Chapter 1

Overview

1.1 Background

This document describes the continuing work on standards for object database management systems (ODBMSs) undertaken by the members of the Object Database Management Group (ODMG). This specification represents an enhancement to ODMG-93, Release 1.2.

We have worked outside of traditional standards bodies for our efforts in order to make quick progress. Standards groups are well suited to incremental changes to a proposal once a good starting point has been established, but it is difficult to perform substantial creative work in such organizations due to their lack of continuity, large membership, and infrequent meetings. It should be noted that relational database standards started with a database model and language implemented by the largest company involved (IBM); for our work, we have picked and combined the best features of implementations we had available to us.

1.1.1 Importance of a Standard

Before ODMG, the lack of a standard for object databases was a major limitation to their more widespread use. The success of relational database systems did not result simply from a higher level of data independence and a simpler data model than previous systems. Much of their success came from the standardization that they offer. The acceptance of the SQL standard allows a high degree of portability and interoperability between systems, simplifies learning new relational DBMSs, and represents a wide endorsement of the relational approach.

All of these factors are important for object DBMSs, as well. The scope of object DBMSs is more far-reaching than that of relational DBMSs, integrating the programming language and database system, and encompassing all of an application's operations and data. A standard is critical to making such applications practical.

The intense ODMG effort has given the object database industry a "jump start" toward standards that would otherwise have taken many years. ODMG enables many vendors to support and endorse a common object database interface to which customers write their applications.

1.1.2 Goals

Our primary goal is to put forward a set of standards allowing an ODBMS customer to write portable applications, i.e., applications that could run on more than one ODBMS product. The data schema, programming language binding, and data manipulation and query languages must be portable. Eventually, we hope our standards proposal will be helpful in allowing interoperability between the ODBMS products as well, e.g., for heterogeneous distributed databases communicating through the OMG Object Request Broker.

We are striving to bring programming languages and database systems to a new level of integration, moving the industry forward as a whole through the practical impetus of real products that conform to a more comprehensive standard than is possible with relational systems. We have gone further than the least common denominator of the first relational standards, and we want to provide portability for the entire application, not just the small portion of the semantics encoded in embedded SQL statements.

The ODMG member companies, representing almost the entire ODBMS industry, are supporting this standard. Thus, our proposal has become a de facto standard for this industry. We have also used our specification in our work with standards groups such as the OMG and the ANSI X3H2 (SQL) committee.

We do not wish to produce identical ODBMS products. Our goal is source code portability; there is a lot of room for future innovation in a number of areas. There will be differences between products in performance, languages supported, functionality unique to particular market segments (e.g., version and configuration management), accompanying programming environments, application construction tools, small versus large scale, multithreading, networking, platform availability, depth of functionality, suites of predefined type libraries, GUI builders, design tools, and so on.

Wherever possible, we have used existing work as the basis for our proposals, from standards groups and from the literature. But, primarily, our work is derived by combining the strongest features of the ODBMS products currently available. These products offer demonstrated implementations of our standards components that have been tried in the field.

1.1.3 Definition

It is important to define the scope of our efforts, since ODBMSs provide an architecture that is significantly different than other DBMSs — they are a revolutionary rather than an evolutionary development. Rather than providing only a high-level language such as SQL for data manipulation, an ODBMS transparently integrates database capability with the application programming language. This transparency makes it unnecessary to learn a separate DML, obviates the need to explicitly copy and translate data between database and programming language representations, and supports substantial

performance advantages through data caching in applications. The ODBMS includes the query language capability of relational systems as well, and the query language model is more powerful; e.g., it incorporates lists, arrays, and results of any type. Figure 1-1 provides a comparison of DBMS architectures.

