Object-relational database metrics

Mario Piattini[1], Coral Calero[1], Houari Sahraoui[2], Hakim Lounis[3]
[1]*Departament of Computer Science*
University of Castilla-La Mancha
Ronda Calatrava, 5, 13071 Ciudad Real (Spain)
e-mail: {mpiattin, ccalero}@inf-cr.uclm.es

[2] *Dep.d'Informatique et Recherche Opérationelle*
Université de Montréal
Montréal (Canada)
e-mail: sahraouh@iro.umontreal.ca

[3] *Dep.d'Informatique*
Université de Quebec a Montréal
Montréal (Canada)
e-mail: hakim.lounis@uqam.ca

ABSTRACT. *Object-relational databases will replace relational systems to become the next great wave of databases because they combine traditional database characteristics with object-oriented principle. So, it is fundamental to propose metrics for control the quality of this kind of databases. But metrics definition must be done in a methodological way, it is necessary to follow a number of steps for ensure the reliability of the proposed metrics. In this work, we present the method we apply for the metrics proposal (which is composed by metrics definition, formal verification and empirical validation) and how we have used it obtain metrics for object-relational databases.*

RÉSUMÉ. *Les bases de données objet-relationnelles substitueront les systèmes apparentés pour devenir la prochaine grande vague des bases de données parce qu'elles combinent des caractéristiques traditionnelles de base de données avec le principe orienté a l'objet. Ainsi, il est fondamental proposer des métriques pour controller la qualité de ce genre de bases de données. Mais la définition de métriques doit être faite d'une voie méthodologique, il est nécessaire suivre un certain nombre d'étapes pour assurer la fiabilité de les métriques proposées. Dans ce travail, nous présentons la méthode que nous utilisons pour la proposition des métriques (qui se compose par définition de métrique, validation formelle et validation empirique) et comment nous l'avons utilisée pour obtenir des métriques pour les bases de données objet-relationnelles.*

KEY WORDS. *Object-relational databases, metrics, quality, theoretical validation, empirical validation.*

MOTS-CLÉS. *Base de données objet-relationnelles, métriques, qualité, validation théorique, validation empirique.*

1. Introduction

Metrics for databases have been neglected in the metric community ([SNE 98]). Most all of the metrics proposed from the McCabe ([MCA 76]) famous cyclomatic number until today have been centered in measuring programs complexity. However, in modern Information Systems (IS) databases have become a crucial component, so there is a need to propose and study some measures to assess its quality. It is important that databases are evaluated for every relevant quality characteristic using validated or widely accepted metrics. These metrics could help designers, to choose the most maintainable, among semantically equivalent alternative schemata. Moreover, the object-relational databases will replace relational systems to become the next great wave of databases ([STO 99]) so, it is fundamental to propose metrics for control the quality of this kind of databases.

Database quality depends on several factors, one of which is maintainability ([ISO 94]). Maintenance is considered the most important concern for modern IS department and requires greater attention by the software community ([FRA 92], [MCL 92], [PIG 97]). Maintainability is affected by understandability, modifiability and probability which depend on complexity ([LI 87]). Three types of complexity can be distinguished (HEN 96]): human factor complexity, problem complexity and product complexity. We focus our work in this last kind of complexity.

We have put forward different measures (for internal attributes) in order to measure the complexity that affects the maintainability (an external attribute) of the object-relational databases which is useful for control its quality.

In this paper we present on section 2 the framework used for metrics definition, metrics proposed for object-relational databases come in section 3. In section 4 we present the formal verification of some of the metrics. We show two experiments made to validate our metrics in section 5, in this section both experiments and the results obtained for each one are described. Finally, conclusions and future work come on the last section.

2. A Framework for Developing and Validating Database Metrics

As we have said previously, our goal is to define metrics for controlling object-relational databases maintainability, through metrics that capture complexity. But metrics definition must be done in a methodological way, it is necessary to follow a number of steps for ensure the reliability of the proposed metrics. Figure 1 presents the method we apply for the metrics proposal.

