The OASIS Multidatabase Prototype

Mark Roantree John Murphy
School of Computer Applications
Dublin City University
Ireland.
mark.roantree, john.murphy@compapp.dcu.ie

Wilhelm Hasselbring
Infolab
Tilburg University
Netherlands
hasselbring@kub.nl

Abstract

The OASIS Prototype is under development at Dublin City University in Ireland. We describe a multidatabase architecture which uses the ODMG model as a canonical model and describe an extension for construction of virtual schemas within the multidatabase system. The OMG model is used to provide a standard distribution layer for data from local databases. This takes the form of CORBA objects representing export schemas from separate data sources.

1 Introduction

The OASIS Project (ODMG Architectures for Specification of Interoperable Systems) is focused on the construction of a multidatabase prototype for usage in a healthcare environment. Research is conducted at Dublin City University with assistance provided by Iona Technologies with their Common Object Request Broker Architecture (CORBA) products, Versant through the usage of their object-oriented database, and St. James's Hospital through the use of their extensive healthcare applications as a suitable test environment for the planned prototype.

This document describes the architecture and the research which is on-going and planned for the project. Is is assumed that the reader is familiar with multidatabase (or federated database) systems. A suitable introduction to these types of systems is available in [SL90, BE96]. The report is structured as follows: in §2 the OASIS architecture is described in general terms; in §3 a more detailed description of the operations at each local database server is provided; in §4 the multidatabase kernel is described; in §5 we describe our efforts on case tool support for specifying multidatabase systems; and finally in §6 we describe the current status of the project and future plans for the prototype.

2 The OASIS Architecture

In this section the architecture is described from the bottom up. It is based on the widely accepted standard: a five-level schema architecture [SL90] where local schemas represent participating software systems containing the physical data, and above these in the architecture a provision exists for the construction of component, export, and federated schemas. When describing participating software systems they will be referred to as local databases even though some of these systems are no more than a flat-file of records. This is because the majority of the participants are database systems. In the current version of the prototype there are three 'databases' participating in the multidatabase system. The first is an object-oriented databases, the second is a flat-file of records which has been exported (as a snapshot) from a legacy system, and the third is a relational database. Participating databases are translated to a canonical model (CDM) representation before they can be merged. In this project the Object Data Management Group (ODMG) model [CB97] is used for this task.

In *figure 1* the overall architecture is illustrated briefly. Each local database is presented to the multidatabase kernel in the form of a view (export schema) of its canonical (ODMG) representation. This is achieved through the use of a mediator which manages data translation between the local data model and ODMG representation, and subsequently converts ODMG queries (global queries) to local database format, and the result set from the local data model format back to an ODMG representation. A separate mediator is used for each heterogenous data model. After an integration process, the mediator will hold information on mappings between the component and local schemas. Persistence for mediators is provided through the Local Mediation Service using an ODMG-compliant database, which is the Versant OODB [Ver97] for the OASIS project.

The CDM must have a facility to define views (export schemas) which are stored in the schema repository of the component schema. Component schemas may be virtual (containing only mappings to local data) or snapshots of local databases depending on integration requirements. Export schemas are loaded as CORBA objects and the multidatabase kernel uses them to construct federated schemas which are stored in the multidatabase schema repository.

