Conditional Mapping in Data Mediation

Paul L. Bergstein and Vishal Shah
Dept. of Computer and Information Science
University of Massachusetts Dartmouth, Dartmouth MA
{pbergstein | g_vshah}@umassd.edu

Abstract

A serious problem facing many organizations today is the need to share information among systems that have been developed separately. Conflicts in the structure and semantics of these disparate data sources create major obstacles to effective sharing. One approach to solving this problem is data mediation. The data mediation approach relies on a common ontology that can be used to describe the structure and semantics of each of the systems that wish to participate in the information sharing.

In our implementation, the common ontology is expressed as a shared conceptual schema, and we use XML to map data elements of real databases onto the conceptual schema. We have found that these mappings sometimes depend on the value of a data element or the value of some other element of the same tuple. Interestingly, the mapping may also depend on the values of data elements in the (unpopulated) conceptual schema.

1. Introduction

A serious problem facing many organizations today is the need to share information among systems that have been developed separately. The information sharing may be within the organization or with external partners. In either case, the heterogeneity of the data creates major obstacles to effective sharing of information. Conflicts may exist in both the structure and the semantics of the data involved. Furthermore, the structure and semantics of a data source may change over time.

Historically, there have been a variety of approaches to this problem [1-4]. The simplest approach is to build a messaging system for each pair of data sources that wish to exchange data. The messaging system translates data to and from the agreed message format at each end. However this approach doesn't scale well if there are many systems that want to participate in the sharing, since a messaging system is needed for each pair. There is another problem as well. The metadata documenting the structure and semantics of an enterprise's data that is required to build the messaging system is a very valuable resource, but it may get lost in the translating code.

Another approach is to define standards. The standardization approach takes several forms. For example, we could standardize the data sources, making the data homogeneous. While seemingly simple, this approach has proven impossible in practice. Since different data sources are designed to be used in different environments, they are heterogeneous for good reasons, and nobody can agree on a common standard.

Standardizing the message format is another possibility. This approach is not new, but has recently been receiving widespread attention in the form of defining standard DTD's for exchanging data in XML format [5]. Given the level of effort in this direction, we expect to see quite a bit of success, especially within limited and well defined domains. On the other hand, prior attempts to define standard message formats have generally failed due to lack of agreement on the format's structure and semantics. Note that agreement to use XML does not solve this problem. It is still necessary to agree on the structure (what tags to use) and the semantics (what the tags mean). Also, two systems exchanging information through a standard message format may lose information

and/or precision during the exchange that could have been preserved using a custom format.

The data mediation approach relies on a common ontology that can be used to describe the structure and semantics of each of the systems that wish to participate in the information sharing. A data mediator uses these descriptions to perform any necessary translation between systems exchanging information. In a variation of this approach, a shared view is created, and the mediator translates queries written against the shared view. This approach has the advantages that there is no need to agree on standard formats, the metadata is made explicit (so it may be reused), and translations only occur where the structure or semantics between two systems differ. In many situations, we believe that mediation will prove to be a better approach than standardization.

In the next section we will describe some of our experiences in implementing a prototype data mediator for relational databases. Then we will focus on a feature we have recently implemented called conditional mapping.

2. Background

In our implementation, the common ontology is expressed as a shared conceptual schema, which includes both ordinary classes (e.g. University, Student) and domain classes (e.g. Money, Date). The ordinary classes have domain classes as their attribute types. For each domain class, we specify subclasses (subdomains) for the known representations. The mediator uses a repository of functions for converting between representations within a domain.

We use XML[1] to map data elements of real databases onto attributes of ordinary classes in the conceptual schema. Each mapping to an attribute of an ordinary class includes the subdomain of the data element. For each data source there is a separate XML file prepared by someone familiar with that data source.

Our data mediator is based on the following scenario: Suppose a user who knows the schema of only their local database, System A, wishes to retrieve information from a foreign database, System B. They write a query against the schema of System A, but indicate that they would like to use System B as the data source. The mediator translates the query against System A into one or more queries against System B, executes the queries, and translates the results into the local format of System A.

In the simplest case each data element of System A corresponds one-to-one with an element of the conceptual schema, which in turn corresponds one-to-one with an element of System B. If the conceptual schema contains an ordinary class called Employee with a salary attribute of type Salary, and System A has a Worker relation with a pay-rate attribute, then the XML file for System A would map Worker/pay-rate to Employee/salary and it would also map pay-rate to one of the subdomains of Salary such as Annual/USDollars or Monthly/Euros[2]. Similar mappings from System B provide the mediator with the information needed for translation.

