Heinrich C. Mayr
Jiri Lazansky
Gerald Quirchmayr
Pavel Vogel (Eds.)

LNCS 2113

Database and Expert Systems Applications

12th International Conference, DEXA 2001
Munich, Germany, September 2001
Proceedings

DEXA 2001

Springer

Heinrich C. Mayr Jiri Lazansky
Gerald Quirchmayr Pavel Vogel (Eds.)

Database and Expert Systems Applications

12th International Conference, DEXA 2001
Munich, Germany, September 3-5, 2001
Proceedings

Springer

Preface

DEXA 2001, the 12th International Conference on Database and Expert Systems Applications was held on September 3–5, 2001, at the Technical University of Munich, Germany. The rapidly growing spectrum of database applications has led to the establishment of more specialized discussion platforms (DaWaK conference, EC-Web conference, and DEXA workshop), which were all held in parallel with the DEXA conference in Munich.

In your hands are the results of much effort, beginning with the preparation of the submitted papers. The papers then passed through the reviewing process, and the accepted papers were revised to final versions by their authors and arranged with the conference program. All this culminated in the conference itself. A total of 175 papers were submitted to this conference, and I would like to thank all the authors. They are the real base of the conference. The program committee and the supporting reviewers produced altogether 497 referee reports, on average of 2.84 reports per paper, and selected 93 papers for presentation.

Comparing the weight or more precisely the number of papers devoted to particular topics at several recent DEXA conferences, an increase can be recognized in the areas of XMS databases, active databases, and multi- and hypermedia efforts. The space devoted to the more classical topics such as information retrieval, distribution and Web aspects, and transaction, indexing and query aspects has remained more or less unchanged. Some decrease is visible for object orientation.

At this point we would like to say many thanks to all the institutions which actively supported this conference and made it possible. These are:
- The Technical University of Munich
- FAW
- DEXA Association
- Austrian Computer Society

A conference like DEXA would not be possible without the enthusiastic efforts of several people in the background. First we would like to thank the whole program committee for the thorough referee process. Many thanks also to Maria Schweikert (Technical University of Vienna) and Monika Neubauer and Gabriela Wagner (FAW, University of Linz).

July 2001

Jiri Lanzanski
Heinrich C. Mayr
Gerald Quirchmayr
Pavel Vogel

Program Committee

General Chairperson:
Heinrich C. Mayr, University of Klagenfurt, Austria

Conference Program Chairpersons:
Jiri Lazansky, Czech Technical University, Czech Republic
Gerald Quirchmayr, University of Vienna, Austria
Pavel Vogel, Technical University of Munich, Germany

Workshop Chairpersons:
A Min Tjoa, Technical University of Vienna, Austria
Roland R. Wagner, FAW, University of Linz, Austria

Publication Chairperson:
Vladimir Marik, Czech Technical University, Czech Republic

Program Committee Members:
Michel Adiba, IMAG - Laboratoire LSR, France
Hamideh Afsarmanesh, University of Amsterdam, The Netherlands
Jens Albrecht, Oracle GmbH, Germany
Ala Al-Zobaidie, University of Greenwich, UK
Bernd Amann, CNAM, France
Frederic Andres, NACSIS, Japan
Kurt Bauknecht, University of Zurich, Switzerland
Trevor Bench-Capon, University of Liverpool, United Kingdom
Alfs Berztiss, University of Pittsburgh, USA
Jon Bing, University of Oslo, Norway
Omran Bukhres, Purdue University, USA
Luis Camarinah-Matos, New University of Lisbon, Portugal
Antonio Cammelli, IDG-CNR, Italy
Wojciech Cellary, University of Economics at Poznan, Poland
Stavros Christodoulakis, Technical University of Crete, Greece
Panos Chrysanthis, Univ. of Pittsburgh & Carnegie Mellon Univ., USA
Paolo Ciaccia, University of Bologna, Italy
Christine Collet, LSR-IMAG, France
Carlo Combi, University of Udine, Italy
William Bruce Croft, University of Massachusetts, USA
John Debenham, University of Technology, Sydney, Australia
Misbah Deen, University of Keele, United Kingdom
Nina Edelweiss, University of Rio Grande do Sul, Brazil
Johann Eder, University of Klagenfurt, Austria
Thomas, Eiter , Technical University of Vienna, Austria
Gregor Engels, University of Paderborn, Germany
Peter Fankhauser, GMD-IPSI, Germany
Eduardo Fernandez, Florida Atlantic University, USA