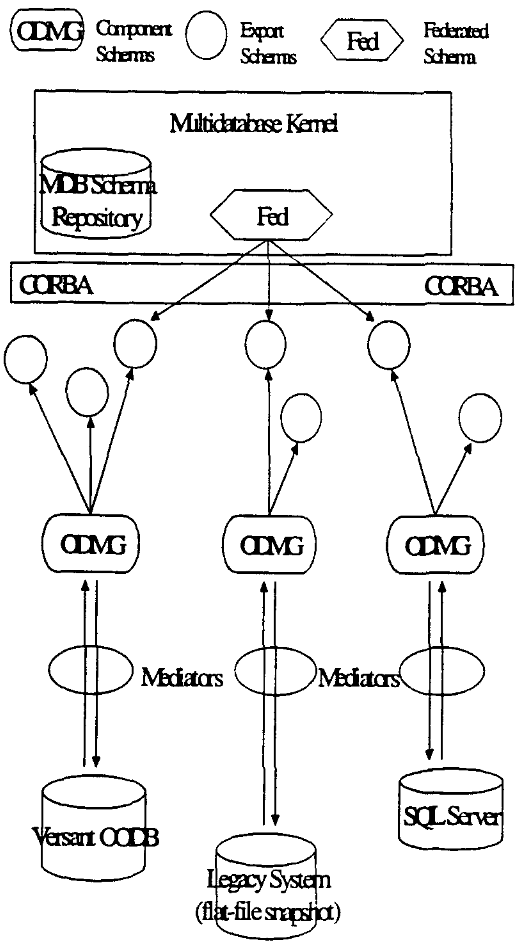

Figure 1: *The OASIS Architecture demonstrates how federated schemas are constructed. Local systems provide exports (schemas) of data which are merged to form the federated schema. The CORBA layer provides connectivity between the local systems and the multidatabase kernel.*

2.1 Software for Database Servers

It is necessary for some multidatabase software modules to reside at the local database server. A complete description of all OASIS components is provided in [Roa98a]. Both the component schema and its export schemas will reside at the local server. Export schemas are the contact point for the multidatabase kernel at each database server. In *figure 2* there are four mediator objects (termed facilitators in [Gra96]), as there are four databases resident on this server. Each database has a number of local applications running, and when executed, the mediator is treated as another local application.

A *Mediation Service* contains a generic mediator, a mechanism for the registration of component schemas, and a collection of mediators which are specific to each database residing at the local database server. In OASIS, the component schema is an ODL specification with no data objects.[1] In the illustration in *figure 2*, the mediation service invokes the appropriate mediator for the SQL Server database which interfaces with the database in the same fashion as the local applications.

2.2 Database Integration

The integration process is controlled by the Local Mediation Service and is briefly described here, with a fuller description in [Roa98b]. The first step in the integration process is the construction of a mediator object which will later be used to mediate between the local database and the ODMG representation (the canonical model representation). The Translation Knowledge Base (see *figure 2*) contains a set of basic rules for translating a schema to an ODMG schema. Refer to [BLN86, SL90, BE96] for details of schema-schema transformations, and more particularly to [FV95] for relational to ODMG schema conversions. At present, the relational rules are specified, and it is planned to include a set of hierarchical

[1] It *can* be populated with a snapshot of the local databases if required. This is done in cases where snapshot data is acceptable for transactions and an improvement in performance is required.

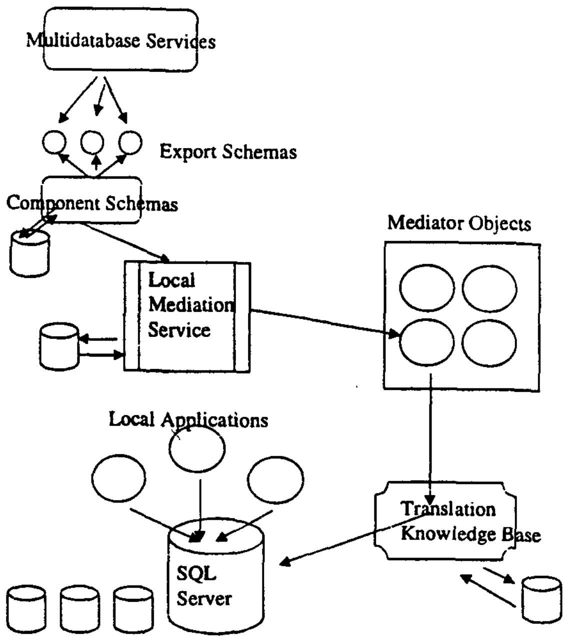

Figure 2: *Software Architecture at Each Database Server*

rules to convert a hierarchical model to an ODMG model, and a set of object-oriented rules to convert non-ODMG schemas to an ODMG version. The relational rules assume an ODBC compliance so that they can operate with any type of relational database. The ruleset is used to provide a first pass of the newly constructed ODMG representation of the local data model. This is augmented by *decisions* made by the user to customise the first pass schema, and create the final translation of the schema. These *decisions* are stored in the Translation Knowledge Base, and are accessible by the mediator object. Thus, the Translation Knowledge Base consists of a set of *rules* and *decisions* for each data model translated.