From the beginning, we realized that the mappings between the conceptual schema and an actual database would not always be one-to-one. Suppose that in the conceptual schema Professor's have a phone-number attribute of type PhoneNumber, but in the actual database Instructor's have area-code, exchange, and extension attributes. For the mediator to work, the PhoneNumber domain class must have a subdomain, say ACEE, for the area-code/exchange/extension representation of phone numbers, with attributes corresponding to the three parts of a phone number. Each of the area-code, exchange, and extension attributes is mapped to the Professor/phone-number attribute (and also to the appropriate attribute of the ACEE subdomain). The mapping (from actual to conceptual) is many-to-one.

If another database uses the same representation of phone numbers, so we have mappings like:

A: (code, exchg, ext) → phone-number
B: (area, exg, extension) → phone-number

[1] For brevity, in this paper we mostly describe mappings without showing the XML syntax since the XML is trivial but verbose.

[2] In theory, the issues of currency units and frequency of payment should be separate, but we combine them for the sake of simplicity in our implementation, in order to focus on more interesting concerns.

then the translation will not use conversion functions. In other cases, such as:

A: (latitude, longitude) → position
B: (point, bearing, range) → position

a conversion function is required. In our (java) implementation all conversion functions have the same interface. They take a java Properties object as parameter, and return a Properties object as the result, so they support many-to-many conversions.

We also support many-to-many mappings between conceptual and actual schemas. Suppose the conceptual schema has latitude and longitude[3] and the actual database has point, bearing, and range. Now the mapping is many-to-many even before the other actual schema is involved.

3. Conditional Mapping

Notice that the many-to-one and many-to-many mappings discussed above involve only logical and's. We have (latitude → position) *and* (longitude → position) in our XML file. Recently, we have been investigating problems that occur when an actual schema isn't a good match for the conceptual schema. This is likely to happen, for example, when a new data source is added after the conceptual schema has been completed. We soon discovered that we needed the equivalent of logical or's as well.

Consider a conceptual schema with separate classes for graduate and undergraduate students; a database, A, with the same structure as the conceptual schema; and a database, B, with a single table for all students. If a user executes a simple query such as "*select gpa from student*" written according to schema B with database A as the data source, the mediator simply runs two queries against database A and combines the results: "*select gpa from g_student*" and "*select gpa from u_student*".

The trouble arises when we try to execute a query such as "*select gpa from u_student*" written according to schema A with database B as data source. The mediator needs to add a where clause when it translates the query, but the mappings discussed up until now are not sufficient to support the translation. What we need to express is that in database B student maps either to g_student or u_student depending on the student's degree program. In our implementation we accomplish this by using mappings (in XML) to express the following two rules:

(student/gpa → u_student/gpa
  when degree = BS or BA)
(student/gpa → g_student/gpa
  when degree = MS or MA or PHD).

In this example there are two mutually exclusive one-to-one mappings. If database B used the area-code/exchange/extension representation of phone numbers and the conceptual schema had simply phone, we would have two mutually exclusive three-to-one mappings expressed by the following 6 rules:

(student/area-code → u_student/phone
  when degree = BS or BA)
(student/exchange → u_student/phone
  when degree = BS or BA)
(student/extension → u_student/phone
  when degree = BS or BA)
(student/area-code → g_student/phone
  when degree = MS or MA or PHD)
(student/exchange → g_student/phone
  when degree = MS or MA or PHD)
(student/extension → g_student/phone
  when degree = MS or MA or PHD)

Next, suppose that the conceptual schema has a single student class, so that it matches database B instead of database A. Now all of the mappings to the conceptual schema become unconditional. All undergraduate students map to student and all graduate students map to student, yet the mediator still needs to add a where clause when it translates the query "*select gpa from u_student*" written according to schema A with database B as data source.

Our solution is to use two kinds of conditions. Whereas we previously thought of mappings as simply *between* elements of the actual and conceptual schemas, we now view them as *to* and *from* the conceptual schema, and we specify separate conditions on mappings in the to and from directions in our XML files. In this example, we only need from conditions. It is interesting to observe that we are specifying

[3] This is a stretch to come up with an example. It would be more appropriate for the conceptual schema to have simply position, with latitude/longitude one of the possible representations.

conditions on data in the conceptual schema (expressed in terms of the actual schema) even though the conceptual schema is purely conceptual, i.e. it is never populated.

When writing a query using schema A to get data from source B, the mediator uses the *to* conditions from B's mappings and the *from* conditions from A's mappings to formulate a where clause in the query executed against database B[4]. Both to and from conditions are written only in terms of the actual schema being mapped, so each individual doing the mapping only needs to know their own schema and the conceptual schema.

So far we have seen examples that specify conditions on mappings either in the to direction or the from direction, but in some cases it is necessary to specify conditions in both directions. Consider a conceptual schema that has full-time students and part-time students, and an actual schema with graduate students and undergraduate students. We would need the mappings in Listing 1, given in XML syntax.

