A Framework for Ad-Hoc Federated Information Systems *

J. Tan
Monash University, Australia
jtan@csse.monash.edu.au

A. Zaslavsky
Monash University, Australia
azaslavs@csse.monash.edu.au

C. A. Ewald
Dept. of the Premier and Cabinet, Queensland, Australia
catherine.ewald@premiers.qld.gov.au

A. Bond
Distributed Systems Technology Centre
bond@dstc.edu.au

Abstract

Today's enterprise computing needs require the interoperability of heterogeneous information sources. This warrants complex architectures with weak central control involving structured, semi-structured or unstructured data. Functionality similar to that of database management systems are also required, e.g., support for queries. This paper presents our approach based on ad-hoc federations using a mediation system of active meta objects. Our framework adheres to established and emerging industrial standards, e.g. XML, CORBA, MOF. We also combine sound transaction processing and agent system concepts for coordinating and controlling component interaction. Our approach is feasible based on the results of a prototyping effort.

1. Introduction

Complex architectures that deal with issues of heterogeneity and interoperability are difficult to implement. It is often necessary to preserve the local autonomy of component information sources, so central control is minimal if not completely nonexistent. As a whole, it is difficult to provide global query access. There is motivation to reuse existing databases while making multi-source information sharing transparent to users. Such integrated solutions are generally possible, but the complexity involved can restrict the functionality of the implemented system.

One of the most researched fields in information sharing is the field of multidatabase systems or interoperable heterogeneous database systems ([1, 2, 3, 6, 8]):

1. Many attempt to combine the structures or *schemas* of databases through *schema integration* [5, 12]. However, this rigid approach can incur prohibitive implementation costs [9, 13], and generally involves considerable manual intervention.

2. Language-based facilities are more flexible [7, 8] but require users to deal with the integration logic.

3. Knowledge-based solutions using semantic support mechanisms, e.g., ontologies and concepts in linguistics, have met with reasonable success [1, 14, 15]. Such solutions recognize that the ability to handle data semantics is a primary but heavy requirement. However, many such solutions do not conform to existing standards, making it difficult to adopt them in existing enterprises.

4. Mediation and cooperation protocols involving wrappers and agents have also met reasonable success [5, 13]. Such systems promise scalability, extensibility and higher resilience against component system evolution. But as with the last point, many of these solutions do not reuse existing standards.

Our approach focuses on three things: ad-hoc federations, a wrapper-based metadata facility, and adherence to technology standards.

Some solutions attempt to combine ALL component data in one global system. With federations, the problem space shrinks. Each federation covers only a subset of the component systems, and are formed to deal with specific goals only, e.g., to respond to a query. By qualifying federations as "ad-hoc", we pertain to the responsiveness to an invocation where either accurate

*The work reported in this paper has been funded in part by the Cooperative Research Centre Programme through the Australian Government's Department of the Industry, Science and Resources.

results are generated, or the system will competently negotiate for a compromise with proportional trade-offs, e.g., response time. Ad-hoc behavior also means less manual intervention and some degree of quality of service. Ad-hoc federations are complex, transient systems involving temporal dependencies and possible uncertainty. We make things more interesting by not limiting ourselves to structured data from databases. We instead deal with *data sources*, making the framework open to semi-structured or unstructured data as well. We also use wrapper concepts in managing metadata to assist in overall system responsiveness. Finally, we adhere to technology standards for better integration with existing and future systems.

2. Our Tools

We advocate the use of technology standards to encourage consistency throughout an organization's software development history and to improve the interoperability of their software with legacy or future components. The Common Object Request Broker Architecture (CORBA) middleware standard[16] of the Object Management Group (OMG) provides interoperable communication in a distributed environment. We also build up our metadata support facility from the Extensible Markup Language (XML) of the World Wide Web Consortium (W3C) and the OMG Meta Object Facility (MOF) [11]. XML is a popular interchange standard for structured data, and we use it for metadata interchange. The MOF provides repository support for *meta-information*, i.e., information about information [4]. The abstract model of the MOF has four levels:

1. At level M0, the universe of information for some domain consists of *entities* and their *relationships* with one another.

2. At level M1, entities are classified into *types*, which may be involved in *type relations* with other types and may define *type schemas*. For instance, a conceptual model for a database may be defined in the form of an Entity Relationship (ER) schema.

3. The M2 level classifies types into *meta-types*, which may be involved in *meta-relations* and, using a *meta-type system*, define *meta-schemas*. This *meta-meta* level is where MOF defines *metamodels* such as the ER data model.