The final step is implementation specific: it can be done in other ways but we choose to take this approach as it meets the needs of the target healthcare environment. A series of mapping objects are created which map to specific data entities in the local database. Although the component schema contains no data objects (other than export schema definitions), it can indirectly access individual data entities in the local schema through the mappings constructed by the Local Mediation Service, and used by the mediator object. These mappings, described as *omaps* and *amaps* are described in §4 where the role of the mediator is described in more detail. The access to local database objects is indirect as it must be made via the mediator object: only the mediator understands the behaviour required to pass a query from the component schema to the local schema; return query results from the local schema to the component schema; and inform the component schema of changes to the local schema.

At this point we have a component schema (in the form of an ODL definition) which is used as an interface to the multidatabase system (through its export schemas), and which can be used to query the local database through the mediator. It only remains to define export schemas which can subsequently be used by the multidatabase kernel to construct federated schemas. Our work also involves the implementation of a view mechanism for the ODMG data model [RKB98] to facilitate this task. Export schemas are accessible through CORBA objects to provide a distribution layer. The CORBAexport object (see §3) defined in OASIS is generic, providing only a description of the export schema (metadata) to the MDB kernel.

3 The Multidatabase Kernel

The multidatabase kernel can reside at one of the participant servers or on a separate server. A fundamental condition of the architecture is that the MDB kernel does not change to suit a new database which is added to the system. This is achieved through the integration process which generates export schemas to provide the interface to the multidatabase kernel. Export schemas are made available through a generic CORBA object which is called the CORBAexport object. As this object is the same for all export schemas the multidatabase kernel deals only with a uniform object. This object is not described here but resides in the CORBA Layer shown in *figure 1.*

3.1 Constructing Federated Schemas

Export schemas are available at each database server as illustrated in *figure 2.* Federated schemas are defined from export schemas using the software of the Multidatabase Schema Repository. Binary operators were defined to restructure schemas so that two schemas resemble each other before they are merged. The sequence of operations and the newly defined federated schema are stored in the MDB schema repository.

MOQL, a multidatabase version of the ODMG's Object Query Language (OQL) is used to define export schemas. The view mechanism and a series of integration operators for MOQL are described in [RKB98]. The Query Service must be able to undo each layer of restructuring or merging operations to map the federated schema back to participant export schemas.

3.2 The Global Mediation Service and Queries

For the next version of the OASIS prototype, we still intend to provide only a read-only query service which provides a snapshot of queries at the multidatabase server (stored in the MDB schema repository). If suitable client software resides at local machines, this data can be downloaded. This cut-down' query service uses the concept of a generative communication service to pass data objects to the MDB server. The MDB server must transfer objects of unknown type, and the Query Service must combine these *foreign* objects to produce the snapshot result. Details of how this operates is described fully in [HR98] and is enhanced in OASIS as it uses the Global Mediation Service to manage cooperation between distributed data sources.

A Global Mediation Service (GMS) exists at the multidatabase kernel with similar properties and responsibilities to the LMS described earlier. The MDB kernel must manage distributed objects (export schemas) and *force* them to cooperate to construct a federated schema. Since export schemas know nothing of each other, and it would be bad engineering practice to force links between them, a mediator is used to manage the cooperating behaviour of these objects. Thus, the Multidatabase Query Service makes use of the GMS to manage communication between data sources when constructing query results.

4 Data Model Interoperability

The integration process described in §2 provides details of how the component schema interoperates with a local schema. In this section we will discuss the need for each of the components involved in the process.

4.1 The Role of Mediator Objects

In [GHJV95] they describe mediators as responsible for controlling the interactions between objects. Where it is necessary for objects to overlap, it is important to ensure that the objects do not have to know about the internal operations or structures of each other (i.e. so as not to break encapsulation). The role of the mediator helps in this process. The two cooperating objects are the component and local schema objects. Their separate roles are clear. The local schema runs local applications.[2] The component database schema is used by the multidatabase to provide details of information available and to accept multidatabase queries. The two schemas must cooperate when:

- queries are passed from the component to the local schema;
- results are passed from the local to the component schema;
- the component schema is informed of changes at the local database level.