Simon Field, IBM Research Division, Switzerland
Peter Funk, Mälardalen University, Sweden
Antonio L. Furtado, University of Rio de Janeiro, Brazil
Georges Gardarin, University of Versailles, France
Parke Godfrey, York University, Canada
Georg Gottlob, Technical University of Vienna, Austria
Paul Grefen, University of Twente, The Netherlands
Abdelkader Hameurlain, Université Paul Sabatier, France
Igor T. Hawryszkiewycz, University of Technology, Sydney, Australia
Mohamed Ibrahim, University of Greenwich, UK
Yahiko Kambayashi, Kyoto University, Japan
Magdi N. Kamel, Naval Postgraduate School, USA
Nabil Kamel, American University in Cairo, Egypt
Gerti Kappel, University of Linz, Austria
Kamalar Karlapalem, University of Science and Technology, Hong Kong
Randi Karlsen, University of Tromsö, Norway
Rudolf Keller, University of Montreal, Canada
Myoung Ho Kim, KAIST, Korea
Masaru Kitsuregawa, Tokyo University, Japan
Gary J. Koehler, University of Florida, USA
Donald Kossmann, Technical University of Munich, Germany
Jaques Kouloumdjian, INSA, France
Petr Kroha, Technical University Chemnitz-Zwickau, Germany
Josef Küng, University of Linz, Austria
Michel Leonard, University of Geneve, Switzerland
Tok Wang Ling, National University of Singapore, Singapore
Mengchi Liu, University of Regina, Canada
Fred Lochovsky, Hong Kong Univ. of Science and Technology, Hong Kong
Peri Loucopoulos, UMIST, United Kingdom
Sanjai Kumar Madria, University of Missouri-Rolla, USA
Akifumi Makinouchi, Kyushu University, Japan
Vladimir Marik, Czech Technical University, Czech Republic
Simone Marinai, University of Florence, Italy
Subhasish Mazumdar, New Mexico Tech, USA
Robert Meersman, Free University Brussels, Belgium
Elisabeth Metais, University of Versailles, France
Mukesh Mohania, Western Michigan University, USA
Sophie Monties, EPFL, Switzerland
Tadeusz Morzy, Poznan University of Technology, Poland
Günter Müller, University of Freiburg, Germany
Erich J. Neuhold, GMD-IPSI, Germany
Gultekin Ozsoyoglu, University Case Western Research, USA
Georgios Pangalos, University of Thessaloniki, Greece
Stott Parker, University of Los Angeles (UCLA), USA
Oscar Pastor, Technical University of Valencia, Spain
Marco Patella, University of Bologna, Italy
Barbara Pernici, Politecnico di Milano, Italy

Günter Pernul, University of Essen, Germany
Fausto Rabitti, CNUCE-CNR, Italy
Isidro Ramos, Technical University of Valencia, Spain
Haralad Reiterer, University of Konstanz, Germany
Norman Revell, Middlesex University, UK
Sally Rice, Unversity of South Australia, Australia
John Roddick, Flinders University of South Australia, Australia
Colette Rolland, University Paris I, Sorbonne, France
Elke Rundensteiner, Worcester Polytechnic Institute, USA
Domenico Sacca, University of Calabria, Italy
Marinette Savonnet, University of Bourgogne, France
Erich Schweighofer, University of Vienna, Austria
Timos Sellis, National Technical University of Athens, Greece
Michael H. Smith, University of California, USA
Giovanni Soda, University of Florence, Italy
Harald Sonnberger, EUROSTAT, Luxembourg
Günther Specht, Technical University of Ilmenau, Germany
Uma Srinivasan, CIRO, Australia
Bala Srinivasan, Monash University, Australia
Olga Stepankova, Czech Technical University, Czech Republic
Zbigniew Struzik, CWI, Amsterdam, The Netherlands
Makoto Takizawa, Tokyo Denki University, Japan
Katsumi Tanaka, Kobe University, Japan
Zahir Tari, University of Melbourne, Australia
Stephanie Teufel, University of Fribourg, Germany
Jukka Teuhola, University of Turku, Finland
Bernd Thalheim, Technical University of Cottbus, Germany
Jean Marc Thevenin, University of Toulouse, France
Helmut Thoma, IBM Global Services Basel, Switzerland
A Min Tjoa, Technical University of Vienna, Austria
Roland Traunmüller, University of Linz, Austria
Aphrodite Tsalgatidou, University of Athens, Greece
Susan Urban, Arizona State University, USA
Krishnamurthy Vidyasankar, Memorial University of Newfoundland, Canada
Roland R. Wagner, University of Linz, Austria
Michael Wing, Middlesex University, UK
Werner Winiwarter, Software Competence Center Hagenberg, Austria
Gian Piero Zarri, CNRS, France
Arkady Zaslavsky, Monash University, Australia

