TYPES IN OBJECTBASES DEVELOPMENT

Adoración de Miguel[1]

Escuela de Informática
Universidad Carlos III
Campus de Getafe
28 MADRID (SPAIN)
phone (+34)(+1)
fax (+34) (+1)

Mario G. Piattini

Departamento de I + D
Cronos Ibérica, S.A.
c/Clara del Rey N° 8
28002 MADRID (SPAIN)
phone (+34)(+1) 4138060
fax (+34) (+1) 4131419

Abstract

During the later years a significant number of data models for objectbases (object models) have been proposed introducing different categories of data types. However, these models usually covers only implementation issues, not considering types throughout all the development phases. In this paper a classification of types in object models is presented and their role along the different development phases is discussed.

Key words:

Data types, Object model, OODBMS, ODMG, SQL3.

1.- INTRODUCTION

Object-oriented technology has spread over very different software areas. As a result, a significant number of different object models have been proposed. Specifically, Objectbase Management Systems (OBMS's) appear in the market without a broadly accepted formal model, the same as the first generation -hierarchical and Codasyl- products. Each of these OBMS products implements a different model, in general, not defined in a formal and precise way. An exception, in this sense, is the O_2 Model, KANEKALLIS, P. et al. (1992), although it is focused more to implementation aspects than analysis or design ones.

Recently, the mainly "purist" vendors have agreed in an objectbase model known as "ODMG-93", CATTELL (1994), as an extension of the OMG model. At the same time, the most important relational database manufacturers have been collaborating within ANSI and ISO committees to elaborate the next version of the SQL language ("SQL3"),

[1] This paper has been written while the author was at the Facultad de Informática, Universidad Politécnica de Madrid.

MELTON (1994). These models proposed so far focus on implementation issues, neglecting some important concepts necessaries in the analysis and design phases. So, all the needs we have in developing an objectbase from analysis to construction are not well supported. Moreover, the model proposed by the ODMG is disjoint from the SQL-92 as is demostrated in KIM (1994). In our opinion a model that includes relational model and allows interoperability between relational and objectbases is needed.

Objectbase concepts are also forgotten in many formal object models like HONG and MARYANSKY (1990) or WAND (1989). During the definition of a methodology for objectbases development named "MEDEA", DE MIGUEL and PIATTINI (1995), a formal object-oriented data model has been defined. This model combines the most important OO concepts with the main advances in conceptual modelling and relational database extensions. Whole compatibility with SQL-92 and SQL3, as in MELTON (1994), was a main objetive. Our work followed the pattern proposed by JENSEN et al. (1992) for temporal databases. stablishing the definition, sinonyms, discussion and description of each concept.

Several versions of our model were defined in the development of a PhD Thesis, PIATTINI (1994). The model has been validated as a basis for the MEDEA methodology. The model has also been implemented in the core subsystem of an OOIRDS (Object-Oriented Information Resource Dictionary System[2]). This last work demostrated the validity, need and appropiateness of the objectmodel's types and also its self-description capability.

Due to space limitations in this paper, we cannot present the whole model. We just discuss the types defined in the model and present some conclusions of our work.

2.- TYPES

As we can see in fig. 1, we consider that in the real world there are "objects" and "values", which are the "elements" we want to represent in the database system. This distinction between objects and values is very important and usual in some objectbase models, like ODMG-93 -CATTELL (1994)- and SQL3 -MELTON (1994)-, and in object-oriented technology in general. We have kept this distinction throughout all the development phases.

An object is characterised for having an state that could change, an unique identity, relationships with other objects, a behaviour, and sending and receiving messages. The identity could be implemented as an OID (object identifier) assigned by the objectbase management system or as an inmutable primary key defined by the user, keeping the compability between object models requiring object identifier and systems based on relational model. The exigence of primary key inmutability could have been relaxed, but we have kept it in order to follow more strictly object-oriented principles.

[2] Treated in a thesis that is to be finished in next months.

