Ninth International Conference

IPMU 2002

Information Processing and Management of Uncertainty in Knowledge-based Systems

Annecy - France
July 1-5, 2002

Proceedings
Volume I

Proceedings

Ninth International Conference

IPMU 2002

Information **P**rocessing and
Management of **U**ncertainty
in Knowledge-Based Systems

July 1-5, 2002
Volume I

Organized by Université de Savoie, LISTIC - ESIA
ANNECY, FRANCE

Editor: ESIA - Université de Savoie

Printed in France

ISBN Vol. 1: 2-9516453-1-7
Dépot légal - Juillet 2002
Imprimé par Gutenberg

To order this book, mail to ipmu2002@univ-savoie.fr

HONORARY PRESIDENT

Lotfi A. ZADEH (University of California at Berkeley)

CHAIRPERSONS

Bernadette BOUCHON-MEUNIER (CNRS, Université Paris VI, France)
Laurent FOULLOY (Université de Savoie, France)
Ronald R. YAGER (Iona College, USA)

ORGANIZING COMMITTEE

LISTIC-ESIA, Université de Savoie
G. Mauris (President), I. Alloui, E. Benoit, L. Berrah, S. Galichet

INTERNATIONAL PROGRAM COMMITTEE

J. Aczel (Canada)
J. Aguilar-Martin (France)
A. Appriou (France)
J. Baldwin (U.K.)
H. Berenji (USA)
P. Besnard (France)
J. Bezdek (USA)
I. Bloch (France)
P. Borne (France)
P. Bosc (France)
G. Coletti (Italy)
P. Diamond (Australia)
M. Delgado (Spain)
T. Denœux (France)
D. Dubois (France)
F. Esteva (Spain)
M. Fedrizzi (Italy)
D. Fogel (USA)
F. Gomide (Brazil)
M. Grabisch (France)
S. Grossberg (USA)
S. Guiasu (Canada)
J. Gutiérrez-Rios (Spain)
K. Hirota (Japan)
J. Kacprzyk (Poland)
A. Kandel (USA)
E. Klement (Austria)
G. Klir (USA)
T. Kohonen (Finland)
R. Kruse (Germany)
H. Larsen (Denmark)
L. Magdalena (Spain)
C. Marsala (France)
H. Nguyen (USA)
S. Ovchinnikov (USA)
Z. Pawlak (Poland)
W. Pedrycz (Canada)
H. Prade (France)
A. Ralescu (USA)
D. Ralescu (USA)
M. Rifqi (France)
A. Rocha (Brazil)
E. Ruspini (USA)
E. Sanchez (France)
R. Scozzafava (Italy)
G. Shafer (USA)
P. Shenoy (USA)
R. Slowinski (Poland)
P. Smets (Belgium)
M. Sugeno (Japan)
H. Thiele (Germany)
C. Tijus (France)
A. Titli (France)
L. Travé-Massuyès (Fr.)
E. Trillas (Spain)
I.B. Turksen (Canada)
L. Valverde (Spain)
A. Ventre (Italy)
J. Verdegay (Spain)
M. Vila (Spain)
H. Zimmermann (Germany)

ORGANIZERS OF SPECIAL SESSIONS

F. d'Alche-Buc
J. Benois-Pineau
I. Bloch
G. Coletti
R. Decaluwe
G. De Tre
E. Diday
M. Grabisch
F. Herrera
E. Hullermeier
C. Marsala
M.J. Martin-Bautista
R. Mesiar
E. Pap
N. Perrot
L. Polkowski
O. Pons
A. Ralescu
D. Ralescu
M. Ramdani
A.F. Rocha
U. Sandler
E. Szmidt
C. Tijus
X. Zeng

REVIEWERS

(Other non-program committee members)