Table of Contents

User Interfaces

Advanced Databases II

Information Retrieval Aspects II

Multimedia Databases

Workflow Aspects

Advanced Databases III

Information Retrieval Aspects III

Active Databases

Spatial Databases

Advanced Databases IV

Distributed Databases

Web Aspects I

Knowledge Aspects I

XML

Datawarehouses

Web Aspects II

Knowledge Aspects II

Hypermedia

Index Aspects

Object-Oriented Databases I

Transaction Aspects I

Query Aspects I

Object-Oriented Databases II

Transaction Aspects II

Query Aspects II

DEXA Position Paper

[18] Shenoy S.T, Ozsoyoglu Z. M., Design and implementation of semantic query optimiser. IEEE Transactions on Knowledge and Data Engineering, 1(3) 1989, 344-361.
[19] Siegel, M., Sciore E. and Salveter S., A method for automatic rule derivation to support semantic query optimisation, ACM Transactions on Database Systems, Vol. 17, No. 4, 563-600, 1992.
[20] Yu C. and Sun W., Automatic knowledge acquisition and maintenance for semantic query optimisation, IEEE Transactions on Knowledge and Data Engineering, 1(3) 362-375, 1989.
[21] Robinson J. and Lowden B.G.T., *Extending the Re-use of Query Results at Remote Client Sites*, Proc. 11th Intl. Conf. on Database and Expert Systems Applications, DEXA 2000, pages 536-547. Springer (LNCS 1873).
[22] Dar, S., Franklin, M. J., Jonsson, B. T., Srivastava, D., Tan, M.: Semantic Data Caching and Replacement, Proc. 22nd VLDB Conference (1996) 330-341.
[23] Keller, A. M., Basu, J.: A Predicate-based Caching Scheme for Client-Server Database Architectures. VLDB Journal 5(1) 1996, 35-47.
[24] Robinson J. and Lowden B.G.T., *Semantic Query Optimisation and Rule Graphs*. Proc. KRDB'98, 5th Intl. Workshop on Knowledge Representation meets DataBases, 1998, pp 14.1 - 14.10. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-10/
[25] Geist, A, Beguelin A, et. al., PVM: Parallel Virtual Machine, A Users' Guide and Tutorial for Networked Parallel Computing, MIT Press, 1994.

Estimating Object-Relational Database Understandability Using Structural Metrics

Coral Calero[1], Houari A. Sahraoui[2], Mario Piattini[1], Hakim Lounis[3]

[1] Dep. Informática Universidad de Castilla-La Mancha Ronda Calatrava, 5
13071 Ciudad Real Spain
{ccalero, mpiattin}@inf-cr.uclm.es

[2] Dep. d'Informatique et de Recherche Opérationnelle Université de Montréal
CP 6128 succ. Centre Ville
Montréal QC H3C 3J7 Canada
Sahraouh@iro.umontreal.ca

[3] Département d'informatique Université du Québec a Montréal
CP 8888, succ. Centre-ville
Montréal QC H3C 3P8 Canada
hlounis@uqam.ca

Abstract. New Object-Relational Database Management Systems (ORDBMSs) are replacing existing relational ones. In spite of the high expressiveness, application systems built upon ORDBMS are more complex and difficult to maintain due to the mixing of two paradigms, the relational and the object-oriented. This paper describes a suite of metrics for measuring different aspects of an object-relational database. An empirical validation of the usefulness of the proposed metrics in estimating the understandability of an object-relational schema is given. The analysis procedure comprises the use of two techniques: C4.5, a machine learning algorithm, and RoC, a robust Bayesian classifier. The results demonstrate that a subset of the proposed measures is relevant for the estimation of the understandability.

