Implementation of an Interface to Multiple Databases

Elizabeth R. Towell
Northern Illinois University

William D. Haseman
University of Wisconsin-Milwaukee

This paper presents the design of an interface to multiple databases and a test implementation. The interface offers a reusable shell for designers who wish to create a single platform of access to multiple databases which have significantly different data types. The interface is built in a hypermedia environment and facilitates browsing and extraction of information stored in multiple databases. A generic, application-independent set of manipulation capabilities is provided. The test implementation integrates information from video, map, and record-oriented databases.

Recent technical advances, including broadband optical networks and the availability of massive high speed storage devices, has spawned a movement towards computer storage of more complex data types (Ghafoor, 1988). Application specific research includes multimedia document systems (Thomas, 1985; Naffah, 1986; Morris, 1990) and cartographic systems (Dataquest, 1988; IBM, 1992). Many multimedia DBMS’s and front-end products are now on the market (Barry, 1992).

Another area of current effort is the development of multi-database systems (Kambayashi, Rusinkiewicz, and Sheth, 1991). Large, globally dispersed organizations want system interoperability for strategic reasons (Cannata, 1991, Bertino, 1991). Public and private agencies together with hardware and software vendors are creating standards for the necessary telecommunications infrastructure. The same factor that motivated the central database concept in the 1960’s, isolation of related information, is motivating the development of heterogeneous database concepts and theory in the 1990’s.

Tapscott reinforces the notion of these two fundamental technology shifts, towards integrated systems and towards multimedia, in Paradigm Shift (Tapscott, 1993). The research presented here addresses multi-database interface design with particular emphasis on situations where there are significantly different data types. The study of such interfaces is not yet considered mature (Blattner & Dannenberg, 1992). The contribution is a reusable shell providing browsing, extraction, and manipulation capabilities.

The next two sections of this paper briefly describe the structural constructs of the object-oriented model which underpins the interface design, first discussing object representation and relationships and then object behavior in the database environment. This is followed by sections describing the interface, which was built on an IBM multimedia machine using Toolbook by Asymetrix. It also discusses the navigation and extraction capabilities and details the manipulation capabilities provided. Finally, a summary is presented.

Object Representation and Relationships

To develop an interface to multiple databases, a unified representation of the information space must be created. A variety of integrating data models have been used (Geller, Perl, & Neuhold, 1991). The object-oriented data model is particularly appropriate when there are wide differences in the kind of information stored at the different nodes (Czejdo & Taylor, 1991).

The schema of each database must be mapped to a common representation. The object-oriented approach maps schemas, that are based on efficient (or easily understood) access and storage mechanisms, back to a cognitive model based on objects in the user’s environment. It permits the user to view data of all types in a uniform way, at a level of
abstraction appropriate to his needs. The seminal paper describing object-oriented development is by Booch (Booch, 1986).

The object-oriented representation for the three databases in the test implementation is given in Figure 1. The diagrammatic technique employed is from Kroenke (1992). Rectangles represent object classes. Properties of a class are listed within the rectangle. If the properties themselves represent object classes, they are capitalized. The notation MV denotes multivalued properties.

Wonders and subclips are from a videodisc database. Maps are from a CD-ROM database. Sites and campgrounds (near wonders), cities, and states are derived from dBase III+ database files. Schema integration depends on an understanding of the semantics of the schema components in the participating databases. For more sophisticated applications case-based reasoning techniques can be used to semi-automate the schema integration process (Dao et al., 1991).

After establishing distinct object classes and their properties, relationships between these objects can be explored. Object classes A and B may be related in two important ways. A may be an object property of B (or vice versa), or A may be a subclass or superclass of B. These data abstractions are more fully described by Smith & Smith (Smith & Smith, 1977). These fundamental relationships should be identified so that navigation pathways between related class views can be facilitated.

**Object Behavior**

Object-orientation involves structural modeling of object classes as described in section 2 as well as the modeling of object class behavior. This message-passing behavior is described in detail in various literature (Cox, 1984; Stefik & Bobrow, 1986).

In database environments, this behavior is, essentially, access behavior, or the reading and writing of data values (Bertino, 1991). In multi-database systems, where update occurs in isolation, messages sent to objects and object classes, are generally retrieval operations.

The retrieval operations for the test implementation are listed in Figure 2. The operations are defined for each object class in terms of what makes sense from a (time, space, or logical) complexity standpoint out of a potential complete set of operations. A complete set of operations would allow retrieval of one instance by name, retrieval of all instances, and retrieval of all instance names. For each property X in the class, it might also be desirable to include operations for retrieving all instances with the same X value and retrieving all instance names with the same X value. Further combination and refinement of data retrieved, however, is alternately provided through manipulation utilities.

**Design of a Hypermedia Interface: Navigation and Extraction**

Once a set of objects has been identified and the relationships between the objects has been specified, the user interface for exploring object views or representations can be developed. A distinctive feature of the interface described is the
Related Content

Antecedents of the Closeness of Human-Avatar Relationships in a Virtual World
www.igi-global.com/chapter/antecedents-closeness-human-avatar-relationships/63666?camid=4v1a

A Unified Fuzzy Data Model: Representation and Processing
www.igi-global.com/article/unified-fuzzy-data-model/62033?camid=4v1a

Mobile Commerce Agents in WAP-Based Services
www.igi-global.com/article/mobile-commerce-agents-wap-based/3266?camid=4v1a

Elitist and Ensemble Strategies for Cascade Generalization
www.igi-global.com/article/elitist-ensemble-strategies-cascade-generalization/3359?camid=4v1a