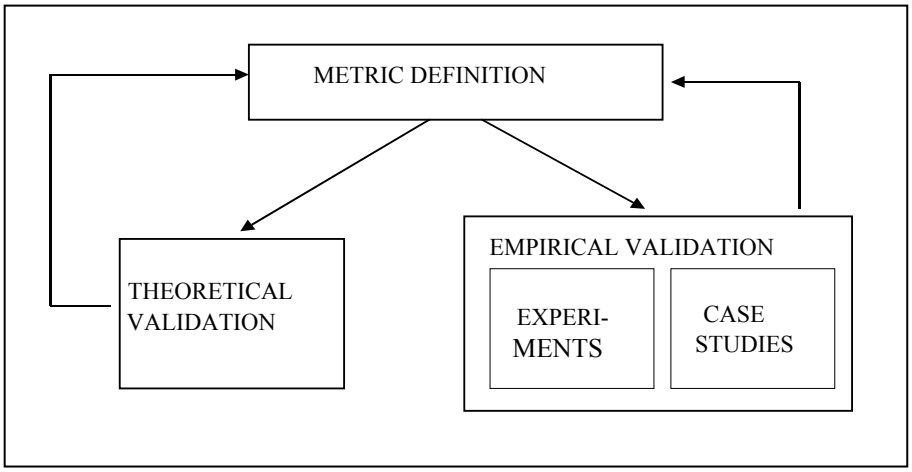

Figure 1. Steps followed in the definition and validation of the database metrics

The first step is the metrics proposal. This step must be made taking into account the specific characteristics of the object-relational databases and the experience of database designers and administrators of these databases. One methodological way to made the metrics proposal is by following the Goal-Question-Metric (GQM) approach. The goal of this approach is based in the fact that any metric can be defined by a top-down design with three levels, the conceptual level (Goal) where the objectives are defined, the operational level (Question) where the questions are made and the quantitative level where the metrics are defined. In this way, the goal is defined by a set of questions and every question is redefined through a set of metrics.

It is also important to validate the metrics from a formal point of view in order to ensure its usefulness. Several frameworks for measure characterization have been proposed. Some of them ([BRI 96], [WEY 88], [BRI 97]) are based on axiomatic approaches. The goal of these approaches is merely definitional by proposing formally desirable properties for measures for a given software attribute, so axioms must be used as guidelines for the definition of a measure. Others ([ZUS 98}) are based on measurement theory which specifies the general framework in which measures should be defined.

However, into the aspects of software measurement, the research is needed ([NEI 94]), from theoretical but also from a practical point of view ([GLA 96]). So, it is necessary to do experiments to validate the metrics. Empirical validation can be

used to investigate the association between proposed software metrics and other indicators of software quality as maintainability ([HAR 98]). So, the goal of this step is to prove the practical utility of the proposed metrics. There are a lot of ways to made it but basically we can divide the empirical validation in two: experimentation and case studies. The experimentation is usually made using controlled experiments and the case studies usually work with real data. Both of them are necessary, the controlled experiments for having a first approach and the case studies for making the results stronger. In both cases, the results are analyzed using either statistics tests or advanced techniques. Also is necessary the replication of the experiment because with the results isolate of an experiment it is difficult to understand how widely applicable the results are and, thus, to assess the true contribution to the field ([BAS 99]).

As we can see in figure 2, the process of defining and validating database metrics is evolutionary and iterative. As a result of the feedback metrics could be redefined of discarded depending of the theoretical, empirical or psychological validations.

In the rest of this paper we will show the different steps of this framework applied to obtain metrics for object-relational databases.

3. Object-relational metrics definition

One of the problems of relational databases is related with representativeness limitations (complex elements which are present in several domain like graphics, geography are hard to represent). On the other hand, object oriented (OO) databases are not enough mature to be accepted. and it is really difficult to convert relational specialists and to convince managers to adopt this new paradigm with all the possible risks involved.

From this point of view, object-relational paradigm proposes a good compromise between both worlds. Object-relational databases combine traditional database characteristics (data model, recovery, security, concurrency, high-level language, etc.) with object-oriented principles (e.g. encapsulation, generalization, aggregation, polymorphism, ...). These products offer the possibility of defining classes or abstract data types, in addition to tables, primary and foreign keys and constraints[1], as do relational databases.

Furthermore, generalization hierarchies can be defined between classes (super and subclasses) and between tables, subtables and supertables. Table attributes can be defined in a simple domain, e.g. CHAR(25), or in a user-defined class as a complex number or image. In Figure 2 we present an example of two object-relational tables definition.

[1] In this first approximation constraints are not considered for measure purposes.

