Transformational Object-Relational Database Model in Formal Methods

PUJIANTO YUGOPUSPITO[1,2] AND KEIJIRO ARAKI[1]

[1] Department of Computer Science and Communication Engineering
Graduate School of Information Science and Electrical Engineering
Kyushu University,
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, JAPAN
E-mail: (pyugop, araki)@dontaku.csce.kyushu-u.ac.jp

[2] PT. IPTN, Indonesia Aerospace Industries.
Jl. Pajajaran 154, Bandung 40174, Indonesia

Object-Relational Database is an emerging breed of database that combines the state-of-the-art Relational Database with object-oriented capabilities. The transformation of Relational Database to Object-Relational Database is a preferable option. The correctness and completeness should be guaranteed during the transformation. Since the ambiguity of conventional representations of database model, e.g. UML diagrams and the diversities of SQLs, utilization of formal methods can be beneficial. Furthermore, formal specification makes further reasoning and proof of considered cases able to be established. We proposed the transformation model and the database process new sequence. The formal specification is given in Z notation.

1. Introduction

The motivation of this paper is to anticipate the "great wave' of Object-Relational Databases (ORDB)[6]. The condition is given by a long standing Relational Database (RDB) that faces a problem of supporting object-oriented applications, e.g. aircraft manufacture company database for demanding support of object-oriented software applications. Transformation to object-relational type is a preferable choice than migration to object-oriented database, because the hard core of the legacy of RDB in industry can be preserved, while a complete re-development of operated softwares are required for radical migration to an object-oriented one[1),4)].

In general, the current state-of-the-art RDB could not satisfy the emerging force from users of information technology to support the object-oriented programming language. The requirement to support multimedia data types and the fast growing internet are the technological driver for object-relational database.

We expect that by giving a precise transformation of RDB to ORDB, we will help the object-oriented software developers to have a better understanding of the current running relational database model in an object-oriented view.

The conventional logical design of database is represented in diagrammatic model then it is created by structured query language (SQL) statements. The correlation of diagrams and the SQL statements cannot be guaranteed. We proposed that the formal methods approach be used to guarantee that correct transformations. We considered that the RDB model is originated from IDEF1X while the ORDB model is proposed to be represented in an Unified Modeling Language (UML) class Diagram.

We believe that Formal Methods is an approach that can rigorously describe the informal representation of relational and object-relational model, IDEF1X and UML; furthermore we can make a further reasoning for those models. We apply Z notation for those formal specifications. Since we also captured the SQL statements in the same level of formality then the logical refinement of those sets of formal specifications can be proceeded with.

The structure of this paper is the object-relational database in the next section, followed by the approach model. The formal specification model is presented after that. Then, a discussion about further reasoning. We concluded our paper with the conclusion and future works, after discussing the sequence data base design.

2. Object-Relational Database

Many vendors and researchers are still arguing about the true object-relational database. In our opinion the main feature of the object-relational data model is a relational data model with extension features. This opinion is supported by our selected strategy of the incremental evolution of relational database to obtain an object-relational database. The main additional features that accepted by almost all researchers are *base type extension*, *complex type*, *inheritance* and *rules*. The ultimate goal of an evolution of object-relational database is to be a fully object-oriented database[2)] while preserving its legacy of relational database.

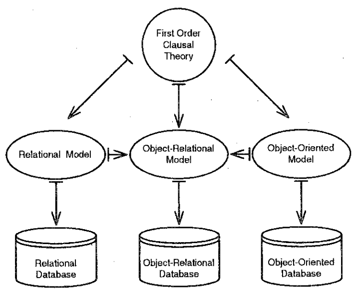

Fig. 1 Space Theories of Databases

We considered a mapping from FOCT to ORDM, in which it will include representation assumptions, object-relational assumptions and object-relational dynamic assumptions. We considered object-relational data model as a sink of relational data model and object-oriented data model. **Figure 1** depicts a space theory for Relational, Object-Oriented and Object-relational databases as modified space theories of Fernandez[3)].

3. The Approach Model

The utilization of formal methods is depicted in **Fig. 2**. The problem of transformation relational database to object-relational database is initiated by extracting the current relational database to its logical design representation in the IDEF1X diagram. An intermediate informal solution can be provided by mapping IDEF1X to a class diagram of UML. Formal specifications based on those diagrams by considering the domain problem can be given. The rigorous solution is provided by proving a correlation between RDB and ORDB specifications.

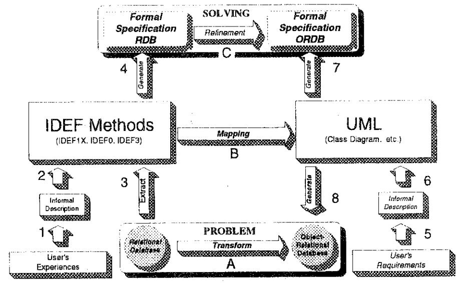

Fig. 2 The Approach Model

The mapping of IDEF1X to a UML class diagram can be trivially given because IDEF1X and UML class diagrams have the same topography[8)]. A set of formal specification of IDEF1X has been presented[7)] by adapting from Kouno's works[5)] with modifications.