1 Introduction

Software maintenance is the most expensive stage in the software life cycle. It is one of the greatest problems in the software industry, representing between 67% and 90% of total life cycle costs ([3], [11]). Until now, software maintenance efforts have mainly been centered on program maintenance tasks, as data maintenance was relatively easy in simple files or simple relational tables.

Nowadays, we are witnessing important advances in database technology; a new "generation" of DBMS (Database Management System) is coming out, among which object-relational ones (e.g. Oracle 8, Informix Dynamic Server, DB2) stand out. Object-relational databases will replace relational systems to become the next great wave of databases ([16]). This kind of DBMSs supports a more complex data model having a stronger influence on the overall application maintenance effort.

Therefore, it is very important to have maintainability metrics for this new kind of databases. Metrics for databases have been neglected in the metric community ([15]). In fact, most of the metrics put forward, as the famous McCabe ([9]) cyclomatic number, have been centered on measuring program complexity, quality, maintenance, etc.

Maintainability is achieved by three factors: understandability, modifiability and testability, which in turn are influenced by complexity ([8]). We must be conscious, however, that a general complexity measure is "the impossible holy grail" ([4]). Henderson-Sellers ([6]) distinguishes three types of complexity: computational, psychological and representational, and for psychological complexity three components are considered: problem complexity, human cognitive factors and product complexity. The last one will be our focus.

In this paper we propose some complexity metrics for measuring object-relational databases and we use them to study their impact on the schema understandability. As understandability is one of the components of maintainability, these metrics may be used as partial indicators of the overall IS maintainability.

However, the goal is not only to propose metrics, because when a new measure is proposed, it is natural to ask whether the measure captures the attribute it claims to describe. We want to be sure that the measures we use reflect the behavior of entities in the real world ([5]). This validation must be made according to two perspectives: theoretical and experimental. The formal verification of the metrics presented can be found in [1]. In this paper we describe two experiments carried out with these metrics.

In the next section we summarize the features of object-relational databases. In Section 3, we describe the proposed metrics. Metrics empirical validation is presented in Section 4. Finally, Section 5 summarizes the paper and draws our conclusions.

2 Object-Relational Databases

Relational databases are widely accepted and used by the database community but they present some problems such as the representativeness limitations (complex elements which are present in several domain like graphics, geography are hard to represent). On the other hand, object-oriented databases propose a more powerful model to represent such elements. However, for multiple reasons, adopting this technology is still difficult. Object Oriented (OO) databases are not as mature as the relational ones. Another more practical reason is the difficulty of converting 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 1 an example based on the one presented in [2] is shown.

```
CREATE TABLE house(
idhouse INTEGER,
idagency INTEGER,
price INTEGER,
rooms INTEGER,
size DECIMAL (8,2),
location address,
desc text,
front_view bitmap,
document doc,
seller employee,
PRIMARY KEY (idhouse),
FOREIGN KEY idagency
REFERENCES agency(id));

CREATE TABLE agency(
id INTEGER,
name VARCHAR(20),
location address);
```

```
CREATE TYPE address AS(
street CHAR(30),
city CHAR(20),
state CHAR(2),
zip INTEGER) NOT FINAL;

CREATE TYPE employee
AS(
name CHAR(40),
base_salary DECIMAL(9,2),
bonus DECIMAL(9,2))
INSTANTIABLE NOT FINAL
METHOD salary() RETURNS
DECIMAL(9,2);
CREATE METHOD salary()
FOR employee
BEGIN
...
END;
```

Fig. 1. Example of table definition in SQL:1999

In this example we can notice that part of the data is 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 ([16]). For this reason it is very important to have metrics that allow for the complexity of this kind of databases to be controlled.

3 Working Hypotheses

Our main hypothesis is that the understandability is highly influenced by the complexity of an object-relational database schema.

Moreover, we claim that complexity is itself impacted by the size and the coupling between the elements of the schema (tables and classes). Taking into account this idea and the characteristics of an object-relational schema, we present six metrics which cover these two aspects and which are defined at table level. Three of them have been

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

previously presented and theoretically validated in [1] (TS, DRT and RD). The other three metrics have been defined for the present experiment.

