Rich Base Schema (RiBS)$^1$: A Unified Framework for OODB Schema Version Management

SANG-WON LEE, Seoul National University
HYOUNG-JOO KIM, Seoul National University

In this paper, we propose a model of schema versions for object-oriented databases called RiBS. At the heart of this model is the concept of the Rich Base Schema (RiBS). Each schema version is in the form of a class hierarchy view over one base schema, RiBS, which accumulates all the schema information ever defined in the schema versions. Users, insulated from RiBS layer, access databases only through schema versions. Users impose schema evolution directly on schema versions, and the effects are, if necessary, automatically propagated to RiBS. We first describe the structure of the model and then introduce a set of invariants that should be satisfied by the model structure. As the third element of our model, we give a set of schema update operations, of which semantics are defined so as to preserve all the invariants.

Introduction

The functionality of schema evolution is one of the important differences between object-oriented database management systems (OODBMS) and relational database management systems (RDBMS). Object-oriented data models emerged in the mid 1980s and since then many approaches to schema evolution have been proposed (Banerjee, Kim, Kim, & Korth, 1987; Penney & Stein, 1987; Zicari & Ferrandina, 1997). This is because the applications of OODBMSs, such as CAD/CAM, CASE, and multi-media, require dynamic schema changes and flexible schema management. Currently, several commercial OODBMSs, such as GemStone (Penney & Stein, 1987, O2 (Zicari & Ferrandina, 1997), ObjectStore (Object Design, Inc., 1994), and Objectivity (Objectivity, Inc., 1998) support various schema update primitives and provide online schema evolution mechanisms.

Under these systems, however, only a single schema can exist at any time; if a schema evolution operation completes, the old schema is no longer maintained. This schema modification mechanism has several drawbacks (Kim, 1991). First, schema updates may invalidate programs written against old schema. Second, because all the users share a single schema, schema updates by one user may change the views of all the other users. In summary, the current schema evolution approach raises the problem of lack of logical data independence in OODBMSs. One of the most required characteristics of a good data model is logical data independence; the ability to modify the database schema without causing application programs to be rewritten. In a practical aspect, the property of logical data independence relieves the database users including the database administrator of much of the database application maintenance nightmare. In fact, much of the success of relational data model is attributable to its support for logical data independence via the concept of view. Considering the requirements for dynamic schema changes of OODBMS applications, we think OODBMSs also should be empowered with logical data independence at a level comparable to that of relational data model.

Schema version mechanisms were introduced to enhance the logical data independence of OODBMSs, and many researchers have stressed their importance since a characteristic of design applications should cope with frequent schema changes (Kim & Chou, 1988; Lautemann, 1996). Recently, the necessity for schema versions has been
rejuvenated in several new OODB applications including Repositories (Bernstein, 1998; Silberschatz, Stonebraker, & Ullman, 1995), Portable Common Tool Environment (PCTE) (Loomis, 1992), and the Word Wide Web (WWW) (Atwood, 1996; Yang & Kaiser, 1996), all of which may use an OODBMS as an integrator of data. Data repositories are expected to be one of the important new uses of DBMS technology (Silberschatz et al., 1995). Though much work has been done to provide schema version mechanisms for object-oriented databases (OODB) (Bertino, 1992; Kim & Chou, 1988; Monk & Sommerville, 1993; Ra & Rundensteiner, 1995), this field has not reached a satisfactory status. Traditional schema version approaches have three outstanding problems: (1) storage overhead for redundant objects (Kim & Chou, 1988; Ra & Rundensteiner, 1995), (2) limited schema update capability (Bertino, 1992), and (3) complexity for managing consistent schema versions (Monk & Sommerville, 1993). The RiBS model overcomes all these problems, enabling efficient and flexible schema version management. Our main contribution can be summarized as follows.

1. The RiBS model, the first view-based schema version approach, proposes a framework for mapping each schema evolution over schema version(s) to schema evolution(s) over base schema. This framework includes: (1) structural representation of schema versions and base schema, (2) a set of invariants, and (3) a set of schema evolution operations and its semantics.

2. The RiBS model solves the limited schema update capability of a traditional view-based schema evolutions and, unlike class versioning approach, allows for managing schema version more easily. That is, with the RiBS model, a user can impose the well-known schema evolution operations (Banerjee et al., 1987) over schema versions without any restrictions as in Bertino (1992), and their effects are reflected in database without user interventions as in Monk & Sommerville (1993).

3. Finally, the RiBS model does not incur any storage overhead; every physical object resides only in RiBS, and every object accessed from schema versions are view objects of its corresponding physical object.

The remainder of this paper is organized as follows. Next, a brief overview of the RiBS model using an illustrative example is given. Then, the object model assumed in this paper is described. This is followed by a detailed description of each component of the RiBS model, which includes the (1) structural part, (2) a set of invariants, and (3) schema evolution operations and their semantics, respectively. A few issues about the implementation of the RiBS mode are given and the work is compared to related work. Finally, the paper concludes with a summary and an outline of future work.

Preliminary

In this section, we illustrate some basic ideas of the RiBS model with an example. For brevity, we assume the following informal description of a schema in an OODB: A schema in a database consists of classes that are organized into a class hierarchy through ‘is-a’ (ISA) relationships between them. Each class, in turn, consists of properties including both attributes and methods. To every class is attached a collection of objects, extent. Each instance object belongs to the extent of a single class, and is referred to as a direct instance of the class.

Rich Base Schema and Schema Versions

Before proceeding with the example, we introduce the concept of “Rich Base Schema,” and discuss how it can be exploited in supporting schema versions. We say that a schema S1, is richer in schema information than the other schema S2 if all the following conditions hold.

1. Every class in S2 has a corresponding class in S1.
2. Every property in S2 has a corresponding property in S1.
3. Every direct ISA relationship in S2 has a corresponding (direct or indirect) ISA relationship in S1.

If S1 is richer than S2, it means intuitively that S1 has more schema information than S2. This, in turn, means that S2 can be specified as a view of S1. This concept of rich schema can be restated in terms of relative information capacity (Hull, 1984); S1 dominates (or subsumes) S2. Our model is based on this concept of rich schema. A physical base schema, RiBS (Rich Base Schema), which is richer in schema information than any schema version, is maintained, and each schema version is represented as a view over RiBS. In addition, when a schema update is imposed on a schema version, RiBS, is, if necessary, automatically augmented so as to be richer than the modified schema version in addition to all other ones. In summary, a schema version is an updatable class hierarchical view over RiBS, in the sense that schema evolution operations can be directly imposed on the view.

In our model, schema versions are strictly separated from RiBS. This separation prevents several problems arising when the schema information of schema versions is mingled with that of RiBS. Some previous works on views in OODB (Abiteboul & Bonner, 1991; Bertino, 1992) put normal classes and derived views together in a class hierarchy. However, this approach has several disadvantages (Kim, 1995). First, it is difficult to understand the complicated class hierarchy. Next, the extents of classes may overlap. Finally, it is difficult and, in certain cases impossible, to decide where to locate the view class in the class hierarchy.

An Intuitive Example

Now let us consider the example in Figure 1, where two