4. Formal Specification Model

The model of transforming a relational database to an object-relational database is depicted in **Fig. 3**. The complete transformation can be achieved in two stages; first a bare transformation from an relational database to an object-relational database without additions and then adding the additions to the object-relational database. The additions, e.g. embedded methods, have no counterparts in the relational database model and it must therefore be done by the users. So we concentrated on the first step.

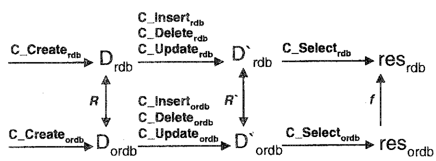

Fig. 3 Transformation Object-Relational Database Model

We make the assumption that there is a relation between SQL commands on an relational database and the corresponding SQL commands on object-relational database. An ORDB may have a richer SQL language, but for correctness we are only concerned to present that we can do in an ORDB as whatever we can do in an RDB. It is possible that a primitive RDB command may be modeled by a more complex ORDB command, but in practice we do not expect this as SQL-92 is essentially a subset of SQL-3.

We assume that we can divide the SQL commands on a database into three groups: *creation* commands that create database values, *manipulation* commands that manipulate the RDB values but do not generate output, and *selection* commands that generate output but do not change database values. This assumption is just to simplify the analysis. The equality result(s) of selection commands between RDB and ORDB should be held by establishing equivalent function f between both results.

The correlation between both SQL commands can be regarded as more abstract operation to a more concrete operation as *refinement* correlation is held. If the relation R represents the correlation between RDB specifications, Sp_r to ORDB specifications Sp_o, Op_r is the concerned operation in RDB and Op_o is the corresponding concerned operation in ORDB. Then we can apply the logical *refinement* conditions: *Initialization*, *Applicability (Safety)*, *Correctness (Liveness)* defined as below respectively.

$$\forall Sp_o' \bullet InitSp_o \Rightarrow (\forall Sp_r \bullet InitSp_r \wedge R')$$

$$\forall Sp_r;\ Sp_o \bullet \mathrm{pre}\ Op_r \wedge R \Rightarrow \mathrm{pre}\ Op_o$$

$$\forall Sp_r;\ Sp_o;\ Sp_o' \bullet \mathrm{pre}\ Op_r \wedge Op_o \wedge R \Rightarrow (\exists Sp_r \bullet Op_r \wedge R')$$

4.1 Specifications in Glance

A table of relational database consists of entities *Entity* that constructed by records *Record* that primary keys *PKeyAtt* and other keys, *OKeyAtt* as the smallest construct. Those data-type of keys are pre-defined data-types.

$$\begin{array}{l} \underline{\ Record[PKeyAtt, OKeyAtt]\ } \\ p : PKeyAtt \\ o : OKeyAtt \\ \hline \end{array}$$

$$\begin{array}{l} \underline{\ Entity[PKeyAtt, OKeyAtt]\ } \\ entity : \mathbb{P}\ Record[PKeyAtt, OKeyAtt] \\ eType : EntityType \\ \hline \forall r1, r2 : entity \bullet r1.p = r2.p \Rightarrow r1 = r2 \\ \hline \end{array}$$

Supposing a database consists of a vendor entity that has primary key, *vendorNo* and an other key *vendorName*. Then the specification for vendor entity *VendorEnt* is given as:

$$\begin{array}{l} [Char] \\ VendorPK \mathrel{\widehat=} [vendorNo : \mathrm{seq}\ Char] \\ VendorAtt \mathrel{\widehat=} [vendorName : \mathrm{seq}\ Char] \\ VendorRec \mathrel{\widehat=} Record[VendorPK, VendorAtt] \\ VendorEnt \mathrel{\widehat=} Entity[VendorPK, VendorAtt] \\ \quad [vendorRecs / entity] \end{array}$$

An object-relational database is free constructed by the user-defined objects. For the same example above, vendor class *UVendorExt* is constructed by object *UVendor* with a user-defined type *VndNo* and *VndName* and the *VndID* as the default object-id then specifications will be:

$$\begin{array}{l} [VndID] \\ VndNo == \mathrm{seq}\ Char \\ VndName == \mathrm{seq}\ Char \end{array}$$

$$\begin{array}{l} \underline{\ UVendor\ } \\ vendorId : VndID \\ vendorNo : VndNo \\ vendorName : VndName \\ \hline \end{array}$$

$$\begin{array}{l} \underline{\ UVendorExt\ } \\ uvendors : \mathbb{F}\ UVendor \\ \hline \forall uv1, uv2 : uvendors \bullet uv1 = uv2 \Leftrightarrow \\ \quad uv1.vendorId = uv2.vendorId \\ \quad \wedge\ uv1.vendorNo = uv2.vendorNo \\ \hline \end{array}$$

We introduce a function f that takes the result of the selection command of ORDB unstructured. The result is such that the value of ORDB result can be shown as the value of RDB result.