3.1 Description of the Metrics[2]

TS Metric

The table size (TS) measures the size not only in terms of the simple columns (defined using simple domains), but also in terms of complex columns (defined using user-defined classes). Formally it can be defined as the sum of the total size of the simple columns (TSSC) and the total size of the complex columns (TSCC) in the table. TSSC is simply the number of simple columns in the table (considering that each simple domain has a size equal to one). TSCC is defined as the sum of complex columns size (CCS). The size of a complex column is no more than the size of the class hierarchy above the column is defined weighted by the number of complex columns which use the hierarchy. Finally, the size of a class hierarchy is defined as the sum of the size of each class in the hierarchy. For more details about the precise definition of this metric see [1].

RD Metric

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

DRT Metric

Depth of Relational Tree (DRT) is defined as the longest path among the table concerned and the rest of the tables in the schema database, by considering the later as a graph where nodes are tables and arcs are referential integrity relations 1between tables (Foreign key-Primary key link).

PCC Metric

PCC is defined as the percentage of complex columns.

NIC Metric

NIC is the number of involved classes and it measures the number of all the classes that compose the types of the complex columns of a table using the generalization and the aggregation relationships.

[2] On [16] it is possible to find all the basic concepts used for the definition of the metrics presented.

NSC Metric

NSC is the number of shared classes and it measures the number of involved classes for a table that are used by other tables.

Table 1 summarizes the relation between metrics and size and coupling.

Table 1. Relation between our metrics and coupling and size

	SIZE	COUPLING
TS	✓	✓
RD		✓
DRT		✓
PCC	✓	
NIC	✓	✓
NSC		✓

3.2 Example

We present the values for the different metrics for the example presented in Figure 1. For simplifying the calculus, let us assume that all methods have a cyclomatic complexity equal to 1 and that all the large objects (LOBs), such as text or bitmap, also have a size equal to one.

We can calculate the value for the address class (four simple attributes, each one with a size equal to one, divided by two, because there are two tables which use the address class) and the employee class (three simple attributes, size three, plus a method, size equal to one, divided by two, because there are two tables which use the employee class) as

$$CCS_{address} = \frac{4}{2} = 2 \qquad (1)$$

$$CCS_{employee} = \frac{3+1}{1} = 4$$

And with these values we can obtain the values shown in Table 2 for each column size of each table:

Table 2. Size for each column

	COLUMN NAME	COLUMN TYPE	COLUMN SIZE
HOUSE	Idhouse	Simple	1
	Idagency	Simple	1
	Price	Simple	1
	rooms	Simple	1
	size	Simple	1
	location	Complex	2
	desc	LOB	1
	front_view	LOB	1
	document	LOB	1
	seller	Complex	4
AGENCY	id	Simple	1
	name	Simple	1
	location	Complex	2

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

$$TS_{house} = 5+9 = 14 \qquad (2)$$
$$TS_{agency} = 2+2 = 4$$

The other metrics for the house and agency tables are summarized in Table 3.

Table 3. Metric values for the example of Figure 1

	HOUSE	Comments	AGENCY	Comments
TS	14	See above	4	See above
RD	1	Foreign Key idagency	0	No Foreign Keys
DRT	1	House to Agency	0	No Foreign Keys
PCC	20%	2 complex att. over 10	33%	1 complex att. over 3
NIC	2	address and employee	1	address
NSC	1	Address	1	address

4 Empirical Validation

In this section, we want to evaluate whether or not the proposed measures can be used as indicators for estimating the understandability of an OR database. In the remainder of this section, we present the way we collect the experimental data, the techniques used to assess the usefulness of the measures, and the results of the experiment.

4.1 Data Collection

Five object-relational databases were used in this experiment with an average of 10 relations per database. These databases were originally relational ones. For the purpose of the experiment, they were redesigned as OR databases. The maximum and minimum values for the metrics are given in Table 4.

Table 4. Range of values used in the experiment

	Minimum value	Maximum value
TS	2	17.5
RD	0	5
DRT	0	3
PCC	0%	80%
NIC	0	6
NSC	0	5

Five subjects participated in the experiment (one researcher, two research assistants and two graduate students). All of them are experienced in both relational databases and object-oriented programming. One subject did not complete the experiment, and we had to discard his partial results.