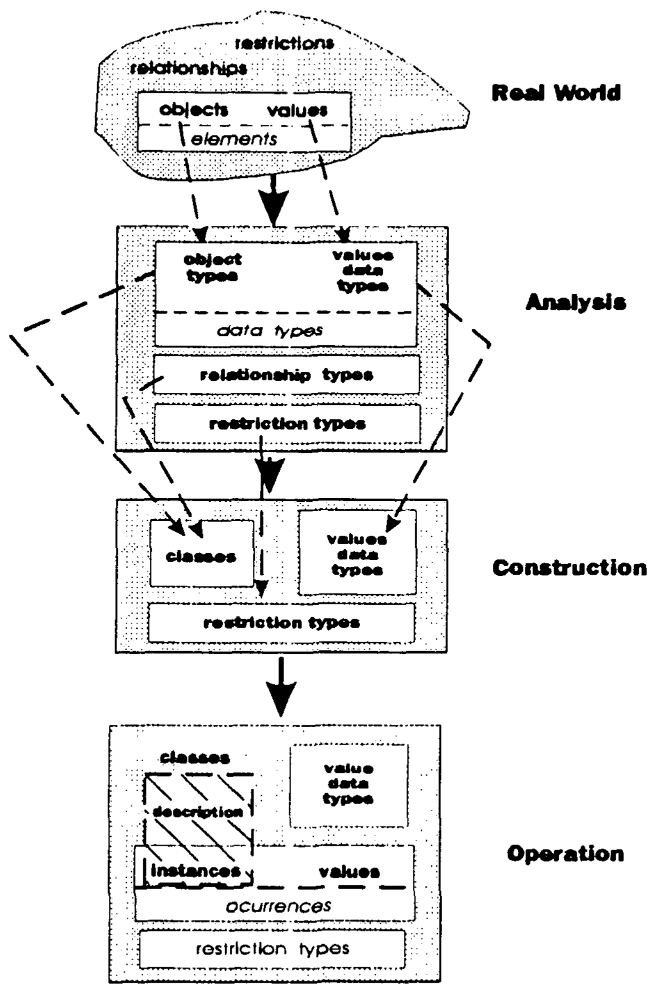

Fig 1.- Types throughout development phases

Following with the distinction between objects and values, a value is inmutable, has no identifier (it is represented by itself), does not hold relationships, and is manipulated by arithmetic or comparison operations.

The objects of the real world have properties (attributes) and present a behaviour (seen as a set of services). Both objects and values are subjected to restrictions.

In the analysis phase the objects are described by "**object types**" and the values by "**values data types**". The value data types could be:

- basic types, which cannot be built over other types: numeric (integer, float), character, boolean and bit.
- OID type, which supports system generated identifiers;
- domains (extended types), which describe a set of values of the same basic type;
- NOADT (Not object abstract data types), which are constructed from others NOADT, domains, OID or basic types.

We consider NOADT and object types as abstract data types. Abstract data types have characteristics: attributes, services and restrictions.

Data types could also be classified in two other dimensions: such as

- *primitives* (built-in or predefined), provided by the system or the administrator
- *specifics*, defined by the users;

or, in other dimension, as:

- *simple*, if the elements described by the data type -always values- are atomics.
- *composed*

Some NOADT (e.g. complex number) and object types are in this category. For composed data types we agree with the definition exposed in ODMG-93 about "Inmutable Collection/Structure" and "Structured objects" that could be collections or structures.

Others authors, as DATE (1992) and some standards like SQL-3, MELTON (1994), consider that basic types are the same as primitive or built-in types. In our model, these last two dimensions are completely orthogonal. For example, we consider "boolean" as a basic type that sometimes is not provided by the system (so is not a primitive type), and "complex" is a NOADT that could be primitive.

In the construction phase (which groups design and implementation phases) "classes" are present (as the implementation of several object types and/or relationship types), together with values data types and restriction types.

In the operational system, classes do not have only their intensional part but also their extensions (instances). We consider classes both as a "*factory*" and a "*warehouse*" of instances. In this context values are also present, an instance or a value is called an "ocurrence".