$$\begin{array}{l} f : UVendor \rightarrowtail\!\!\!\!\!\!\!\!\!\;+\;\; VendorRec \\ \hline \mathrm{dom} f = UVendor \\ \forall v : VendorRec;\ uv : UVendor \mid f(uv) = v \\ \quad \bullet\ v.p.vendorNo = uv.vendorNo \\ \quad \wedge\ v.o.vendorName = uv.vendorName \end{array}$$

A correlation between the entity-class can be given as:

$$\begin{array}{l} \underline{\ MappingVendorEntityClass\ } \\ VendorEnt;\ VendorExt \\ \hline \forall vs : vendorRecs;\ uvs : uvendors \\ \quad \bullet\ f(uvs) = vs \\ \hline \end{array}$$

The refinement conditions can be applied on this correlation of $VendorEnt \sqsubseteq UVendorExt$. The proof can be provided in the Z/Eves tool.

5. Further Reasoning

The mapping result of UML class diagram can be refined as a kind of schema evolution of databases. These pictorial refinements can be reasoned in formal specification that is given by the ORDB specification. Since it is given in a precise description, the proof of this refinement can be supported.

A refinement process can be constructed by more basic refinement processes; those are *augmentation,reduction* and *preservation*. The basic refinement processes can be applied to any components of the UML class diagrams, e.g. attributes, operations, classes, and relationships. A detail information of this extension is presented in 9)

6. Sequence of Database Design

In conjunction to the formal method existence in database design, we proposed the sequence of transformation RDB to ORDB as shown in **Fig. 4**. Currently, the established automatic processes are the generation of Z specification from UML class diagram, the generation of SQL syntax of persistence classes. Those are available inside the Rational Tool by adding some scripts. The reasoning, analysis and proving of the specifications can be provided by the Z/Eves tool, as its input is the output of the Rational Rose tool.

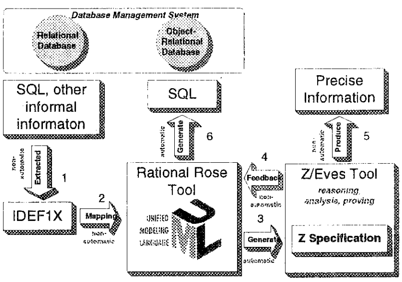

Fig. 4 Sequence of Databases Design Process

7. Conclusion and Future Works

A significant part of the transformation of RDB to ORDB has been shown as a logical refinement correlation. Its correctness and completeness can be observed in formal methods such that its clear understanding can be expected. We considered that this is a significant result to be applied to the rest of the complete components.

Formal methods have been presented as a tool for reasoning and producing a better understanding of a problem. Since it uses the precise description in mathematical notations, then the proof can be shown. Its effectiveness depends on the tools' availability.

The tools' integration between Rational Rose and Z/Eves can be investigated and provided for practical purposes. Further work on complete translation of UML to Z specification can be expected to review the other domain problems, e.g, software engineering domain.

Acknowledgments We would like to express our thanks to Tanaka Toshiyuki, Nazir A. Zafar and Mr. Chris George (UNU/IIST-Macau) for their valuable input and fruitful discussions. Personally, Yugopuspito would like to thanks to the Minister of Science and Technology of the Republic of Indonesia who provides the scholarship for studying in Japan.

References

1) Behm, A., Geppert, A., Dittrich, K. R: On the Migration of Relational Schemas and Data to Object-Oriented Database Systems, *International Conference on Re-Technologies in Information Systems Proceeding* Klagenfurt, Austria (1997).
2) Demphlous. S: Database Evolution: A Metaobject Approach, *Third International Workshop on Advances in Databases and Information Systems ADBIS'96 Proceeding*, ACM SIGMOD Chapter and Russian Foundation for Basic Research, vol. 1, pp. 31–37. Moscow Engineering-Physical Institute (MEPhI) publisher (1996).
3) Fernandez A. A. A.: *An Axiomatic Approach to Deductive Object-Oriented Databases* Ph.D thesis, Dept. Computer Science, Heriot-Watt University (1995).
4) Joeris, G. and Hertzog, O: *Managing Evolving Workflow Specification with Schema Versioning and Migration Rules*, TZI Technical Report 15, University of Bremen (1999).
5) Kouno, S., Chang, H-M, Araki, K.: Consistency Checking Between Data and Process Diagrams Based on Formal Methods, *COMPSAC'96 Proceeding*, pp. 261-269 (1996).
6) Stonebraker, M., and Brown, P. with Moore, D: *Object-Relational DBMSs Tracking the Next Great Wave*, Morgan Kauffman Publisher, Inc. (1999).
7) Yugopuspito, P. Taguchi, K., and Araki, K: Formalizing IDEF1X in Z Specification, *International Symposium on Future Software Technology, ISFST 1998 Proceeding* (1998).
8) Yugopuspito, P. and Araki, K: Re-engineering of Relational Database into Object-Relational Database in Informal and Formal Levels. *International Symposium on Future Software Technology ISFST 1999 Proceeding* (1999).
9) Yugopuspito, P. and Araki, K: UML for Object-Relational Database and Its Refinement, *International Symposium on Future Software Technology ISFST 2000 Proceeding* (2000).