H. Akdag
F. d'Alche-Buc
A. Attoui
C. Barret
S. Benferhat
J. Benois-Pineau
G. Bordogna
R. Boukezzoula
S. Boverie
J. Buckley
L. Cauffriez
C. Cayrol
S. Cimpan
R. Dapoigny
B. De Baets
M. Detyniecki
L. De Campos
G. De Cooman
J. Fauqueur
H. Fargier
J. Fodor
P.M. Frank
C. Frelicot
L. Gacogne
S. Gentil
M.A. Gil
L. Godo
Y. Grandvalet
T.M. Guerra
Z. Guessoum
M. Herbin
F. Herrera
E. Herrera-Viedma
M.C. Jaulent
P. Joly
L. Kam
L. Koczy
V. Kreinovich
L. Kuncheva
P. Lambert
J. Lawry
E. Levrat
L. Lietard
R. Lopez De Mantaras
D. Luzeaux
J.L. Marichal
R. Martin-Clouaire
M. Masson
R. Mesiar
P.A. Monney
S. Moral
N. Mouaddib
V. Novak
A. Nürnberger
F. Oquendo
E. Pairel
R. Palm
E. Pap
G. Pasi
J.I. Pelaez
I. Perfilieva
O. Pivert
L. Polkowski
L. Ralaivola
M. Rombaut
L. Saitta
S. Sandri
H. Sawada
A. Skowron
E. Szmidt
S. Termini
V. Torra
E. Trouve
L. Ughetto
D. Van De Ville
F. Vernadat
S. Zadrozny

ACKNOWLEDGEMENTS

The organizers of the 9th IPMU Conference would like to thank all institutions and people without them this conference could not be held.

For their sponsorship

- Association Française pour l'Intelligence Artificielle (AFIA)
- European Society for Fuzzy Logic and Technology (EUSFLAT),
- GDR Information, Signal, Images et ViSion (ISIS)
- Institute of Electrical and Electronics Engineers (IEEE),
- International Fuzzy Systems Association (IFSA)
- World Organisation of Systems and Cybernetics (WOSC)

For their support

- Laboratoire d'Informatique de Paris 6 (LIP6)
- Ecole Supérieure d'Ingénieurs d'Annecy (ESIA)
- Université de Savoie
- Ministère de l'Education Nationale
- Ministère des Affaires Etrangères
- Centre National de la Recherche Scientifique (CNRS)
- Chambre de Commerce et d'Industrie (CCI)
- Communauté d'Agglomération d'Annecy (CAA)
- Conseil Général de la Haute-Savoie
- Région Rhône-Alpes

For their help in the organisation

- Pôle Mécatronique et Management de Haute-Savoie (THESAME)

For their help in preparing the conference

- The members of the International Program Committee
- The organizers of special sessions
- The members of the organizing committee
- The session chairpersons
- The referees
- and all those who helped

Table of Contents

Fuzzy Prototypical Knowledge Discovery to predict information systems maintainability

José A. Olivas
Department of Computer Science
University of Castilla-La Mancha
Paseo de la Universidad 4
13071, Ciudad Real (Spain)
joseangel.olivas@uclm.es

Marcela Genero
Department of Computer Science
University of Castilla-La Mancha
Paseo de la Universidad 4
13071, Ciudad Real (Spain)
marcela.genero@uclm.es

Mario Piattini
Department of Computer Science
University of Castilla-La Mancha
Paseo de la Universidad 4
13071, Ciudad Real (Spain)
mario.piattini@uclm.es

Francisco P. Romero
Soluziona Consultoría
Soluziona Software Factory,
c/ Pedro Muñoz 1,
13071-Ciudad Real, (Spain)
fromero@uf.isf.es

Abstract

This contribution presents a new approach, based on Fuzzy Prototypical Knowledge Discovery, to predict the maintainability of class diagrams done using the Unified Modelling Language (UML), which has great importance on the final quality of object-oriented information systems (OOIS). The prediction model is built from metrics values obtained at the early phases of OOIS life-cycle. We will start with Fuzzy Prototypical Knowledge Discovery process (FPKD) for finding Fuzzy Deformable Prototypes of class diagram maintainability, and later we will predict a real case class diagram maintainability deforming the similar prototypes using the degree of compatibility with them.