To avoid forcing both separate objects to understand each other's internal operations, a mediator object is used to model the cooperating behaviour of the objects. The mediator can manage these operations if it has knowledge of how the component schema maps to the local schema.

When a component schema is constructed, it is used merely as an interface to the multidatabase system and contains no physical objects. It is therefore necessary to construct some form of mapping between the component and local schemas. The mediator object is used to construct and manage these mappings. There are two types of structural mappings: an *omap* maps between an object in the component schema and some entity, object, table, or structure in a local schema, and an *amap* maps a component schema attribute to a local schema attribute. A mediator object is composed of identifiers for the component and local schemas, and a set of *omaps* and *maps*. It must also make use of a Translation Knowledge Base which contains rules for data model translation, and query transformation knowledge (decisions) between data models. The function of this knowledge base is explained in the following section. A worked example of the integration process for a relational schema and the construction of *omaps* and *amaps* is provided in [Roa98b].

4.2 The Translation Knowledge Base

The Translation Knowledge Base (TKB) is used to help translate different data models to the ODMG data model. Predefined rules are built into the TKB for different conversion types and these can be augmented with user decisions in the form of predefined operators for object-oriented schema manipulations.

[2] The mediator is also treated as a local application when it needs to communicate with the local database.

For example, an operator exists which can create a generalisation of two classes, a process which could be regarded as enriching the original schema. Decisions are recorded and stored by the TKB. Mediators subsequently interpret this information when providing communication between component and local schemas. Although a more detailed description of the TKB is provided In [Roa98a], this component is in the early stages of design, and is not used in the current version of the prototype. At present a cut-down version of the TKB is used for ODBC (relational model) to ODMG data model translation.

4.3 XML-Based Data Exchange

The Extensible Markup Language (XML) is subset of SGML that is designed to make it easy to interchange structured documents over the Internet [Hol98]. The more familiar HTML, is an *application* of SGML. XML files always clearly mark where the start and end of each of the logical parts (called elements) of an interchanged document occurs. XML is most often used as a data-description language [Hol98]. XML is not limited to textual applications, it is used in EDI (Electronic Data Interchange) and other forms of structured electronic data exchange.

The main role of XML (as opposed to HTML) in interoperable systems is likely to be for defining the structure of data to be exchanged between heterogeneous information systems. The flexibility of XML data type definitions allow the structure of any document or record to be described. XML can provide one way of handling various categories of information such as archived data from legacy systems as well as (semi-) structured information in various formats from other sources (e.g., reports from diagnostic service departments).

Strictly speaking, XML is a *meta-language* for formally describing a markup language. *Markup* is a term that covers any means of making explicit an interpretation of a text. A markup language specifies what markup is allowed, what is required and how markup is to be distinguished from text. This is achieved in XML by specifying a Document Type Definition (DTD), which may be thought of as a model or template of the document. Instances of XML documents can only be understood in relationship to their DTD. When we talk about XML documents we need to refer to explicit DTDs.

Within the OASIS project, the use of XML as a data-exchange format is considered, whereby the Translation Knowledge Base is extended with ODMG-XML translation rules.

The XML standard is not concerned with the semantics of textual elements. The developers of XML tags have to choose intelligible names for the elements they identify and document their proper use. The DTD sets out the structure using a simple syntax to specify the order and contents of each element. Various features can be combined to allow complex document structures to be defined.

To achieve semantic interoperability, the data exchange will be based on the Health Level 7 protocol, which has been designed to standardize the data transfer within hospitals [Ham93]. HL7 is an application level protocol and so relates to level seven of the ISO/OSI-protocol hierarchy. HL7 covers various aspects of data exchange in HISs, e.g., admission, discharge, and transfer of patients, as well as the exchange of analysis and treatment data. The HL7 standard represents hospital related transactions as standardized messages as HL7 is a de-facto standard for data exchange between commercial systems for hospitals [Don95]. The research on integrating ODMG with XML is a collaboration with researchers at Tilburg University and is expected to be included in the next version of the prototype.