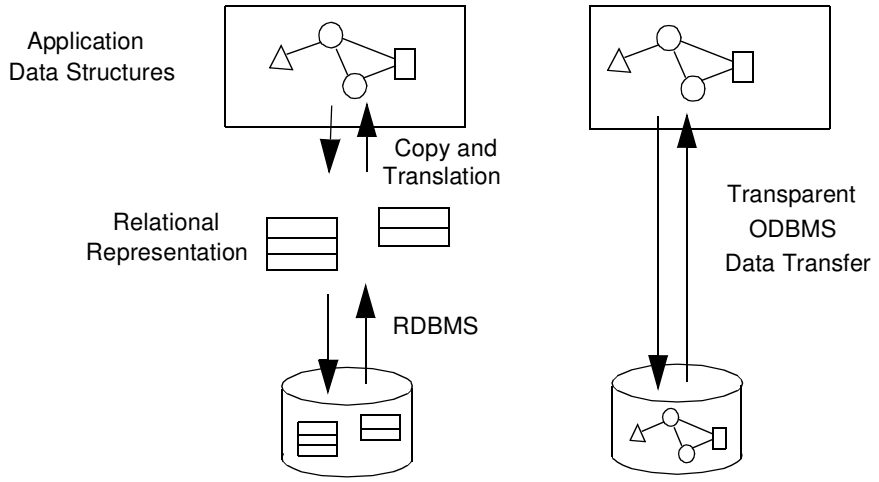

Figure 1-1. Comparison of DBMS Architectures

In summary, we define an *ODBMS* to be a DBMS that integrates database capabilities with object-oriented programming language capabilities. An ODBMS makes database objects appear as programming language objects, in one or more existing programming languages. The ODBMS extends the language with transparently persistent data, concurrency control, data recovery, associative queries, and other database capabilities. For more extensive definition and discussion of ODBMSs, the reader is referred to textbooks in this area (e.g., Cattell, *Object Data Management*).

1.2 Architecture

In order to understand the chapters of this book, it is necessary to understand the overall architecture of ODBMSs.

1.2.1 Major Components

The major components of ODMG 2.0 are described in subsequent chapters:

> *Object Model.* The common data model to be supported by ODBMSs is described in Chapter 2. We have used the OMG Object Model as the basis for

our model. The OMG core model was designed to be a common denominator for object request brokers, object database systems, object programming languages, and other applications. In keeping with the OMG Architecture, we have designed an ODBMS *profile* for their model, adding components (e.g., relationships) to the OMG core object model to support our needs. Release 2.0 introduces a meta model in this chapter.

Object Specification Languages. The specification languages for ODBMSs are described in Chapter 3. One is the object definition language, or ODL, to distinguish it from traditional database data definition languages, or DDLs. We use the OMG interface definition language (IDL) as the basis for ODL syntax. Release 2.0 adds another language, the object interchange format, or OIF, which can be used to exchange objects between databases, provide database documentation, or drive database test suites.

Object Query Language. We define a declarative (nonprocedural) language for querying and updating database objects. This object query language, or OQL, is described in Chapter 4. We have used the relational standard SQL as the basis for OQL, where possible, though OQL supports more powerful capabilities.

C++ Language Binding. Chapter 5 presents the standard binding of ODBMSs to C++; it explains how to write portable C++ code that manipulates persistent objects. This is called the C++ OML, or object manipulation language. The C++ binding also includes a version of the ODL that uses C++ syntax, a mechanism to invoke OQL, and procedures for operations on databases and transactions.

Smalltalk Language Binding. Chapter 6 presents the standard binding of ODBMSs to Smalltalk; it defines the binding in terms of the mapping between ODL and Smalltalk, which is based on the OMG Smalltalk binding for IDL. The Smalltalk binding also includes a mechanism to invoke OQL and procedures for operations on databases and transactions.

Java Language Binding. Chapter 7 defines the binding between the ODMG Object Model (ODL and OML) and the Java programming language as defined by Version 1.1 of the Java™ Language Specification. The Java language binding also includes a mechanism to invoke OQL and procedures for operations on databases and transactions.