Keywords: Fuzzy Prototypes, Data Mining, OOIS Maintainability, Class Diagrams.

1. Introduction

A great effort has been made in the field of software measurement for improving the quality of the OOIS [14], [18], [31], [11], but most of them pursue the goal of evaluating -by means of quantitative measures- the quality of the final product, i.e. the code or the advanced design. But, in Software Engineering it is widely accepted that the quality of OOIS is highly dependent on the decision taken early in the development. So that, we believe that in order to get better quality OOIS we should focus on measuring internal quality characteristics of early artifacts, such as class diagrams, and based on those measurements thereby obtain early in the life-cycle a prediction model for external quality characteristics, such as for example maintainability, which is one of the major concerns of software developers and industries.

In response to the great demand for measures for measuring quality characteristics of class diagrams, such as maintainability [13], and after a thorough review of some of the existing OO measures that can be applied at a high level design stage [8], [16], [5], [17], we have proposed a set of measures for UML class diagram structural complexity [12]. As maintainability is an external quality characteristic that can be evaluated once a product is finished or nearly finished, we centre our work on measuring an internal quality characteristic, the structural complexity of class diagrams. Our idea is to use those measures to predict class diagram maintainability early in the OOIS development.

This paper is organised in the following way. In section 2 we will present a set of metrics for UML class diagram structural complexity. In section 3 we explain how we carried out a controlled experiment in order to evaluate if there is empirical evidence that UML class diagram structural complexity metrics are correlated with maintainability sub-characteristics: such as understandability, analysability and modifiability (ISO, 1999). In section 4 we will use the empirical data for building fuzzy prototypes, that characterise UML class diagram maintainability, and based on those prototypes, in section 5, we will build a prediction model for UML class diagram maintainability. The last section summarises the paper, presents some conclusions and identifies further work.

2. Metrics for UML class diagram structural complexity

We only present here those metrics presented in [12], which can be applied at class diagram level as a whole (see table 1). These metrics measure the structural complexity of UML class diagrams due to the use of relationships, such as associations, generalisations, aggregations and dependencies. We also consider traditional metrics such as, the number of classes, the number of attributes, etc.

Table 1. Metrics for UML class diagram structural complexity

Metric name	Metric definition
NUMBER OF CLASSES (NC)	The total number of classes.
NUMBER OF ATTRIBUTES (NA)	The total number of attributes.
NUMBER OF METHODS (NM)	The total number of methods
NUMBER OF ASSOCIATIONS (NAssoc)	The total number of associations
NUMBER OF AGGREGATION (NAgg)	The total number of aggregation relationships within a class diagram (each whole-part pair in an aggregation relationship)
NUMBER OF DEPENDENCIES (NDep)	The total number of dependency relationships
NUMBER OF GENERALISATIONS (NGen)	The total number of generalisation relationships within a class diagram (each parent-child pair in a generalisation relationship)
NUMBER OF AGGREGATIONS HIERARCHIES (NAGGH)	The total number of aggregation hierarchies in a class diagram.
NUMBER OF GENERALISATIONS HIERARCHIES (NgenH)	The total number of generalisation hierarchies in a class diagram
MAXIMUM DIT	It is the maximum between the DIT value obtained for each class of the class diagram. The DIT value for a class within a generalisation hierarchy is the longest path from the class to the root of the hierarchy.
MAXIMUM HAGG	It is the maximum between the HAgg value obtained for each class of the class diagram. The HAgg value for a class within an aggregation hierarchy is the longest path from the class to the leaves.

3. Empirical validation of the proposed metrics

In this section we describe an experiment we have carried out for empirically validating the proposed metrics (see section 2). We will only be able to draw conclusions about the relationship between the cause and the effect for which we stated a hypothesis (which we want to corroborate by means of experiments); if the experiment is properly set up.

To perform an experiment, several steps have to be taken and they have to be in a certain order.

3.1 Definition

As Wholin et al. [29], suggested, we follow the GQM template [1], [2], [3], [28], for goal definition.