We can summarize the main characteristics of these types, considering for each if they have or not structure, behaviour, restrictions, etc. as follows:

	BASIC	DOMAIN	NOADT	OBJECT TYPE
STRUCTURE	YES	YES	YES	YES
BEHAVIOUR	NO	NO	YES	YES
RESTRICTIONS	NO	YES	YES	YES
RELATIONSHIPS	NO	NO	YES[3]	YES
IDENTIFIER	NO	NO	NO	YES
DEFAULT VALUES	NO	YES	YES	NO
VALUES	YES	YES	YES	NO

3.- CONCLUSIONS

The database community needs a precise and formal object model which can serve not only as an implementation pattern but also as a basis of a development methodology. A model in which object-oriented, extended relational and conceptual modelling concepts are combined. One important aspect of this model, types, has been discussed in this paper.

Our type system comprises either the SQL-92, SQL3 and OMG-93 types, considering also the types in the analysis and design phases. The integration of object-oriented, conceptual modelling and relational extensions, convert this model in a appropiate candidate to support the interoperability among different models, ÖZSU et al. (1994).

We have implemented this type system in a OOIRDS, deducing that it is complete and adequate in the practice.

Future work.

We are now using the MEDEA methodology in different projects and we want to extend the model in the following directions: include temporal aspects, integrate active and deductive databases concepts and cover distributed issues. A CASE tool supporting the methodology is to be built using one of the commercial "meta-CASE" available at the moment, in which the whole model could be defined.

ACKNOWLEDGMENTS

The authors wishes to thank Begoña Voyer, co-developer of the model.

[3] We considered different kinds of relationships: **levelled,** if the objects involved have the same range, **hierarchized,** if there is a subordination between the participants in the relationship. In this case it could be distinguished as: existence and dependency relationships, generalization (inheritance) or meronymic relationships, as in WINSTON (1987). NOADT only supports generalization and meronymic relationships.

REFERENCES

ANSI (1991)
"Final Technical Report". Accredited Standards Committee X3, Information Processing Systems DBSSG/OODBTG Final Report. Fong, Kent, Moore y Thompson (eds.), 17th september.

CATTELL, R. G. G. (1994)
"The Object Database Standard ODMG-93". Morgan Kaufman, San Mateo, CA.

CODD, E.F. (1990)
"The Relational Model for Database Management: Version 2". Reading, Mass. Addison Wesley, Reading, Massachussetts.

DATE, C.J. (1992)
"Date on Databases". In: Proc. of the Codd & Date Relational Database Symposium", Madrid, 16-18 March.

DE MIGUEL and PIATTINI (1995)
"MEDEA: A Methodology for object-oriented database development". Submitted to the BIWIT (Basque International Workshop on Information Technology), San Sebastián, 19-21 July 1995.

HONG, S. and MARYANSKY, F. (1990)
"Using a Meta Model to Represent Object-Oriented Data Models". In: Proc. of the Sixth International Conference on Data Engineering, Los Angeles, California, 5-9 February.

JENSEN et al. (1992)
"A Glossary of Temporal Databases". In: SIGMOD RECORD, June.

KANEKALLIS, P. et al. (1992)
"Introduction to the Data Model". In: Building an Object-Oriented Database System. The Story of O_2". Morgan Kaufmann Publishers. San Mateo, California.

KIM, W. (1994)
"Observations on the ODMG-93 Proposal". En: SIGMOD RECORD Vol. 23, N° 1, marzo, pp. 4-9.

MELTON, J. (1994)
"(ISO/ANSI working draft) Database Language SQL3". ISO/IEC JCT1/SC 21 WG3 N1223, September.

MONARCHI, E.D. and PUHR, G.I. (1992)
"A Research Typology for Object-Oriented Analysis and Design". In: CACM Vol 35 N° 9, September.

OMG (1992)
"The OMG Object Model". Object Management Group.

ÖZSU, M.T. et al. (eds.) (1994)
"Distributed Object Management". Morgan Kaufmann.

PIATTINI, M. (1994)
"Definición de una metodología de desarrollo para bases de datos orientadas al objeto fundamentadas en extensiones del modelo relacional". PhD Thesis, Facultad de Informática, Universidad Politécnica de Madrid.

WAND, Y. (1989)
"A Proposal for a Formal Model of Objects". En: Object-Oriented Concepts, Databases, and Applications". KIM y LOCHOVSKY (eds.), ACM Press. Addison-Wesley, Reading, Massachusetts.

WINSTON, M.E. et al. (1987)
"A Taxonomy of part-whole relations". In: Cognitive Science Vol. 11, pp. 417-444.