In this example we can notice that part of the data are expressed using relational concepts (tables, primary and foreign keys and references) and the other part using OO concepts (types, and methods). The richness of the resulting model somewhat increases its complexity ([STO 99]). For this reason it is very important to have metrics that allow for the complexity of this kind of databases to be controlled.

```
CREATE TABLE subs(
idsubs INTEGER,
name VARCHAR(20),
subs_add address,
PRIMARY KEY (idsubs));

CREATE TABLE dep(
iddep INTEGER,
name VARCHAR(20),
dep_loc location,
budget DECIMAL (8,2),
PRIMARY KEY (iddep));

CREATE TABLE subs-dep(
idsubs INTEGER,
iddep INTEGER
PRIMARY KEY (idsubs,iddep),
FOREIGN KEY idsubs REFERENCES subs(idsubs)
FOREIGN KEY iddep REFERENCES dep(iddep));

CREATE TABLE employee(
idemp INTEGER,
name VARCHAR2(40),
emp_date date,
emp_loc location,
emp_add address,
manager INTEGER,
dep INTEGER,
PRIMARY KEY (idemp),
FOREIGN KEY manager REFERENCES
employee(idemp),
FOREIGN KEY dep REFERENCES dep(iddep));
```

```
CREATE TYPE address AS(
street CHAR(30),
city CHAR(20),
state CHAR(2),
zip INTEGER);

CREATE TYPE location AS(
building CHAR(4),
office CHAR(4),
table CHAR(4);
```

Figure 2. Example of table definition in SQL:1999

For this kind of database we can propose table related metrics (when we apply the metrics to a table) and schema oriented metrics (when the metrics are applied to the schema).

3.1. Table level metrics

At the table level we propose, being T a table, the metrics DRT(T), RD(T), PCC(T), NIC(T), NSC(T) and TS defined as follows:

DRT(T) metric. Depth of Relational Tree of a table T (DRT(T)) is defined as the longest referential path between tables, from the table T to any other table in the schema database

RD(T) metric. Referential Degree of a table T (RD(T)) is defined as the number of foreign keys in the table T.

PCC(T) metric. Percentage of complex columns of a table T.

NIC(T) metric. Number of involved classes. This measures the number of all classes that compose the types of the complex columns of T using the generalization and the aggregation relationships.

NSC(T) metric. Number of shared classes. This measures the number of involved classes for T that are used by other tables.

TS metric. The table size metric is defined as the sum of the total size of the simple columns (TSSC) and the total size of the complex columns (TSCC), each of these complex columns can be a class or an user defined type UDT) in the table:

$$TS_i = TSSC + TSCC$$

We consider that all simple columns have a size equal to one, then the TSSC metric is equal to the number of simple attributes in the table (NSA).

$$TSSC = NSA$$

And the TSCC is defined as the sum of each complex column size (CCS):

$TSCC = \sum_{i=1}^{NCC} CCS_i$ being NCC the number of complex columns in the table.

The value for CCS is obtained with:

$$CCS = \frac{SHC}{NCU}$$

Being SHC the size of the hierarchy above which the column is defined and NCU is the number of columns defined above this hierarchy. This expression is due to the fact that the understandablility is less if more than one column is defined above the same class. If the number of columns that are defined above a class is greater than one, the complexity of this class decreases (respect to each column, but not for the total columns) and this fact must be appointed when we calculate the complexity of a class.

The SHC may be defined as the sum of each class size in the hierarchy (SC):

$SHC = \sum_{i=1}^{NCH} SC_i$ being NCH the number of classes in the hierarchy.

The size of a class is defined as:

$$SC = \frac{SAC + SMC}{NHC}$$

Being SAC the sum of the size attributes of the class, SMC the size methods of the class and NHC the number of hierarchies to which the class pertain.

The attributes of a class may also be simple or complex (which can be a class or an UDT), then the SAC is defined as the sum of the simple attributes size (SAS, that have size equal to one then the metric corresponds with the number of simple attributes) and the complex attributes size (CAS) in the class.

$$SAC = SAS + CAS$$

And the SMC is calculated with the version of the cyclomatic complexity of McCabe given by ([LI 93]):

$SMC = \sum_{i=1}^{NMC} V_i(G)$ being NMC the number of methods in the class

3.2. Schema level metrics

At the schema level, we can apply the next metrics:

DRT metric. Depth of referential Tree, defined of the longest referential path between tables in the database schema.

RD metric. Referential Degree is defined as the number of foreign keys in the schema database.

PCC metric. Percentage of complex columns in the schema database.

NIC metric. Number of involved classes, number of all classes that composes the types of the complex columns, using the generalization and aggregation relationships, of all tables in the schema.

NSC metric. Number of shared classes, number of shared classes by tables of the schema.

SS metric. Size of a Schema defined as the sum of the tables size (TS) in the schema:

$SS = \sum_{i=1}^{NT} TS_i$ being NT the number of tables in the schema.

3.3. Example

We present the values for the different metrics for the example presented in Figure 2. Let us assume that the date type has a size equal to one.