5 Case Tool Support

As part of our ongoing research we investigate advanced modelling techniques which will help us to improve the specification and implementation of interoperable systems.

While it is true that many interoperable systems components evolve in a bottom-up manner, it is very useful to have an abstract model of the IS and its components. Many new OO methods are available to the system modeller and there is a wide choice of models, notations and methods which can be used. Our preliminary research has shown that few if any OO methods or notations scale to the problems of interoperable and federated systems. To this end we have adopted a hybrid approach and constructed our own formalism which combines informal OO notations with more formal notations in a synthesis which allows description of interoperable systems models.

We did not feel that any of the first, second or third generation models such as OMT, Fusion, Syntropy or even the UML had sufficient modelling power for our application requirements. We have extended conventional OO modelling notations with constructs and operations for modelling interoperable systems. An additional augmentation of our method (called Minerva [MG98]) is the adoption of a formal method which allows us to specify the semantics of our meta-model seamlessly and without recourse to a definition in terms of Minerva itself (as is the case with UML semantics). We are actively testing and evolving our method and are constructing tools which will support experimentation and further evolution of Minerva.

It is clear from our work and from the work of many researchers in the field that traditional OO methods do not scale well with regard to interoperable systems. It is beyond the scope of this article to detail the components and constructs of Minerva and we intend to elucidate on it in detail in a forthcoming paper. Conceptual modelling of information systems is a critical task and interoperable systems are no exception. ISs provide a very challenging environment for the model and method developer.

6 Conclusions and Future Research

In this report the OASIS architecture and several key components have been described. Although the architecture is specified in detail, most of the components require varying amounts of research and thus, only some components have been constructed. An initial requirement was to remodel some of the applications at St James' Hospital using separate database systems to provide a basic environment in which to implement and test ideas for system components. This step has been completed in addition to the construction of some test components by researchers on the OASIS project. However, OASIS version 1 has been constructed using Microsoft Visual C++ (version 5.0), Orbix [Orb97] for NT (version 2.3c), and Versant Object Oriented Database (version 5.0.8). It contains three participating databases as described with real data (without confidential information such as names and address) supplied by the large Hospital Information System at St. James' Hospital in Dublin. Completed components include:

- ODBC to ODMG Model Translator. This involves the rule set for the TKB and the implementation of operators and the persistent storage of operations.
- ODMG extensions to construct dynamic collections. These dynamic collections are exported and joined at the multidatabase kernel [SR98]. It also includes a daemon to constantly update collections as changes occur in the local database.
- Federated Schema Builder. It uses CORBA objects as export schemas and provides a series of operators to build the federated schema.
- Construction of an ODMG-compatable Schema Repository for Versant databases. This is an unfortunate necessity as no ODMG vendor has yet to provide the ODMG 2.0 access interface for the Schema Repository. Since we use Versant for all component schemas it was necessary to be able to query the schema repository as indicated in the ODMG 2.0 specification [CB97].

Research is at an advanced stage and/or construction of new prototype components has started in the following areas:

- The OASIS View Mechanism
  - Defintion of a view mechanism for ODMG. This is used to create export schemas.
  - Definition of integration operators. They are used to restructure export schemas to build the federated schema.
  - Specification of an Multidatabase OQL Interpreter. We are building our own extended ODMG interpreter using JAVA to convert global MOQL queries to standard OQL queries for each component schema.
- A Communication Framework for interoperating with separate databases. This work concentrated on the problem of a global service which must manage objects of an unknown type [HR98].
- Development of a sub-language for querying an ODMG Schema Repository.

Research is either on-going or due to start shortly on the following OASIS components:

- Construction of the Translation Knowledge Base. This is a two-year assignment and is intended to provide a generic transaltion mechanism for selected data models to an ODMG format.
- Specification of a Local Mediation Service.
- The OASIS Multidatabase Query Service.
- Using the CORBAexport object to describe MOQL export schemas (metadata definition).
- Delevopment of the Minerva CASE tool.