The subjects were given a form, which included three questions for each table. Our idea was that in order to answer these questions, they would need to understand the subschema (objects and relations) defined by the concerned table. A table (and thus the corresponding subschema) is easy to understand if (almost) all the subjects find the right answers in a limited time (2 minutes per table). Formally, a value 1 is assigned to the understandability of a table if at least 10 out of 12 questions are answered correctly in the specified time (4 subjects and 3 questions). A value 0 is assigned otherwise. The tables are given to the subjects in random order and not by database to minimize the effect of familiarity with the schema of a particular table.

After compiling the results, 28 tables were classified as difficult to understand (0) and 22 easy to understand (1).

4.2 Validation Technique

To analyze the usefulness of the metrics proposed in Section 3, we used two machine learning (ML) techniques: C4.5 ([12]), a Top Down Induction of Decision Trees algorithm, and RoC [13], a robust Bayesian classifier.

Most of the work done in ML has focused on supervised machine learning algorithms. Starting from the description of classified examples, these algorithms produce definitions for each class. In general, they use an attribute-value representation language that allows for the learning set statistical properties to be exploited, leading to efficient software quality models. C4.5 is representative of the Top Down Induction of Decision Trees (TDIDT) approach ([12]). C4.5 belongs to the divide and conquer algorithms family. In this family, the induced knowledge is generally represented by a decision tree. 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.

Closer to probabilistic approaches, RoC is a Bayesian classifier [7]. 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 post probability of each class label, given the attribute values, by using Bayes' theorem. The case is then assigned to the class with the highest posterior probability. 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. More information about this process is given in [13].

We uphold the use of these ML algorithms for several reasons. One of them is that real-life software engineering data are incomplete, inexact, and often imprecise; in this context, ML could provide good solutions. ML is also fairly easy to understand and use. However, perhaps the greatest advantage of an ML algorithm –as a modeling technique- over statistical analysis lies in the fact that the interpretation of production rules is more straightforward and intelligible to human beings than principal components and patterns with numbers that represent their meaning. This is very important for us because we want to obtain information about what kind of relationship can exist between our metrics and the understandability.

To evaluate the database schemata understandability characterization model based on our measures, we need criteria for evaluating the overall model accuracy. Evaluating model accuracy tells us how good the model is expected to be as a predictor. If the characterization model based on our suite of measures provides good accuracy it means that our measures are useful in identifying understandable schemes. Two criteria for evaluating the accuracy of predictions are the measures of correctness and completeness.

Correctness is defined as the percentage of database schemes that were deemed as understandable and were actually understandable. We want to maximize correctness because if correctness is low, then the model is identifying more database schemes as being understandable when they really are not understandable. Completeness is defined as the percentage of those schemes that were judged as understandable (response not understandable). We want also to maximize completeness because as completeness decreases, more schemes that were understandable are misidentified as not understandable.

The following table (Table 5) summarizes the formal measures of the learned model classification performance.

Table 5. Formal measures in the learned model

		Predicted understandability 0	Predicted understandability 1	Completeness
Real understandability	0	n_{11}	n_{12}	$\frac{n_{11}}{\sum_{j=1..2} n_{1j}}$
	1	n_{21}	n_{22}	$\frac{n_{22}}{\sum_{j=1..2} n_{2j}}$
	Correctness	$\frac{n_{11}}{\sum_{i=1..2} n_{i1}}$	$\frac{n_{22}}{\sum_{i=1..2} n_{i2}}$	

Finally the model accuracy measures how correct the model is. It is given by the following formula:

$$Accuracy = \frac{\sum_{i=1..2} n_{ii}}{\sum_{i,j=1..2} n_{ij}} \quad (3)$$

In order to calculate values for correctness and completeness, and thus fill the table given above, we used a cross-validation procedure. In this procedure, the available data is divided into N blocks so as to make each block's number of cases and class distribution as uniform as possible. N different classification models are then built, in each of which one block is omitted from the training data, and the resulting model is tested on the cases in that omitted block. In this way, each case appears in exactly one test set. Provided that N is not too small, the average error rate over the N unseen test sets is a good predictor of the error rate of a model built from all the data.

The next sub-section presents the quantitative and qualitative results obtained following our verification strategy.

4.3 Results

As specified in validation the technique section, we applied RoC and C4.5 to evaluate the usefulness of the OR metrics in estimating the understandability of the tables in an OR schema. After applying these techniques, the results are very promising. The details of the results are given below.