We can calculate the values for the address and location classes as:

$$CCS_{address} = \frac{4}{2} = 2$$

$$CCS_{location} = \frac{3}{2} = 1.5$$

And with these values we can obtain the values shown in table 1 for each column size of each table:

	COLUMN NAME	COLUMN TYPE	COLUMN SIZE
SUBS	idsubs	Simple	1
	name	Simple	1
	subs_add	Complex	2
DEP	iddep	Simple	1
	name	Simple	1
	dep_loc	Complex	1.5
	budget	Simple	1
SUBS_DEP	idsubs	Simple	1
	iddep	Simple	1
EMPLOYEE	idemp	Simple	1
	name	Simple	1
	emp_date	Simple	1
	emp_loc	Complex	1.5
	emp_add	Complex	2
	manager	Simple	1
	dep	Simple	1

Table 1. Size for each column

With these data, we obtain the following values for the table size metric:

$$TS_{succ} = 2 + 2 = 4$$

$$TS_{dep} = 3 + 1.5 = 4.5$$

$$TS_{succ-dep} = 2$$

$$TS_{employee} = 5 + 3.5 = 8.5$$

$$SS = 4 + 4.5 + 2 + 8.5 = 19$$

The other metrics for the tables are summarized in Table 2.

	SUBS	DEP	SUBS_DEP	EMPLOYEE
TS	4	4.5	2	8.5
RD	0	0	2	2
DRT	0	0	1	2
PCC	33%	25%	0%	28.57%
NIC	1	1	0	2
NSC	1	1	0	2

Table 2. Metric values for the example of Figure 2

4. Object-relational metrics formal verification

As we have said previously, it is important to validate the metrics from a formal point of view in order to ensure its usefulness and there are two main tendencies for making it: axiomatic approaches (the goal of these approaches is merely definitional by proposing formally desirable properties for measures) and the formal frameworks based on measurement theory which specifies the general framework in which measures should be defined.

The strength of measurement theory is the formulation of empirical conditions from which we can derive hypothesis of reality. Measurement theory gives clear definitions of terminology, a sound basis of software measures, criteria for experimentation, conditions for validation of software measures, foundations of prediction models, empirical properties of software measures, and criteria for measurement scales.

In this section we present the formal verification of the TS, the RD and the DRT metrics made in the formal framework proposed by Zuse ([ZUS 98]) and based on the measurement theory. All the information related with this framework can be found in ([ZUS 98]).

For our purposes, the Empirical Relational System could be defined as:

$$\mathbf{R} = (R, \bullet >=, o)$$

Where R is a non-empty set of relations (tables), • >= is the empirical relation "more or equal complex than" on R and o is a closed binary (concatenation) operation on R. In our case we will choose natural join as the concatenation operation. Natural join is defined generally as ([ELM 99]) :

$$Q \leftarrow R \ {}_{(<list1>^*,<list2>)} \ S$$

Where <list1> specifies a list of i attributes of R and <list2> is a list of i attributes of S. These lists are used in order to make the comparison equality conditions between pairs of attributes. These conditions are afterwards related with the AND operator. Only the list corresponding to the R relation is preserved in Q.

Depending on the characteristics of the combined tables, natural join can derive in Cartesian product. Furthermore, it is possible to make the natural join through foreign key-primary key or between any columns of two tables defined over the same domain.

All these characteristics of the natural join will be useful in order to design the combination rule of the metrics.

4.1. TS metric formal verification

The TS (Table Size) measure is a mapping: TS: R -> $\Re$ such that the following holds for all relations Ri and Rj $\in$ R: Ri • >= Rj $\Leftrightarrow$ TS(Ri) >= TS(Rj).

In order to obtain the combination rule for TS when we combine tables by natural join we may think that if the combined tables have not common columns, the attributes of the obtained table is the union of the attributes of the two table combined and the size will be the sum of each attribute size, but if the tables have any common column, the size of the obtained tables will be the sum of each size attribute minus the size of the duplicate simple column (by definition we must subtract only the simple column size because on the size of a complex column is reflected if the hierarchy, among to which the column is defined, is shared by more than one column).

So, we can define the combination rule for TS as:

$$TS (Ri \ o \ Rj) = TS (Ri) + TS(Rj) – SASC(Ri \cup Rj)$$

Where SASC(Ri$\cup$Rj) is the size of the common simple attributes of Ri and Rj.

We can rename this last expression as v (being v a variable) and define the combination rule for TS as:

$$TS(RioRj) = TS(Ri) + TS(Rj) - v$$

TS fulfils the first axiom of weak order, because if we have two relations R1 and R2, it is obvious that TS(R1) >= TS(R2) or TS(R2) >= TS(R1) (completeness) and let R1, R2 and R3 three relations, transitivity is always fulfilled: TS(R1) >= TS(R2) and TS(R2) >= TS(R3), then TS(R1) >= TS(R3).