  It is intended to complete all components of the OASIS mlutidatabase prototype within 24 months whereupon it will be tested using real applications at St. James' Hospital in Dublin. The OASIS website [Oas98] contains status reports on the current version of the prototype and a list of internal technical reports.

References

[BE96] Bukhres O. and Elmagarmid A. (eds). *Object-Oriented Multidatabase Systems.* Prentice Hall, 1996.

[BLN86] Batini C., Lenzerini M., and Navathe S. A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys*, 3:(4), 1986.

[CB97] Catell R. and Barry D.(eds). *The Object Database Standard: ODMG 2.0.* Morgan Kaufmann, 1997.

[CHHS97] Conrad S., Hasselbring W., Heuer A., Saake G. Engineering Federated Database Systems. *Proceedings of the CAISE 97 Workshop*, Preprint 6/1997, University of Magdeburg, 1997.

[Don95] McDonald C.J. News on U.S. Health Informatics Standards. *M.D. Computing* 12:3, 1995.

[FV95] Fahrner C. and Vossen G. Transformaing Relational Database Schemas into Object Oriented Schemas According to ODMG-93. *4th International Conference on Deductive and Object-Oriented Databases*, LNCS 1013, 1995.

[GHJV95] Gamma E., Helm R., Johnson R. and Vlissedes J. *Design Patterns - Elements of Reusable Software.* Addison-Wesley, 1995.

[Gra96] Gray P.M.D. *Large Databases and Knowledge Re-use.* In *Computing Tomorrow: Future Research Directions in computer Science*, (Ian Wand & Robin Milner eds), Cambridge University Press, 1996.

[Ham93] Hammond W.E. *Health Level 7: A Protocol for the Interchange of Healthcare Data.* In *Progress in Standardization in Health Care Informatics* (DeMoor/McDonald/vanGoor eds), IOS Press, Amsterdam, 1993.

[Hol98] Holzner S. *XML Complete.* McGraw-Hill, 1998.

[HR98] Hasselbring W. and Roantree M. A Generative Communication Service for Database Interoperability. Proceedings of *3rd IFCIS Conference on Cooperative Information Systems*, IEEE Computer Society Press, 1998.

[Jor98] Jordan D. C++ *Object Databases: Programming with the ODMG Standard.* Addison Wesley, 1998.

[MG98] Murphy J. and Grimson J. Cooperative Information Systems: Interoperability in Health Care Legacy Applications, *International Journal of Cooperative Information Systems*, 7:(1), World Scientific Publishing, 1998.

[Oas98] The OASIS Project, Dublin City University, www.compapp.dcu.ie/~mark/TEXT /OASIS/overview.html, 1998.

[OHE96] Orfali R., Harkey D., and Edwards J. *The Essential CORBA: System Integration Using Distributed Objects*, Wiley, 1996.

[Orb97] Orbix 2.3c Programmer's Guide, IONA Technologies PLC, 1997.

[PBE95] E. Pitoura, O. Bukhres and A. Elmagarmid. Object Orientation in Multidatabase Systems. *ACM Computing Surveys* 27:2, 1995.

[RKB98] Roantree M., Kennedy J. and Barclay P. A Multidatabase Layer for the ODMG Object Model, Proceedings of *5th International Conference on Object-Oriented Information Systems*, Springer, 1998.

[Roa98a] Roantree M. The OASIS Multidatabase Architecture. *Oasis Report No. OAS-01*, Dublin City University, May 1998.

[Roa98b] Roantree M. Database Integration in the OASIS Multidatabase System. *Oasis Report No. OAS-02,* Dublin City University, May 1998.

[SR98] Shehill A, and Roantree M. Constructed Multidatabase Collections Using an Extended ODMG Object Model. *Submitted for publication.*

[SL90] A. Sheth and J. Larson. Federated Database Systems for Managing Distributed, Heterogenous and Autonomous Databases. *ACM Computing Surveys*, 22:3, 1990.

[Ver97] *Versant System Manual.* Versant Object Technology Corporation, 1997.