Using the cross-validation technique, the algorithm RoC was applied 500 times. A total of 369 cases were correctly estimated (accuracy 73.8%) and all the other cases were misclassified. Contrary to C4.5, RoC does not propose a default classification rule which guarantees coverage of all the proposed cases. However, in this experiment, it succeeded in covering all 500 cases (100% coverage). These results are summarized in Table 6.

Table 6. RoC quantitative results

Correct:	369
Incorrect:	131
Not classified:	0
Accuracy:	73.8 %
Coverage:	100.0 %

RoC produces the model presented in Figure 2. From this model, it is hard to say which metric is more relevant than another in an absolute manner. However, we can notice that the smaller the TS is the higher the probability that the table is understandable is (for example 54% for TS <= 3). This probability decreases when the table size increases (9.5% for TS >10). Inversely, the same probability increase in estimating the tables that are not understandable (varying from 13.6% for TS <= 3 to 33.6% for TS >10). For the NSC metric, a table that shares classes with other tables is hard to understand and vice versa (the highest probability in the model given by RoC, 80%). For the other metrics, 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 are defined in a narrow range (for example [0, 3] for DRT metric and [0, 5] for the RD metric).

The model of C4.5 is more accurate in estimating the understandability of a table (94%). As shown in Table 7, the model presents a high level of completeness (up to 100% for not understandable tables) and correctness (up to 100% for understandable tables).

From a qualitative point of view, C4.5 produces a more understandable model (Figure 3). As for Roc, TS seems to be an important indicator for the understandability of the tables. Rules 1, 2 and 7, which determine if a table is understandable, all have as part of the conditions that TS must be small. Inversely, in Rule 5, it is stated that a large size is sufficient to declare the table as not understandable. A small DRT is also required for Rules 1 and 7 as a partial condition to classify the table as understandable. At the same time, a high value of DRT means that the table is hard to understand (Rule 2). The number of shared classes can also be considered as a good indicator since, if there are shared classes, the table is not understandable (Rule 4) and vice versa (Rules 1 and 7). RD, PCC and NIC do not seem to be interesting indicators.

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

Fig. 2. The model generated by RoC

Table 7. C4.5 quantitative results

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

Accuracy = 94%

```
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 4:
NSC > 0 -> class 0                            [89.9%]

Rule 5:
TS > 9 -> class 0                             [82.2%]

Rule 6:
DRT > 2 -> class 0                            [82.0%]

Default class: 0
```

Fig. 3. C4.5 estimation model

In conclusion, both techniques indicate that table size and the number of shared classes are good indicators for the understandability of a table. The depth of the referential tree is also presented as an indicator by C4.5, but not clearly by RoC. The rest of the metrics do not seem to have a real impact on the understandability of a table.

A limitation of the presented work is the size and representativeness of the training sample. Object-relational databases are not widely used. The only criterion we used to choose the databases was availability. This kind of sampling, known as convenience sampling (see [10]), does not allow for the results to the whole population of OR databases to be generalized. However, as the 5 databases have small/medium schema size and the distribution of the metrics is uniform, we can consider the results as reasonably accurate for similar databases. For large schema, more experiments are needed to draw a final conclusion.

5 Conclusion

Object-relational database management systems are replacing simpler relational ones. One of the main consequences of this change will be the stronger weight of the ORDBMSs in software systems maintainability.

In this paper, we have presented a first approach for measuring object-relational database maintainability using three different metrics. To validate our measures for the understandability purpose, we have used 5 existing object-relational databases. We have applied two different techniques: C4.5, a machine learning algorithm and RoC, a Bayesian theorem-based algorithm. Two estimation models have been generated according to the two techniques. The results of our experimentation demonstrate that our measures can estimate the understandability of OR tables with a

higher level of accuracy. In particular we have found that a sub-set of our measures proved to be quite accurate (table size, depth of referential tree, and number of shared classes). This suggests that these measures can be reasonably used as indicators for the understandability of a table, and to a certain degree of its maintainability.

In spite of the obtained results, this work presents a major limitation related to the threshold values of the two models (C4.5 and RoC). These values are specific to the sample and are hard to generalize for others databases. However, we are convinced that these specific values do not significantly affect the results. The trends shown by the model are more important that the values. To solve the problem of the threshold values, we are working on a machine learning algorithm that derives fuzzy threshold values (a first version is published in [14]).

Acknowledgment. The experience was conducted with the support of CRIM (Computer Science Research Institute of Montreal).