TS does not fulfil positivity, because if we combine a relation R1 with itself without cycles: TS(R1 o R1) is not greater than TS(R1). But it fulfils weak positivity , because it is always true that: TS(R1 o R2) >= TS(R1) for all R1, R2 $\in$ R.

TS fulfils associativity and commutativity (axioms 3 and 4) , because the natural join operation is associative and commutative.

TS does not fulfil weak monotonicity because if we have two tables (R1 and R2) with the same number of attributes with the same size and we combine every one of these tables with a third table (R3) that has one common attribute with the first table (R1) and none common attribute with the second table (R2), the table that results of R1oR3 will have less size than the table that results of R2oR3.

Due to the fact that the number of attributes vary when we combine one table with itself, we can conclude that the metric is not idempotent and is necessary to prove the Archimedean axiom.

In order to prove that the Archimedean axiom is not accomplished is important to observe that when two tables are combined by natural join successively, the number of attributes vary and also the size. Moreover the tables obtained in successive concatenations will be the same than these obtained in the first concatenation. Then, if we have four tables R1, R2, R3 and R4, and R3 has three attributes and a size equal to three, R4 has two attributes and a size equal to two, R1 has three attributes (one of them common with R3) and a size equal to three and R2 has four attributes and a size equal to four, and we make the concatenation R3o R1 (that is equal to the concatenation R3oR21oR1o...), obtaining a table with five attributes and a size equal to 5, and we make the concatenation R4 oR2 (that is equal to the concatenation R4oR2oR2o...), obtaining a table with six attributes and a size equal to six, the Archimedean axiom is not accomplished.

So, measure TS does not assume an extensive structure.

Would TS verify the independence conditions?. As we have seen the metric do not accomplish the axiom of weak monotonicity, then it can accomplish neither independence conditions.

In fact, this type of combination rules do not assume the independence conditions. The part –v rejects the condition C1 that implies the rejection of the axiom of weak monotonicity, monotonicity and extensive structure.

Then we must study if TS fulfils some of the modified relations of belief.

MRB1 is fulfilled, because giving two relations R1 and R2 $\in$ $\Im$ ($\Im$ is the set of all the possible relations made with the attributes of the relational schema) TS(R1) >= TS(R2) or TS(R2) >= TS(R1).

MRB2 is also fulfilled (transitivity of natural join). For MRB3 we will consider that a relation R1 $\supseteq$ R2 if all the attributes of R2 are present in R1. In this case it is evident that TS(R1) >= TS(R2), and MRB3 is fulfilled.

MRB4 is fulfilled because if a relation R1 $\supset$ R2 then TS(R1) > TS(R2) and TS(R1 U R3) > TS(R2 U R3), being R1 $\cap$ R3 = ϕ. If the relations R1 and R3 do not have any attribute in common, adding the attributes of R3 to both R1 and R2, (if R1 subsumes R2), then the number of attributes of R1 and R3 is greater than the number of attributes of R2 and R3, and also their size.

MRB5 is fulfilled because a relation must always have zero or more attributes, then the size must be equal or greater than zero.

In summary, we can characterize TS as a measure above the level of the ordinal scale, assuming the modified relation of belief.

The validation of the other metrics can be made following the same steps: defining the combination rule for the metric and proving the different properties in order to obtain the appropriate scale for the metric.

5. Object-relational metrics empirical validation

In this section, we present the experiment developed in order to evaluate whether the proposed measures can be used as indicators for estimating the maintainability of an OR database.

5.1. Data Collection

Five object-relational databases were used in this experiment with the average of 10 relations per database (ranging from 6 to 13). These databases were originally relational ones. For the purpose of the experiment, they were redesigned as OR databases. A brief description of these databases is given in table 3.

Database	Number of tables	Average attributes/table	Average complex attributes/table
Airlines	6	4,16	1,83
Animals	10	2,7	0,6
Library	12	2,91	0,75
Movies	9	4,33	0,88
Treebase	13	3,46	0,86

Table 3. Databases used in the experiment

Five people participated in the experiment the first time we made it (Canadian experiment): one researcher, two research assistants and two graduate students. All of them are experienced in both relational databases and object-oriented programming. On the first experiment, one person did not complete the experiment, and we had to discard his partial results. So, in the replication (Spanish experiment) only four people made the experiment. Also all of them are experienced in both relational databases and object-oriented programming

The people were given a form, which include for each table, a triplet of values to compute using the corresponding schema. These values are those of three measures TS, DRT and RD. Our idea is that to compute these measures, we need to understand the subschema (objects and relations) defined by the concerned table. A table (and then the corresponding subschema) is easy to understand if (almost) all the people find the right values of hte metrics in a limited time (2 minutes per table). We wanted to measure understandability, we decided to give our people a limited time to finish the tests they had carry out and then, use all the tests that had been answered in the given time and in a correct way (following all the indications given for the development of the experiment). So, our study would focus on the amount of metrics correctly calculated. Formally, a value 1 is assigned to the maintainability of a table if at least 10 of 12 measures are computed correctly in the specified time (4 people and 3 measures). A value 0 is assigned otherwise. The tables are given to the people in a random order and not by database.