4. At the topmost level, M3, the MOF model defines how metamodels are defined.

Metadata management using the MOF begins with the metamodel from which a repository is generated to

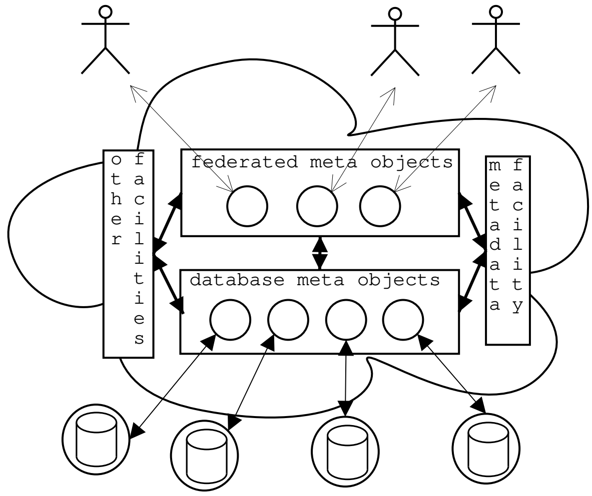

Figure 1. A Conceptual Framework.

store and manage metadata. Metadata packages are instantiated through factory objects to store specific metadata, i.e., for one data source. Metadata tools can then be written to manage the metadata. MOF meta objects can also be programmed with behavior. This is a key feature that we use, as will be explained later.

3. Ad-Hoc Federations

Figure 1 shows a conceptual framework for the development of ad-hoc federated systems. Our solution consists of the following key features:

1. We provide metadata management through *active meta objects* to manage metadata and to mediate federations. There are two types: *data source meta objects* and *federation meta objects*.

2. Data source meta objects are wrappers [5, 13] which assimilate metadata from the associated data source. Whenever possible, access patterns and transaction history are also observed. This usually requires that the data source meta object is able to work from within the data source site.

3. Federation meta objects embody federations in the form of global queries or views. These mediators will work with data source meta objects to realize the respective federations they were created for.

Figure 2 shows the architecture of an ad-hoc federation. Participation in such a federation is dynamic, since federation meta objects are created on demand

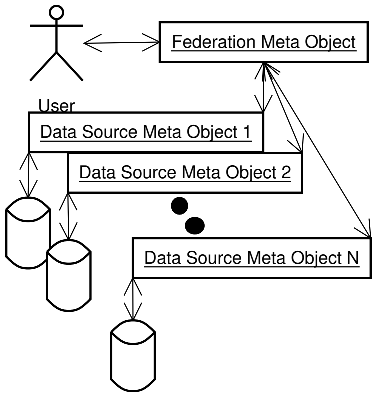

Figure 2. Snapshot of an ad-hoc federation.

and the involvement of data source meta objects will depend on the needs of the federation.

3.1. Data Source Meta Objects

Details on how to access the data source must initially be known in order to reach the data. Connection is possible through various means:

- As Listing 1 shows, queries can be invoked via method calls to a CORBA object servant.
- Some direct protocol, e.g., JDBC, can be used, as we do in our prototype. While this raises security issues, *export views* may be defined on top of live data, so that access is limited to the export views.
- Data can be invoked via server-side functions, e.g., Java *servlets*. This has the advantage of shielding the data source from direct access.

Data source meta objects will normally explore the data source associated with it to extract metadata. If this is not possible, then the data source administrator will have to “spoon-feed” the data source metadata in XML format to the data source meta object.

3.2. Federation Meta Objects

When a new federation meta object is instantiated, we can manually feed it a *federation schema*, as we do in the early stages of our prototype. On the other hand, it is better to allow an *imprecise federation schema* from which precise schemas can be inferred semi-automatically. This is generally better than providing precise query definitions which we may not assume to be known to begin with. There has been reasonable success in this area, e.g., *imprecise queries* [1]

```
<?xml version='1.0'?>
<!DOCTYPE rqset SYSTEM "relquery.dtd">
<rqset name="rqset1" ConnectProtocol="CORBA"
  URL="IOR:00000000000000003149444C3A6D6F746F7
  0000000005861746963504F410020202020000076A76"
  UserName="" PassWord="">
<results>
<resultset name="people()" definition=
 "(SELECT uid,uname,dept,'CSSE' as org
   FROM CSSEDB.user ) UNION
  (SELECT sid AS uid, sname AS uname,
   faculty AS dept, 'DLSU' AS org
   FROM DLSUDB.staff)">
<columns>
<column data_type="short" unique="T"
 not_null="F">uid</column>
<column data_type="short" unique="F"
 not_null="F">uname</column>
<column data_type="string" unique="F"
 not_null="F">dept</column>
<column data_type="string" unique="F"
 not_null="F">org</column>
</columns>
</resultset>
</results>
</rqset>
```

Listing 1. Sample global query definition in XML.

and *query discovery* [10]. Federation meta objects must be able to learn from the data source meta objects, analyze the imprecise federation definitions and infer the corresponding precise definitions. Successful inferences will generate the accurate federation metadata. Listing 1 shows a query schema over two databases, where the query is invoked as a method, `people()`.