References

1. Calero, C., Piattini, M., Ruiz, F. and Polo, M. Validation of metrics for Object-Relational Databases, International Workshop on Quantitative Approaches in Object-Oriented Software Engineering (ECOOP99), (Lisbon ,Portugal. June 1999), 14-18
2. Cannan, S.J. (1999), The New SQL Standard: Good, Bad or Simply Ugly, Jornadas de Ingeniería del Software y Bases de Datos (JISBD99), Cáceres, Spain, November 1999.
3. Card, D.N. and Glass, R.L. (1990). *Measuring Software Design Quality*. Englewood Cliffs. USA.
4. Fenton, N. Software Measurement: A Necessary Scientific Basis. *IEEE Transactions on Software Engineering*, (1994), 20(3): 199-206.
5. Fenton, N. and Pfleeger, S. L. *Software Metrics: A Rigorous Approach* 2nd edition. London, Chapman & Hall. (1997).
6. Henderson-Sellers, B. *Object-oriented Metrics - Measures of complexity*. (Upper Saddle River, New Jersey, 1996). Prentice-Hall,
7. Langley, P., Iba, W., and Thompson, K. An analysis of Bayesian Classifiers. *In Proc. of the National Conference on Artificial Intelligence*, p. 223-228, (San Mateo, CA, 1992). Morgan Kaufman.
8. Li, H.F. and Chen, W.K. An empirical study of software metrics. *IEEE Trans. on Software Engineering*, (1987), 13 (6): 679-708.
9. McCabe, T.J. A complexity measure. IEEE Trans. Software Engineering, (1976,) 2(5): 308-320.
10. Patton M. Q., Qualitative Evaluation and Research Methods. Sage Publications, 1990.
11. Pigoski, T.M. (1997). Practical Software Maintenance. Wiley Computer Publishing. New York, USA.
12. Quinlan, J.R., C4.5: Programs for Machine Learning, (1993), Morgan Kaufmann Publishers.
13. Ramoni, M. and Sebastiani, P. Bayesian methods for intelligent data analysis. In M. Berthold and D.J. Hand, editors, An Introduction to Intelligent Data Analysis, (New York, 1999). Springer.

14. Sahraoui H. A., Adel Serhani, M. and Boukadoum M. A., Extending Software Quality Predictive Models Domain Knowledge, Proc. of the 5th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, (Budapest, 2001)
15. 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.
16. Stonebraker, M. and Brown, P. *Object-Relational DBMSs tracking the next great wave*, (California, 1999), Morgan Kauffman Publishers.

CDM – Collaborative Data Model for Databases Supporting Groupware Applications

Waldemar Wieczerzycki

Department of Information Technology
The Poznan University of Economics, Poznan, Poland
wiecz@kti.ae.poznan.pl

Abstract. In the paper the CDM data model for a database that could become a kernel of cooperative applications is presented. The CDM model is oriented for the specificity of multiuser environments, in particular: cooperation scenarios (e.g. sequential, parallel, reciprocal), cooperation techniques and cooperation management.

1 Introduction

A common feature of the majority of cooperative systems is that they require functions and mechanisms naturally available in DBMSs, e.g. data persistency, access authorization, concurrency control, consistency checking and assuring, data recovery after failures, etc. Notice, however, that these functions are generally implemented in collaborative systems from scratch, without any reference to the database technology. Some systems provide gateways to classical databases, however these databases are autonomous and external to them, thus database access is organized in a conventional manner.

Since the theory and technology of classical databases is very mature, commonly accepted and verified over many years, the following question naturally arises: can we apply this technology in collaborative systems, instead of re-implementing database functions from scratch and embedding them in collaborative systems? In other words: can we develop collaborative systems as database applications, thus probably saving time normally spent on re-implementation of selected database functions? As usually we can obviously try, but there is one substantial drawback we have to take into account. The classical database paradigm assumes namely that database users are totally isolated.

In such situation, in order to develop collaborative database applications, we have to extend database technology. The required extensions should be applied simultaneously to both data modeling techniques and transaction management algorithms. Former techniques have to facilitate modeling data structures that are specific to cooperation processes, while the latter techniques have to support human interaction and exchange of non-committed data.

There are many data models proposed in the literature that are addressed to advanced domains of database applications, in particular to computer aided design (CAD) and computer aided software engineering (CASE). Most of them provide