitos, CA: IEEE Computer Society Press (available at www.llnl.gov/liv_comp/metadata/md99/md99.html).

11. Madnick, S., and Wang, R.Y. Corporate household knowledge processing: Challenges, concepts, and solution approaches. Sloan Working Paper #4222–01, Massachusetts Institute of Technology and Composite Information Systems Laboratory Working Paper #2001–09, MIT, Cambridge, MA, 2001.

12. Madnick, S.; Wang, R.Y.; and Zhang, W. A framework for corporate householding. In C. Fisher and B. Dabidson (eds.), *Proceedings of the Seventh International Conference on Information Quality.* Cambridge, MA: MIT Total Quality Management Program, 2002, pp. 36–40.

13. Madnick, S.; Wang, R.Y.; Dravis, F.; and Chen, X. Improving the quality of corporate household data: Current practices and research directions. In E. Pierce and R. Katz-Haas (eds.), *Proceedings of the Sixth International Conference on Information Quality.* Cambridge, MA: MIT Total Data Quality Management Program, 2001, pp. 92–104.

14. Madnick, S.; Wang, R.Y.; Chettayar, K.; Dravis, F.; Funk, J.; Katz-Haas, R.; Lee, C.; Lee, Y.; Xian, X.; and Bhansali, S. Exemplifying business opportunities from corporate household research. *Advances in Management Information Systems (AMIS),* forthcoming 2004.

15. Shankaranarayan, G.; Ziad, M.; and Wang, R.Y. Managing data quality in dynamic decision environments: An information product approach. *Journal of Database Management Systems, 14,* 4 (2003), 14–32.

16. Wang, R.Y., and Strong, D.M. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems, 12,* 4 (Spring 1996), 5–32.

17. Wang, R.Y., and Madnick, S. Evolution towards strategic applications of data bases through composite information systems. *Journal of Management Information Systems, 5,* 2, (Fall 1988), 5–22.

18. Wang, R.Y.; Allen, T.; Harris, W.; and Madnick, S. An information product approach for total information awareness. Paper presented at the IEEE Aerospace Conference. Big Sky, Montana, March 2003.

Copyright of Journal of Management Information Systems is the property of M.E. Sharpe Inc. and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.