5.2. Validation Technique

To analyze the usefulness of the metrics proposed, we used two techniques: C4.5 ([QUI 93]), a machine learning algorithms and RoC ([RAM 99]), a robust Bayesian classifier.

C4.5 belongs to the divide and conquer algorithms family. In this family, the induced knowledge is generally represented by a decision tree. The principle of this approach could be summarized by this algorithm:

If the examples are all of the same class

Then - create a leaf labelled by the class name;

Else - select a test based on one attribute;

- divide the training set into subsets, each associated to one of the possible values of the tested attribute;

- apply the same procedure to each subset;

Endif.

The key step of the algorithm above is the selection of the "best" attribute to obtain compact trees with high predictive accuracy. Information theory-based heuristics have provided effective guidance for this division process. C4.5 induces Classification Models, also called Decision Trees, from data. It works with a set of examples where each example has the same structure, consisting of a number of attribute/value pairs. One of these attributes represents the class of the example. The problem is to determine a decision tree that correctly predicts the value of the class attribute (i.e., the dependent variable), based on answers to questions about the non-class attributes (i.e., the independent variables).

In our study, the C4.5 algorithm partitions continuous attributes (the database metrics), finding the best threshold among the set of training cases to classify them on the dependent variable (i.e. understandability of the database schemes).

RoC is a Bayesian classifier. It is trained by estimating the conditional probability distributions of each attribute, given the class label. The classification of a case, represented by a set of values for each attribute, is accomplished by computing the posterior probability of each class label, given the attributes values, by using Bayes' theorem. The case is then assigned to the class with the highest posterior probability.

The simplifying assumptions underpinning the Bayesian classifier are that the classes are mutually exclusive and exhaustive and that the attributes are conditionally independent once the class is known. RoC extends the capabilities of the Bayesian classifier to situations in which the database reports some entries as unknown. It can then train a Bayesian classifier from an incomplete database.

One of the great advantages of C4.5 comparing to RoC is that, it produces a set of rules, directly understandable by software manager and engineers.

5.3. Results

As specified in validation technique section, we applied RoC and C4.5 to evaluate the usefulness of the OR metrics in estimating the maintainability of the tables in an OR schema.

5.3.1. RoC technique

Using the cross-validation technique, the algorithm RoC was applied 10 times on the 50 examples obtained from the 50 tables of the five schematas (500 cases). 369 cases were correctly estimated for the Canadian experiment (accuracy 73.8%) and 407 cases for the Spanish one (accuracy 81.4%) and all the other cases in both experiments were missclassified. Contrary to C4.5, RoC does not propose a default classification rule which guaranteed a coverage of all the proposed cases. However, in this experiment, it succeeded to cover all the 500 cases (coverage of 100%).

These results are summarized in the table 4.

	Spain	**Canada**
Correct:	407	369
Incorrect:	93	131
Not classified:	0	0
Accuracy:	81.4 %	73.8 %
Coverage:	100.0 %	100.0 %

Table 4. RoC quantitative results with data from Spain and from Canada

RoC produces the model presented in figure 5 with the Canadian data. From this model, it is hard to say which metric is more relevant than another in an absolute manner. However, we can notice that when TS is smaller, the probability that the table is understandable is higher (for example 55% for TS <= 3). This probability decrease when the table size increase (9.5% for TS >10). Inversely, the same probability increase in estimating the table that are not understandable (varying from 13.6% for TS <= 3 to 33.6% for TS >10).

For DRT and RD, it is hard to draw a conclusion since no uniform variation is shown. This can be explained by the fact that for the sample used in this experiment, the values of DRT and RD are in defined in a narrow range ([0, 3] and [0, 5]).

RoC produces the model presented in figure 6 with the Spanish data. The conclusions from this second model are the same as the first one because the models are very similar.