3.2 Planning

After the definition of the experiment, the planning took place. The definition determines the foundation of the experiment -why the experiment is conducted- while the planning prepares for how the experiment is conducted.

3.2.1 Context selection

The context of the experiment is a group related to the area of Software Engineering at the university, and hence the experiment is run-off line (not industrial software development), it is conducted by 7 teachers and 10 students enrolled in the final-year of Computer Science in the Department of Computer Science at the University of Castilla-La Mancha in Spain. All of the teachers belong to the Software Engineering area.

The experiment is specific since it is focused on UML class diagram structural complexity metrics. The ability to generalise from this specific context is further elaborated below when discussing threats to the experiment. The experiment addresses a real problem the correlation between metrics and maintainability sub-characteristics.

3.2.2 Hypothesis formulation

An important aspect of experiments is to know and to state in a clear sand formal fashion what we intend to evaluate in the experiment. This lead us to the formulation of a hypothesis (or several hypothesis). We wish to test the hypothesis that there is a significant correlation between the current metric data set (NC, NA, NM, NAssoc, NAgg, NDep, NGen, NAggH, NGenH, MaxHAgg, MaxDIT) and the subject's rating of three maintainability sub-characteristics, such as understandability, analysability and modifiability.

3.2.3 Instrumentation

The objects were class diagrams done using UML.

The independent variable (UML class diagram structural complexity) was measured through the metrics, presented in section 2.

The dependent variables (understandability, analysability and modifiability) were measured according to subject's rating.

3.2.4 Validity evaluation

We will discuss the empirical study's various threats to validity and the way we attempted to alleviate them:

Threats to Construct Validity. We propose subjective metrics for measuring each of the dependent variables (maintainability sub-characteristics) based on the judgement of the subjects. As the subjects involved in this experiment have medium experience in UML class diagram design we think their ratings can be considered significant. The independent variables (each of the metrics proposed in section 2) that measure the structural complexity of class diagrams can also be considered constructively valid, because from a system theory point of view, a system is called complex if it is composed of many (different types of elements), with many (different types of) (dynamically changing) relationships between them [25].

Threats to Internal Validity. Seeing the results of the experiment we can conclude that empirical evidence of the existing relationship between the independent and the dependent variables exists. We have tackled different aspects that could threaten the internal validity of the study, such as: differences among subjects, knowledge of the universe of discourse among class diagrams, accuracy of subject responses, learning effects, fatigue effects, persistence effects and subject motivation.

Threats to External Validity. Two threats to external validity have been identified which limit the ability to apply any such generalisation, and we have tried to alleviate them: materials and tasks, and subject selection. In general in order to extract a final conclusion that can be generalised, we need to replicate this experiment with a greater number of subjects, including practitioners. After doing replication we will have a cumulative body of knowledge; which will lead us to confirm if the presented metrics could really be used as early quality indicators, and could be used to predict class diagram maintainability.

3.3 Operation

3.3.1 Preparation

By the time the experiment was done all of the students had had two courses on Software Engineering, in which they learnt in depth how to build OO software using UML. All the selected professors had enough experience in the design and development of OOIS. Moreover, subjects were given an intensive training session before the experiment took place.The subjects were not aware of what aspects we intended to study. Neither they were aware of the actual hypothesis stated.

We prepared the material we had to give to the subjects, consisting of 28 class diagrams of the same universe of discourse, related to Bank Information Systems. Each diagram has a test enclosed which includes the description of maintainability sub-characteristics, such as: understandability, analysability, modifiability. Each subject has to rate each sub-characteristic using a scale consisting of seven linguistic labels. For example for understandability we proposed the following linguistic labels:

Extremely difficult to underst	Very difficult to underst	A bit difficult to underst	Neither difficult nor easy to underst	Quite easy to underst	Very easy to underst	Extremely easy to underst

We also prepared a debriefing questionnaire. This questionnaire included (i) personal details and experience, (ii) opinions on the influence of different components of UML class diagrams, such as: classes, attributes, associations, generalisations, etc... on their maintainability.