It is possible to read and write the same database from C++, Smalltalk, and Java, as long as the programmer stays within the common subset of supported data types. More

chapters may be added at a future date for other language bindings. Note that unlike SQL in relational systems, ODBMS data manipulation languages are tailored to specific application programming languages, in order to provide a single, integrated environment for programming and data manipulation. We don't believe exclusively in a universal DML syntax. We go further than relational systems, as we support a unified object model for sharing data across programming languages, as well as a common query language.

1.2.2 Additional Components

In addition to the object database standards, ODMG has produced some ancillary results aimed at forwarding the ODBMS industry. These are included as appendices:

OMG Object Model Profile. Appendix A describes the differences between our object model and the OMG Object Model, so that Chapter 2 can stand alone. As just mentioned, we have defined the components in an ODBMS profile for OMG's model. This appendix delineates these components.

OMG ORB Binding. Appendix B describes how ODBMS objects could participate as OMG objects, through an adaptor to an object request broker (ORB) that routes object invocations through object identifiers provided by an ODBMS. We also outline how ODBMSs can make use of the OMG ORB.

1.2.3 ODBMS Architecture Perspective

A better understanding of the architecture of an ODBMS will help put the components we have discussed into perspective.

Figure 1-2 illustrates the use of the typical ODBMS product that we are trying to standardize. The programmer writes declarations for the application schema (both data and interfaces) plus a source program for the application implementation. The source program is written in a programming language (PL) such as C++, using a class library that provides full database OML, including transactions and object query. The schema declarations may be written in an extension of the programming language syntax, labeled PL ODL in the figure, or in a programming language–independent ODL. The latter could be used as a higher-level design language, or to allow schema definition independent of programming language.

The declarations and source program are then compiled and linked with the ODBMS to produce the running application. The application accesses a new or existing database, whose types must conform to the declarations. Databases may be shared with other applications on a network; the ODBMS provides a shared service for transaction and lock management, allowing data to be cached in the application.

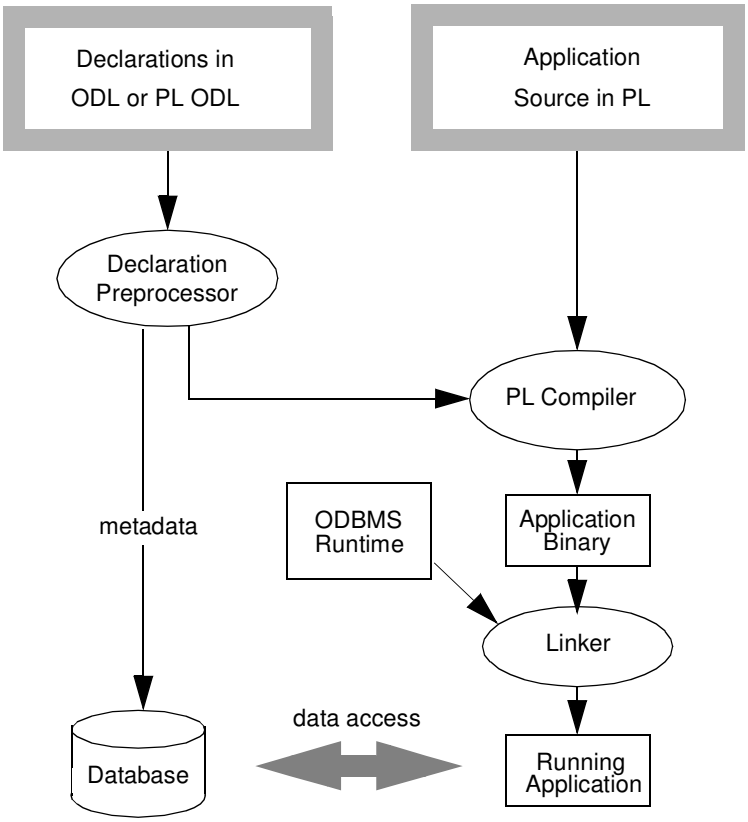

Figure 1-2. Using an ODBMS

1.3 Status

This document describes Release 2.0 of the ODMG standard. The ODMG voting member companies and many of the reviewer member companies are committed to support this standard in their products by the end of 1998.