How are conflicts and dissimilarities handled? Solutions to such issues in the literature can be adopted. Many are automated to some degree using ontologies and other aids that resolve semantic issues [1, 10, 14]. Many such solutions have already been tested by others and can be plugged into our flexible and portable framework. Inventing our own solutions is unnecessary and beyond the scope of our project.

Ad-hoc protocols govern how federation meta objects behave responsively. This is our basic protocol:

- The ideal response would be the most accurate result set possible.
- If such results are not possible, the federation meta object will inform the client and either provide a less accurate result at once, indicating the accuracy rating, or negotiate for additional time and guarantee a proportional increase in accuracy.

- In the worst case, sufficient time for human intervention will be negotiated for.

4. Conclusion

In this paper, we presented a solution to the problem of information sharing across heterogeneous information sources through ad-hoc federations. We illustrated how this is made possible using MOF-based metadata and active meta objects. We have already prototyped a basic metadata facility using a simplified relational metamodel. We have also successfully experimented with metadata extraction using JDBC.

The next step is to prototype active meta objects with basic learning abilities. We doubt that inventing new learning methods would be necessary as we can try out existing algorithms that work reasonably well in solving similar problems. The same is true for the protocols for ad-hoc federations, which are actually less complex. However, we have to formulate the measurement of result accuracy and extra time requirements for compromise scenarios. We must also move towards XML schemas which are better-suited for complex metadata than standard XML.

References

[1] M. W. Bright, A. R. Hurson, and S. Pakzad. Automated resolution of semantic heterogeneity in multidatabases. *ACM Trans. on Database Systems*, 19(2):212–253, June 1994.

[2] O. A. Bukhres and A. K. Elmagarmid, editors. *Object-oriented multidatabase systems: a solution for advanced applications*. Prentice-Hall, 1996.

[3] S. Ceri, B. Pernici, and G. Wiederhold. Distributed database design methodologies. In *Proc. IEEE*, pages 533–546, May 1987.

[4] S. Crawley, S. Davis, J. Indulska, S. McBride, and K. Raymond. Meta-meta is better-better! In *IFIP WG 6.1 International Working Conference on Distributed Applications and interoperable Systems (DAIS'97)*, 1997.

[5] L. M. Haas, R. J. Miller, B. Niswonger, M. T. Roth, P. M. Schwarz, and E. L. Wimmers. Transforming heterogeneous data with database middleware: Beyond integration. *IEEE Data Engineering Bulletin*, 22(1):31–36, 1999.

[6] A. R. Hurson and M. W. Bright. Object-oriented multidatabase systems. In O. A. Bukhres and A. K. Elmagarmid, editors, *Object-oriented multidatabase systems: a solution for advanced applications*, chapter 1, pages 1–36. Prentice-Hall, 1996.

[7] R. Krishnamurthy, W. Litwin, and W. Kent. Interoperability of heterogeneous databases with schematic discrepancies. In *Proc. First Int'l Workshop on Interoperability in Multidatabase Systems*, pages 144–151, 1991.

[8] W. Litwin. From databases to multidatabases: Why and how. In *Proc. Sixth British Nat'l Conf. on Databases*, pages 161–188. Cambridge Univ. Press, 1988.

[9] W. Litwin and A. Abdellatif. Multidatabase interoperability. *IEEE Computer*, 19(2):10–18, December 1986.

[10] R. Miller, L. M. Haas, and M. A. Hernández. Schema mapping as query discovery. In *Proceedings of the 26th VLDB Conference*, Cairo, Egypt, 2000.

[11] Meta object facility (MOF) specification. Object Management Group (OMG) Document ad97-08-14, 1997. URL: http://www.omg.org/cgi-bin/doc?ad/99-06-05.

[12] S. Navathe and A. Savasere. A schema integration facility using object-oriented data model. In O. A. Bukhres and A. K. Elmagarmid, editors, *Object-oriented multidatabase systems: a solution for advanced applications*, chapter 1, pages 105–128. Prentice-Hall, 1996.

[13] S. B. Navathe and M. J. Donahoo. Towards intelligent integration of heterogeneous information sources. In *In Proceedings of the 6th International Workshop on Database Re-engineering and Interoperability*, 1995.

[14] A. Ouksel and I. Ahmed. Coordinating knowledge elicitation to support context construction in cooperative information systems. In *Proc. First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, June 1996.

[15] A. Sheth and V. Kashyap. So far (schematically) yet so near (semantically). In D. K. Hsiao, E. J. Neuhold, and R. Sacks-Davis, editors, *IFIP DS-5 Semantics of Interoperable Database Systems*, volume 1, pages 282–301, November 1992.

[16] A. Vogel and K. Duddy. *Java programming with CORBA, Second Edition*. John Wiley & Sons, Inc., 1998.