Canadian Model

TS

	(1 . 3)	(3 . 5)	(5 . 10)	(10 . 17.5)
0	0.136	0.193	0.336	0.336
1	0.543	0.233	0.129	0.095

DRT

	0	1	2	3
0	0.336	0.221	0.193	0.250
1	0.336	0.371	0.233	0.060

RD

	0	1	2	3	4	5
0	0.319	0.319	0.148	0.09	0.062	0.062
1	0.316	0.247	0.316	0.040	0.040	0.040

PCC

	(0 . 25)	(25 . 80)
0	0.471	0.529
1	0.603	0.397

NIC

	0	1	2	3	4	5	6
0	0.257	0.114	0.229	0.143	0.143	0.057	0.057
1	0.517	0.241	0.069	0.034	0.069	0.034	0.034

NSC

	0	1	2	3	4	5
0	0.462	0.233	0.090	0.090	0.062	0.062
1	0.799	0.040	0.040	0.040	0.040	0.040

Figure 5. The model generated by RoC with data from Canada

Spanish Model

TS

	(1 . 3)	(3 . 5)	(5 . 10)	(10 . 17.5)
0	0.095	0.129	0.405	0.371
1	0.507	0.279	0.107	0.107

DRT

	0	1	2	3
0	0.371	0.198	0.164	0.267
1	0.307	0.364	0.250	0.079

RD

	0	1	2	3	4	5
0	0.351	0.316	0.075	0.109	0.075	0.075
1	0.290	0.262	0.348	0.033	0.033	0.033

Figure 6. The model generated by RoC with data from Spain

5.3.2. C4.5 technique

The results obtained for the Canadian experiment are shown in table 5. The model of C4.5 was very accurate in estimating the maintainability of a table, 94% and resent a high level of completeness (up to 100% for not understandable tables) and correctness (up to 100% for understandable tables).

		Predicted maintainability		
		0	1	Completeness
Real	0	28	0	100%
Maintainability	1	3	19	86.36%
	Correctness	90.32%	100%	

Accuracy = 94%

Table 5. C4.5 quantitative results from the Canadian experiment

And the rules obtained with C4.5 are:

Rule 1:

TS <= 9 ∧ DRT = 0 ∧ NSC = 0 -> class 1 [84.1%]

Rule 2:

TS <= 3 ∧ RD > 1 -> class 1 [82.0%]

Rule 7:

TS <= 9 ∧ DRT <= 2 ∧ NIC > 0 ∧ NSC = 0 -> class 1 [82.0%]

Rule 5:

TS > 9 -> class 0 [82.2%]

Rule 6:

DRT > 2 -> class 0 [82.0%]

Default class: 0

Figure 7. C4.5 estimation model from the Canadian data

TS seems to be an important indicator for the maintainability of the tables. Rules 1, 2 and 7, which determine if a table is maintainable, have all as part of the conditions that TS must be small. Inversely, in rule 5, it is stated a large size is sufficient to declare the table as not understandable. A small DRT is also required for rules 1 and 7 as partial condition to classify the table as understandable. In the same time, a high value of DRT means that the table is hard to understand (rule 6). RD does not represent an interesting indicator.

The results obtained for the Spanish experiments are shown in table 6. In this case the accuracy in estimating the maintainability was 94% and the levels of completeness and correctness were smaller than the Canadian experiment but were also very high.

		Predicted maintainabilty		
		0	1	Completeness
Real	0	21	1	95.45%
Maintainabilty	1	3	25	89.29%
	Correctness	87.5%	96.15%	

Accuracy = 92%

Table 6. C4.5 quantitative results from the Spanish experiment

And the rules obtained with C4.5 are:

Rule 1:

TS <= 5 ∧ DRT <= 2 -> class 1 [89.4%]

Rule 3:

TS > 5 ∧ PCC <= 66 -> class 0 [82.3%]

Rule 2:

DRT > 2 -> class 0 [66.2%]

Default class: 1

Figure 8. C4.5 estimation model from the Spanish data

The model rules is smaller than the one of the first experiment but it confirms that (at least for the studied sample) TS and DRT are good indicators and not RD.

Both experiments and both techniques find out that the table size metric (TS) is a good indicator for the maintainability of a table. The depth of the referential tree metric (DRT) is also presented as an indicator by C4.5 on both experiments and the referential degree metric (RD) does not seem to have a real impact on the maintainability of a table.

6. Conclusions and future work

It is important that software products, and obviously databases, are evaluated for all relevant quality characteristics, using validated or widely accepted metrics. However, more research is needed into the aspects of software measurement ([NEI 94]), both from a theoretical and from a practical points of view ([GLA 96]). We think it is very interesting to dispose on metrics for object-relational databases. These metrics can be used to flag outlying schemata for special attention, a strong requirement for low testing and maintenance costs would argue for justify extra managerial attention to a quite significant fraction of the object-relational database schemata.

We have put forward different measures (for internal attributes) in order to measure the complexity that affect the maintainability (an external attribute) of the relational database schemata and consequently control its quality. These metrics were developed and characterized in accordance with a set of sound measurement principles, applying the formal framework proposed by Zuse ([ZUS 98]), in order to obtain the scales to which the metrics pertain.