3.3.5 Execution

The subjects were given all the material described in the previous section. We explained to them how to carry out the experiment. We allowed one week to do the experiment, i.e., each subject had carry out the test alone, and could use unlimited time to solve it.

We collected all the data, including subjects´rating obtained from the responses of the experiment and the metrics values automatically calculated by means of a metric tool we had designed.

3.3.6 Data Validation

All tests were considered valid because all of the subjects have at least medium experience in building UML class diagrams and developing OOIS.

3.4 Analysis and Interpretation

Analyzing the Spearman´s correlation coefficients, we can conclude that there exists a high correlation between most of the UML class diagram structural complexity metrics and the subject´s rating of understandability, analysability and modifiability. We can deduce this due to the fact that almost all the metrics have a correlation greater than 0.7. NDep is the only one that has a lesser correlation. This fact should be studied in detail by carrying out further experimentation.

3.5 Presentation and package

The last activity is concerned with presenting and packaging of the findings. The diffusion of the experimental results and the way they are presented are relevant so that they are really put into use. Therefore we published our findings in this paper, and we are also planning to publish a lab package on the web for replication purposes.

4. A prediction model for UML class diagram maintainability

In this section we explain the steps involved in the Fuzzy Prototypical Knowledge Discovery (FPKD) process Olivas et al. [20], [21], [22], which lead us to the construction of fuzzy prototypes (Zadeh, 1982) [30], that characterize the maintainability of UML class diagrams. The FPKD is a fuzzy extension of the traditional Knowledge Discovery in Databases (KDD) [9].

Zadeh [30] mentioned the classical prototype theories from the point of view of psychology, criticizing that these theories do not fit the function that a prototype should have. Zadeh's approach to what must be taken as a prototype is less intuitive than the conceptions of psychological theories but is more rational and closer to the meaning of a prototypical concept displayed in a more detailed examination. In our case, we have observed that Zadeh's idea suggests a concept that encompasses a set of prototypes, which represent the high, medium, or low compatibility of the samples with the concept A. "The prototype is not a single object or even a group of objects in A. Rather, it is a fuzzy schema for generating a set of objects which is roughly coextensive with A" [30].

The prototypes obtained from the FPKD, will form the foundation of the prediction model that allows us to predict class diagram maintainability. This approach is more representative than standard approaches, because the use of an isolated algorithm or method over- simplifies the complexity of the problem. Statistical methods or decision trees (ID3, C4.5, CART) are only classification processes, and it is very important to include a clustering model for finding some kinds of patterns in the initial chaos of data. The use of fuzzy schemas for representing these patterns allows us to achieve better and more understandable results.

4.1 The FPKD process

The FPKD process consist of different steps:

Selection of the target data. We have taken as a start set a relational database that contains 476 records (with 14 fields, 11 represent metric values, 3 represent maintainability sub-characteristics, understandability, analysability and modifiability respectively) obtained from the calculation of the metric values (for each class diagram) and the responses of the experiment given by the subjects.

Preprocessing. The Data-Cleaning was not necessary because we did not find any errors.

Transformation. This step was performed doing different tasks:

Summarising subject responses. We built a table with 28 records (one record for each class diagram) and 14 fields. The metric values were calculated measuring each diagram, and the values for each maintainability sub-characteristics were obtained aggregating subjects's ratings using their mean.

Clustering by Repertory Grids. In order to detect the relationships between the class diagrams, to obtaining those which are easy, medium or difficult to maintain (based on subject rates of each maintainability sub-characteristics), we have carried out a hierarchical clustering process by Repertory Grids, based on subject's rating for each diagram. The set of elements is composed of the 28 class diagrams, the constructions are the intervals of values of the subject's rating. The application of Repertory Grids Analysis Algorithm (hierarchical clustering algorithm) returns a graphic which reflects each prototype (easy, medium and difficult to maintain), and the class diagrams which pertain to them (see figure1).