1.3.1 Participants

As of March 1997, the participants in the ODMG are

- Rick Cattell (ODMG chair, Java workgroup chair, Release 1.0 editor), JavaSoft
- Jeff Eastman (ODMG vice-chair, object model workgroup chair, Smalltalk editor), Windward Solutions

- Douglas Barry (ODMG executive director, Release 1.1, 1.2, and 2.0 editor), Barry & Associates
- Mark Berler (object model and object specification languages editor), American Management Systems
- David Jordan (C++ editor), Lucent Technologies
- Francois Bancilhon, Sophie Gamerman (OQL editor), voting member, O_2 Technology
- Dirk Bartels (C++ workgroup chair), Olaf Schadow, voting member, POET Software
- William Kelly, voting member, UniSQL
- Paul Pazandak, voting member, IBEX
- Ken Sinclair, voting member, Object Design
- Adam Springer (Smalltalk workgroup chair), voting member, GemStone Systems
- Henry Strickland (Java editor), Craig Russell, voting member, Versant Object Technology
- Drew Wade (OQL workgroup chair), Jacob Butcher, Dann Treachler, voting member, Objectivity
- Michael Card, reviewer member, Lockheed Martin
- Edouard Duvillier, reviewer member, MATISSE Software
- Jean-Claude Franchitti, reviewer member, VMARK Software
- Marc Gille, reviewer member, MICRAM Object Technology
- William Herndon, reviewer member, MITRE
- Mamdouh Ibrahim, reviewer member, Electronic Data Systems
- Richard Jensen, reviewer member, Persistence Software
- Yutaka Kimura, reviewer member, NEC
- Shaun Marsh, reviewer member, Fujitsu Open Systems Solutions
- Richard Patterson, reviewer member, Microsoft
- Shirley Schneider, reviewer member, ONTOS
- Jamie Shiers, reviewer member, CERN
- John Shiner, reviewer member, Andersen Consulting
- Jacob Stein, reviewer member, Sybase
- Fernando Velez, reviewer member, Unidata
- Satoshi Wakayama, reviewer member, Hitachi

It is to the personal credit of all participants that the ODMG standard has been produced and revised expeditiously. All of the contributors put substantial time and personal investment into the meetings and this document. They showed remarkable

dedication to our goals; no one attempted to twist the process to his or her company's advantage. The reviewers were also very helpful, always ready to contribute.

In addition to the regular ODMG participants above, we received valuable feedback from others in academia and industry. We would like to thank our academic reviewers, in particular Eliot Moss for his contribution to the object model chapter. We would also like to thank Joshua Duhl for his exhaustive review of Release 1.2 as part of the process to create certification test suites.

1.3.2 History

Some of the history and methodology of ODMG may be helpful in understanding our work and the philosophy behind it. We learned a lot about how to make quick progress in standards in a new industry while avoiding "design by committee."

ODMG was conceived at the invitation of Rick Cattell in the summer of 1991, in an impromptu breakfast with ODBMS vendors frustrated at the lack of progress toward ODBMS standards. Our first meeting was at SunSoft in the fall of 1991.

The group adopted rules that have been instrumental to our quick progress. We wanted to remain small and focused in our work, yet be open to all parties who are interested in our work. The structure evolved over time. Presently, we have established workgroups, one for each chapter of the specification. Each workgroup is intended to remain small, allowing for good discussion. The specifications adopted in each workgroup, however, must go before the ODMG Board for final approval. The Board usually holds open meetings for representatives from all members to attend and comment on our work.

