Chapter II

Comparing Metamodels for ER, ORM and UML Data Models

Terry Halpin, Northface University, USA

ABSTRACT
This chapter provides metamodels for some of the main database modeling notations used in industry. Two Entity Relationship (ER) notations (Information Engineering and Barker ER) are examined in detail, as well as Object Role Modeling (ORM) conceptual schema diagrams. The discussion of optionality, cardinality and multiplicity is widened to include Unified Modeling Language (UML) class diagrams. Issues addressed in the metamodel analysis include the normalization impact of non-derived constraints on derived associations, the influence of orthogonality on language transparency, and trade-offs between simplicity and expressibility. To facilitate comparison, the same modeling notation is used to display each metamodel. For this purpose, ORM is used because of its greater expressibility and clarity.

INTRODUCTION
To ensure the correctness and completeness of an information system being developed, requirements analysis should precede its design and implementation. The analysis phase leads to a conceptual schema that specifies the structure of the universe of discourse (application domain). This conceptual structure should be capable of being readily understood and validated by the domain expert, without requiring this subject matter expert to understand technical aspects of the internal structure used to actually implement the application. Once validated, the conceptual schema can be mapped to logical/physical/external schemas using procedures that are partly or fully automatable.
For industrial database work, the traditional approach for high level data modeling is to use a version of Entity Relationship (ER) modeling (Chen, 1976), such as the Information Engineering (IE) approach (Finkelstein, 1998), the Barker version of ER modeling (Barker, 1990), or IDEF1X (Integration Definition 1 extended). Although the original 1993 version of IDEF1X has a standard metamodel (NIST, 1993), we ignore it here since it is actually a hybrid of ER and relational modeling, and its successor, IDEF1X97, also known as IDEFobject (IEEE, 1999), has so far been largely ignored by the marketplace.

More recently, Unified Modeling Language (UML) class diagrams (OMG, 2003) and the Object-Role Modeling (ORM) approach (Halpin, 2001a) have also gained popularity for information modeling. Following its adoption by the Object Management Group (OMG), the UML is now the de-facto standard in industry for object-oriented code design. ORM is a fact-oriented approach that can be used as a conceptual front-end to attribute-based approaches such as ER and UML, and is currently being considered by the OMG’s Business Rules Special Interest Group as a candidate for business rule modeling at the computation-independent level.

A modeling language can be specified by a metaschema, which is a schema that indicates the grammatical structures to which any application schema formulated in the modeling language must conform. Strictly, a model is the union of a schema (structure) and a population of instances (e.g., objects or facts that instantiate the information-bearing structures in the schema). A metaschema supplemented by structures to capture specific populations is a metamodel. In practice, the term “metamodel” is sometimes loosely used as a synonym for “metaschema”. While published metamodels for UML (OMG, 2001, 2003) have been widely debated, and many suggestions have been made to improve UML (e.g., see Siau & Halpin, 2001), it is difficult to find any in-depth analysis of metaschemas for the other approaches. This paper provides new metaschemas for two ER approaches (IE and Barker) as well as ORM to reveal their commonalities and differences, and to address modeling issues such as the use of derived associations and the virtues of orthogonality. UML has been examined previously (e.g., Halpin & Bloesch, 1999; Halpin, 2001b) and is quite complex; hence only an incomplete analysis of its metamodel for data modeling is given here. For a detailed comparative evaluation of all the methods, including IDEF1X, see Halpin (2001a).

The next section of this chapter provides a metaschema and related discussion of the IE notation. The two sections after that metamodel the Barker ER and ORM approaches, respectively. We then evaluate the different approaches to multiplicity in UML, ER and ORM. Some other aspects of the UML metamodel are then discussed. The final section summarizes the main contributions, notes some advantages of an attribute-free modeling approach, and lists references for further reading.

**INFORMATION ENGINEERING**

The Information Engineering approach was originated mainly by Clive Finkelstein, who developed a modeling procedure for the notation and extended IE to Enterprise Engineering (EE). Finkelstein (1998) provides an overview of IE with further details on his website (www.ies.aust.com/~ieinfo/). The IE notation was later adapted by Martin (1993). Although Martin’s recent books favor the UML notation, IE is still used far more extensively for database design than UML, which is mostly used for object-oriented code design. Different versions of IE exist, with no single standard. In one form or another, IE has long
Related Content

Preparing Clinical Text for Use in Biomedical Research
[www.igi-global.com/article/preparing-clinical-text-use-biomedical/3350?camid=4v1a](www.igi-global.com/article/preparing-clinical-text-use-biomedical/3350?camid=4v1a)

Integrity Maintenance In Extensible Databases
[www.igi-global.com/chapter/integrity-maintenance-extensible-databases/7886?camid=4v1a](www.igi-global.com/chapter/integrity-maintenance-extensible-databases/7886?camid=4v1a)

Introduction and Trends to Fuzzy Logic and Fuzzy Databases
[www.igi-global.com/chapter/introduction-trends-fuzzy-logic-fuzzy/20347?camid=4v1a](www.igi-global.com/chapter/introduction-trends-fuzzy-logic-fuzzy/20347?camid=4v1a)
UB2SQL: A Tool for Building Database Applications Using UML and B Formal Method
www.igi-global.com/article/ub2sql-tool-building-database-applications/3363?camid=4v1a