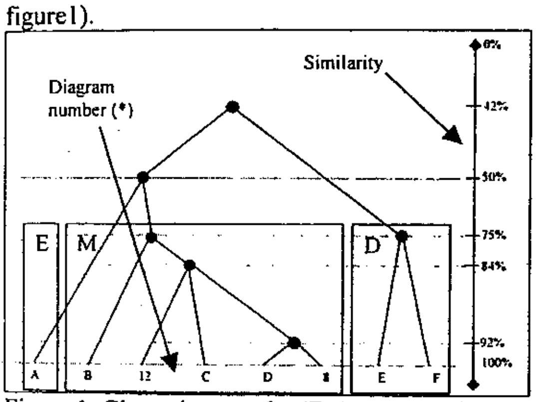

Figure 1. Clustering results (E: Easy to maintain, M: Medium to maintain, D: Difficult to maintain).

(*) We have grouped some class diagrams assigning them one letter because they have 100% similarity (see Clustering Results in Fig.2)

Data Mining. The selected algorithm for the data mining process was summarise functions. Table 3 shows the parametric definition of the prototypes.

Table 3. Prototypes "Easy, Medium and Difficult to maintain"

	Understandability	Analisability	Modifiability
Difficult			
Average	6	6	6
Maximum	6	6	7
Minimum	6	5	6
Medium			
Average	5	5	5
Maximum	5	6	5
Minimum	4	4	4
Easy			
Average	2	2	3
Maximum	3	3	3
Minimum	2	2	2

Formal Representation of conceptual prototypes. The prototypes have been represented as fuzzy numbers, which are going to allow us to obtain a degree of membership in the concept. In order to construct the prototypes (triangular fuzzy numbers) we only need to know their centerpoints ("center of the prototype"), which are obtained by normalising and aggregating the metric values corresponding to the class diagrams of each of the prototypes (see figure 2).

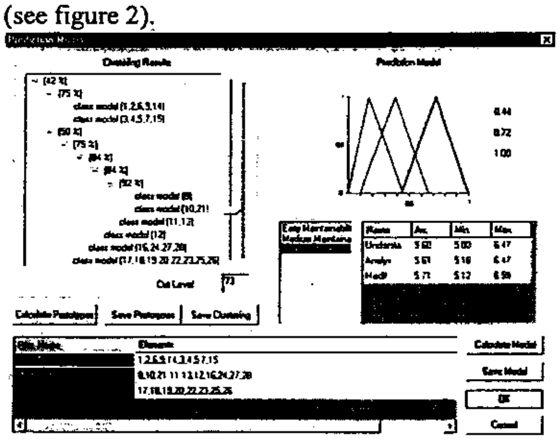

Figure 2. Representation of the prototypes.

5. Example of prediction of UML class diagram maintainability

Using Fuzzy Deformable Prototypes (Olivas et al. [20], [21], [22]), we can deform the prototypes with some degree of affinity to a new class diagram (change the values for describing exactly the new real situation), using a linear combination with the degrees of membership as coefficients of the prototypes vectors. We will give an example of how to deform the fuzzy prototypes found in section 4.1.

Given the following metric values corresponding to a new class diagram:

NC	NA	NM	NAssoc	NAgg	NAggH	NDep	NGenR	NGenH	MaxDIT	MaxHagg
21	30	70	10	6	2	3	20	5	2	3

And their normalised values:

NC	NA	NM	NAssoc	NAgg	NAggH	NDep	NgenR	NGenH	MaxDIT	MaxHAgg
0.69	0.48	0.67	0.71	0.67	0.67	0.75	0.83	1	0.40	0.75

The final average is 0.65. The affinity with the prototypes is shown in figure 3.

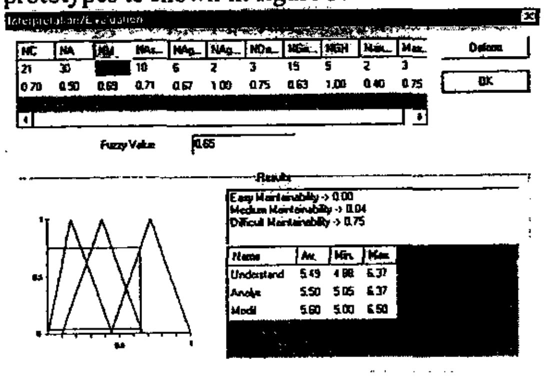

Figure 3. Affinity of the real case with the prototypes.