We have done some experiments to validate the proposed metrics, but more others are being developed at this moment. However the controlled experiments have problems (such as the large number of variables that cause differences, or the fact that these experiments deal with low level issues, microcosms of reality and small sets of variables) and limits (e.g. they do not scale up, are done in a class in training situations, are made in vitro and face a variety of threats of validity). Therefore, it is convenient to run multiple studies, mixing controlled experiments and case studies ([BAS 99]). For these reasons, a deeper empirical evaluation is under way in collaboration with industrial and public organizations in "real" situations.

7. References

[AND 83] Anderson, J.R. (1983). *The Architecture of Cognition*. Cambridge, MA: Harvard Universitiy Press.

[BAS 99] Basili, V.R., Shull, F. and Lanubille, F. (1999), Building Knowledge through families of experiments, *IEEE Transactions on Software Engineering*, July/August, Nr. 4. Pp. 456-473

[BRI 96] Briand, L.C., Morasca, S. and Basili, V. (1996). Property-based software engineering measurement. *IEEE Transactions on Software Engineering*, 22(1): 68-85.

[BRI 97] Briand L.C. and Morasca S. (1997). Towards a Theoretical Framework for Measuring Software Attributes. *Proceeding of the Fourth International, Software Metrics Symposium,* 119-126.

[ELM 99] Elmasri, R. and Navathe, S. (1999). *Fundamentals of Database Systems*. Third edition. Addison-Wesley. Massachussets

[FRA 92] Frazer, A. (1992). Reverse engineering-hype, hope or here?. In: P.A.V. Hall, *Software Reuse and Reverse Engineering in Practice*. Chapman & Hall.

[GLA 96] Glass, R. (1996). The Relationship Between Theory and Practice in Software Engineering. *IEEE Software*, November, 39 (11), 11-13.

[HAR 98] Harrison, R., Counsell, S. And Nithi, R. (1998), Coupling metrics for Object-Oriented Design, *5th. International Symposium on Software Metrics*, IEEE Computer Society. Bethesda, Maryland, 20-21 November.

[HEN 96] Henderson-Sellers, B. (1996). *Object-oriented Metrics - Measures of complexity*. Prentice-Hall, Upper Saddle River, New Jersey.

[ISO 94] ISO, (1994). Software Product Evaluation-Quality Characteristics and Guidelines for their Use. *ISO/IEC Standard 9126*, Geneva.

[LI 87] Li, H.F. and Chen, W.K. An empirical study of software metrics. *IEEE Trans. on Software Engineering*, (1987), 13 (6): 679-708.

[LI 93] Li, W. and Henry, S. (1993). Object-Oriented metrics that predicts maintainability. *J. Sys. Software*, 23, pp. 111-122.

[MCA 76] McCabe, T.J. (1976). A complexity measure. *IEEE Trans. Software Engineering* 2(5), 308-320.

[MCL 92] McClure, C. (1992) *The Three R's of Software Automation: Re-engineering, Repository, Reusability*. Englewood Cliffs: Prentice-Hall.

[NEI 94] Neil, M. (1994) Measurement as an Alternative to Bureaucracy for the Achievement of Software Quality. *Software Quality Journal* 3 (2), 65-78.

[PIG 97] Pigoski, T.M. (1997). Practical Software Maintenance. Wiley Computer Publishing. New York, USA.

[QUI 93] Quinlan, J.R., (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers.

[RAM 99] Ramoni, M. and Sebastiani, P. (1999)Bayesian methods for intelligent data analysis. In M. Berthold and D.J. Hand, editors, *An Introduction to Intelligent Data Analysis*, New York,. Springer.

[SIA 99] Siau, K. (1999). Information Modeling and Method Engineering: A Psychological Perspective. *Journal of Database Management* 10 (4), 44-50.

[SNE 98] Sneed, H.M. and Foshag, O. Measuring Legacy Database Structures. *Proc of The European Software Measurement Conference FESMA 98*, (Antwerp, May 6-8, 1998). Coombes, Van Huysduynen and Peeters (eds.), 199-211.

[STO 99] Stonebraker, M. and Brown, P. *Object-Relational DBMSs tracking the next great wave*, (California, 1999), Morgan Kauffman Publishers.

[WEY 88] Weyuker, E.J. (1988). Evaluating software complexity measures. IEEE Transactions on Software Engineering Vol.14(9). pp. 1357-1365.

[ZUS 98] Zuse, H. (1998). *A Framework of Software Measurement*. Berlin, Walter de Gruyter.