The people who come to our meetings from our member companies are called Technical Representatives. They are required to have a technical background in our industry. We also have established rules requiring the same Technical Representatives come repeatedly to our meetings to maintain continuity of our work. Technical Representatives from voting member companies often contribute 25 percent of their time to the ODMG. Some of the Technical Representatives from our reviewer members contribute an equal amount of time while others are more in the range of 10 percent of their time.

Voting membership is open to organizations that have developed and commercially sell a currently shipping ODBMS as defined on page 3. Reviewer members are individuals or organizations having a direct and material interest in the work of the ODMG. Certification members, our newest group of members, are individuals or organizations having a direct and material interest in the certification work of the ODMG.

1.3.3 Accomplishments

Since the publication of Release 1.0, a number of activities have occurred.

1. Incorporation of the ODMG and the establishment of an office.
2. Affiliation with the Object Management Group (OMG), OMG adoption (February 1994) of a Persistence Service endorsing ODMG-93 as a standard interface for storing persistent state, and OMG adoption (May 1995) of a Query Service endorsing the ODMG OQL for querying OMG objects.
3. Establishment of liaisons with ANSI X3H2 (SQL), X3J16 (C++), and X3J20 (Smalltalk), and ongoing work between ODMG and X3H2 for converging OQL and SQL3.
4. Addition of reviewer membership to allow the user community to participate more fully in the efforts of the ODMG.
5. Addition of certification membership to allow the user community to participate in the development of our certification test suites.
6. Publication of articles written by ODMG participants that explain the goals of the ODMG and how they will affect the industry.
7. Collection of feedback on Release 1.0, 1.1, and 1.2, of which much was used in this release.

1.3.4 Next Steps

We now plan to proceed with several actions in parallel to keep things moving quickly.

1. Distribute Release 2.0 through this book.
2. Complete implementation of the specifications in our respective products.
3. Collect feedback and corrections for the next release of our standards specification.
4. Continue to maintain and develop our work.
5. Continue to submit our work to OMG or ANSI groups, as appropriate.

1.3.5 Suggestion and Proposal Process

If you have suggestions for improvements in future versions of our document, we welcome your input. We recommend that change proposals be submitted as follows:

1. State the essence of your proposal.
2. Outline the motivation and any pros/cons for the change.
3. State exactly what edits should be made to the text, referring to page number, section number, and paragraph.
4. Send your proposal to proposal@odmg.org.

1.3.6 Contact Information

For more information about the ODMG and the latest status of its work, send electronic mail to info@odmg.org. You will receive an automated response.

If you have questions on ODMG 2.0, send them to question@odmg.org.

If you have additional questions, or if you want membership information for the ODMG, please contact ODMG's executive director, Douglas Barry, at dbarry@odmg.org, or contact:

Object Database Management Group
14041 Burnhaven Drive, Suite 105
Burnsville, MN 55337 USA
voice: +1-612-953-7250
fax: +1-612-397-7146
email: info@odmg.org
Web: www.odmg.org

1.3.7 Related Standards

There are references in this book to ANSI X3 documents, including SQL specifications (X3H2), Object Information Management (X3H7), the X3/SPARC/DBSSG OODB Task Group Report (contact fong@ecs.ncsl.nist.gov), and the C++ standard (X3J16). ANSI documents can be obtained from:

X3 Secretariat, CBEMA
1250 Eye Street, NW, Suite 200
Washington, DC 20005-3922 USA

There are also references to Object Management Group (OMG) specifications, from the Object Request Broker (ORB) Task Force (also called CORBA), the Object Model Task Force (OMTF), and the Object Services Task Force (OSTF). OMG can be contacted at:

Object Management Group
Framingham Corporate Center
492 Old Connecticut Path
Framingham, MA 01701 USA
voice: +1-508-820-4300
fax: +1-508-820-4303
email: omg@omg.org
Web: www.omg.org