The prototypes with some degree of affinity for this new class diagram are "Difficult to maintain" and "Medium to maintain". Then, the prediction based on the deformation of these prototypes is (see also fig. 3):

	Understandability	Analysability	Modifiability
Average	5	5	6
Maximum	5	5	6
Minimum	6	5	6

6. Conclusions and future work

In this paper we have used Genero et al.'s metrics [12], with the objective of corroborating that there exists a great correlation between these metrics values and the maintainability of a class diagram, we have carried out a controlled experiment.

Also we have used a fuzzy extension of the traditional KDD process, the FPKD (Olivas et al.[20], [21], [22]), for building the maintainability prototypes which serve as the basis of the prediction model for the sub-characteristics that affect class diagram maintainability.

The prediction model was built using the FPKD process. This process was used not only in the software measurement area, but was also used for different kinds of real problems, such as forest fire prediction [20], [22], financial analysis or medical diagnosis, obtaining satisfactory results.

We want to highlight that this is a first approach to predicting UML class diagram maintainability, we need "real data" about UML class diagram maintainability efforts, such as time spent in maintenance tasks in order to predict data that can be highly useful to software designers and developers.

Nevertheless, despite the promising nature of the obtained results, towards of seeking correct OO metrics applied at a high level design stage, we are aware that we need to do more metric validation, both empirical and theoretical in order to obtain conclusive evidence of the usefulness of the proposed metrics.

Pending is the theoretical validation of the proposed metrics using the DISTANCE framework proposed by Poels and Dedene [24], [25], which is in our knowledge the most appropriate for OO measurements.

In future work, we will also tackle the measurement of other quality factors like those proposed in the ISO 9126 [13], which not only addresses class diagrams, but also evaluates other UML diagrams, such as use-case diagrams, state diagrams, etc. To our knowledge, little work has been done towards measuring dynamic and functional models [25]. As is quoted in Brito e Abreu et al. [6], this is an area which lacks in depth investigation.

Acknowledgements

This research is part of the DOLMEN project supported by CICYT (TIC 2000-1673-C06-06) and the CIPRESES project supported by CICYT (TIC 2000-1362-C02-02).

References

[1] Basili V. and Weiss D. (1984). A Methodology for Collecting Valid Software Engineering Data, *IEEE Transactions on Software Engineering*, 10, 728-738.

[2] Basili V. and Rombach H. (1988). The TAME Project: Towards Improvement-Oriented Software Environments. *IEEE Transactions on Software Engineering 14*, 758-773.

[24] Poels G. and Dedene G. (1999). DISTANCE: A Framework for Software Measure Construction, research report DTEW9937, Dept. Applied Economics, Katholieke Universiteit Leuven, Belgium, 46 p.

[25] Poels G. and Dedene G. Measures for Assessing Dynamic Complexity Aspects of Object-Oriented Conceptual Schemes. In: *Proceedings of the 19th International Conference on Conceptual Modeling* (ER 2000), Salt Lake City, (2000a), 499-512.

[26] Poels G. and Dedene G. (2000b). Distance-based software measurement: necessary and sufficient properties for software measures. *Information and Software Technology*, 42(1), 35-46.

[27] Schneidewind N. (1992). Methodology For Validating Software Metrics. *IEEE Transactions of Software Engineering*, 18(5), 410-422.

[28] Van Solingen R. and Berghout E. (1999). The Goal/Question/Metric Method: *A practical guide for quality improvement of software development*. McGraw-Hill.

[29] Wohlin C., Runeson P., Höst M., Ohlson M., Regnell B. and Wesslén A. (2000) *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers.

[30] Zadeh L. (1982). A note on prototype set theory and fuzzy sets. *Cognition* 12, 291 – 297.

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