4. Conclusions

There are numerous other researchers [1-10] who have investigated mediation as a way of resolving structural and semantic conflicts between data sources. However, as far as we can determine, there are no previous reports of conditional mappings as described here. The implementation of conditional mapping has greatly increased the number of queries that our mediator can handle correctly, and we are now able to successfully mediate most simple SQL queries that don't involve joins or sub-queries.

```
<mapping>
    <actual class="u_student" attribute="gpa" />
    <conceptual class="ft_student" attribute="gpa">
        <domain class="zero2four" attribute="score" />
    </conceptual>
    <to-condition>
        status = 'regular'
    </to-condition>
    <from-condition>
        degree = 'BA' or degree = 'BS'
    </from-condition>
</mapping>

<mapping>
    <actual class="u_student" attribute="gpa" />
    <conceptual class="pt_student" attribute="gpa">
        <domain class="zero2four" attribute="score" />
    </conceptual>
    <to-condition>
        status = 'part time'
    </to-condition>
    <from-condition>
        degree = 'BA' or degree = 'BS'
    </from-condition>
</mapping>

<mapping>
    <actual class="g_student" attribute="gpa" />
    <conceptual class="ft_student" attribute="gpa">
        <domain class="zero2four" attribute="score" />
    </conceptual>
    <to-condition>
        status = 'regular'
    </to-condition>
    <from-condition>
        degree = 'MA' or degree = 'MS'
        or degree = 'PHD'
    </from-condition>
</mapping>

<mapping>
    <actual class="g_student" attribute="gpa" />
    <conceptual class="pt_student" attribute="gpa">
        <domain class="zero2four" attribute="score" />
    </conceptual>
    <to-condition>
        status = 'part time'
    </to-condition>
    <from-condition>
        degree = 'MA' or degree = 'MS'
        or degree = 'PHD'
    </from-condition>
</mapping>
```

Listing 1

[4] In some cases, a where clause is not sufficient, and the mediator must filter data as it translates into the format of schema A.

We are currently working on enhancing the mediator so that it can handle queries involving joins. We are particularly interested in cases where relations have been decomposed differently in the local and foreign databases, so that the joins required in the foreign database are different than those specified for the local database. We are also working on the development of JDBC drivers for the mediator, so that any java application can be readily adapted to use different data sources through mediation.

5. References

[1] E. Sciore, M. Siegel, and A. Rosenthal, "Using Semantic Values to Facilitate Interoperability Among Heterogeneous Information Systems", *ACM Transactions on Database Systems*, vol. 19(2), June 1994, pp. 254-290.

[2] G. Wiederhold, "Mediators in the Architecture of Future Information Systems", Readings in Agents, Eds. M. N. Huhns and M. P. Singh, San Francisco, CA, USA: Morgan Kaufmann, 1997, pp. 185-196.

[3] P. B. Lowry, "XML data mediation and collaboration: A proposed comprehensive architecture and query requirements for using XML to mediate heterogeneous data sources and targets," *34th Annual Hawai'i International Conference On System Sciences (HICSS)*, Mauii, Hawaii, January 3-6, 2001, pp. 2535-2543.

[4] C. H. Goh, S. Bressan, S. Madnick, and M. Siegel, "Context interchange: new features and formalisms for the intelligent integration of information", *ACM Transactions on Information Systems*, vol. 17(3), July 1999, pp. 270.

[5] L. S. Seligman and A. Rosenthal, "XML's Impact on Databases and Data Sharing", IEEE Computer, vol. 34(6), 2001, pp. 59-67.

[6] G. Neugebauer, "GLUE – Using Heterogeneous Sources of Information in a Logic Programming System", *Proceedings of the KI'97 Workshop on Intelligent Information Integration*, Freiburg, 1997.

[7] L. Serafini and F. Giunchiglia and F. Mylopoulos and P. Bernstein, "The Local Relational Model: A Logical Formalization of Database Coordination", *Proceedings of CONTEX'03*, 2003.

[8] H. Wache and H. Stuckenschmidt, "Practical Context Transformation for Information System Interoperability", *Lecture Notes in Computer Science*, vol. 2116, 2001, p. 367.

[9] B. Ludäscher, A. Gupta, and M. Martone, "Model-Based Mediation with Domain Maps", *17th International Conference on Data Engineering (ICDE '01)*, Washington-Brussels-Tokyo, April 2001.

[10] C. Baru, A. Gupta, B. Ludäscher, R. Marciano, Y. Papakonstantinou, P. Velikhov, and V. Chu, "XML-based Information Mediation with MIX", Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data: SIGMOD '99, Philadelphia, PA, June 1-3, 1999, SIGMOD Record, vol. 28(2), 1999, pp. 597-599.