Vol-84

ADIS 2001

Apoyo a la Decisión en Ingeniería del Software - Decision Support in Software Engineering.

Under support of CICYT TIC2000-2052-E and CICYT TIC2001-1143

Proceedings of the II ADIS 2001 Workshop on Decision Support in Software Engineering

Almagro, Ciudad Real, Spain, November 20, 2001

Edited by

José Javier Dolado [1]

Mario Piattini [2]

Miguel Toro [3]

Juan J. Cuadrado [4]

(1) Universidad del País Vasco - Euskal Herriko Unibertsitatea, Spain

(2) Universidad de Castilla û La Mancha, Spain

(3) Universidad de Sevilla, Spain

(4) Universidad Carlos III de Madrid, Spain

Table of Contents

Submitted by J.J. Cuadrado, October 16, 2003

A controlled experiment for corroborating the usefulness of class diagram metrics at the early phases of OO developments

Marcela Genero, José Olivas, Mario Piattini, Francisco Romero
Department of Computer Science
University of Castilla-La Mancha
Ronda de Calatrava, 5
13071, Ciudad Real (Spain)
{mgenero, jaolivas, mpiattin}@inf-cr.uclm.es

Abstract. The quality of class diagrams is critical because it has a great influence on the quality of the object oriented information system (OOIS) which are finally delivered. This fact motivated us to define a set of measures for evaluating the structural complexity (an internal quality attribute) of class diagrams made using the Unified Modeling Language (UML), which nowadays is the standard language for object oriented modelling. These measures could be used to predict class diagram external quality characteristics, such as maintainability, early in the OOIS life-cycle. In order to corroborate the practical utility of those metrics, we have put them under empirical validation by means of a controlled experiment. The description of each of the steps carried out to perform the experiment and the construction of the prediction model for class diagram maintainability are the main goals of this paper.

Keywords. object oriented metrics, class diagram structural complexity, class diagram maintainability, information system quality, empirical validation, prediction models, fuzzy prototypical knowledge discovery

1. Introduction

A great effort has been made in the field of software measurement for improving the quality of the OOIS (Henderson-Sellers, 1996; Melton, 1996, Zuse, 1998; Fenton and Pflegger, 1997), 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 (ISO, 9126) and after a thorough review of some of the existing OO measures that can be applied at a high level design stage (Chidamber and Kemerer, 1994; Lorenz and Kidd, 1994, Brito e Abreu and Carapaçua, 1994; Marchesi, 1998) we have proposed a set of measures for UML class diagram structural complexity in Genero et al. (2000). 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.

We have defined those measures in a methodological way (Calero et al., 2001) including three main tasks: metric definition, theoretical validation and empirical validation. Although the three steps are equally relevant in order to define correct and reliable metrics, we focus this paper on the empirical validation of the proposed metrics. Empirical validation is critical for the success of any measurement activity (Kitchenham et al., 1995; Fenton and Pflegeer, 1997; Schneidewind, 1992; Basili et al., 1999).Through empirical validation we can demonstrate with real evidence that the measures we proposed serve the purpose they were defined for and that they are fruitful in practice.

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 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 Genero et al. (2000) 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.

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.

Table 1. Metrics for UML class diagram structural complexity

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. Therefore, we have followed some suggestions provided by Wholin et al. (2000), Perry et al. (2000) and Briand et al. (1999) about how to perform controlled experiments.

To perform an experiment, several steps have to be taken and they have to be in a certain order. The experiment process can be divided into the following main activities:

1. Definition, where we define the experiment in terms of problem, objective and goals.
2. Planning, where the design of the experiment is determined, the instrumentation is considered and the threats to the experiment are evaluated
3. Operation, in this phase measurements are collected.
4. Analysis and Interpretation, where collected data are analysed and evaluated.
5. Presentation and Package, where results are presented and packaged.

In the remainder of this section we explain how we have performed each of the activities described above.

3.1 Definition

As Wholin et al. (2000) suggested, we follow the GQM template (Basili and Weiss, 1984; Basili and Rombach , 1988; Van Solingen and Berghout, 1999) for goal definition. This results in the following goal:

Analyse	*UML class diagrams complexity metrics*
For the purpose of	*Evaluating*
With respect to	the correlation with maintainability sub-characteristics
From the point of view of	*researchers*
In the context of	*M.Sc. students and professors of the Engineering Software Area in the Department of Computer Science in the University of Castilla-La Mancha.*

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 professors 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 professors 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 Variables selection

The independent variable is the UML class diagram structural complexity

The dependent variables are three maintainability sub-characteristics: understandability, analysability and modifiability.

3.2.4 Selection of subjects

The subjects are chosen for convenience, i.e. the subjects are students and professors that have experience in the design and development of OOIS.

3.2.5 Experiment design

We selected a within-subject design experiment, i.e. all the tests were solved by the same group of subjects. The tests were put in a different order for each subject.

3.2.6 Instrumentation

The objects were class diagrams done using UML.
The independent variable was measured through the metrics, presented in section 2.
The dependent variables were measured according to subject's rating.

3.2.7 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 (see section 3.2). 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 (Poels and Dedene, 2000a).
- 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 understand	Very difficult to understand	A bit difficult to understand	Neither difficult nor easy to understand	Quite easy to understand	Very easy to understand	Extremely easy to understand

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 prveous section. We explained tothem 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

As we have said before, our goal is to ascertain if any correlation exists between each of the proposed metrics (see section 2) and three of the maintainability sub-characteristics: understandability, analisabilisty and modifiability.

Spearman's correlation was used to determine the correlation of the data collected in the experiment, shown in Appendix A. The correlation coefficient is a measure of the ability of one variable to predict the value of another variable. Using Spearman's correlation coefficient, each of the metrics was correlated separately to the different subject's rates of understandability, analysability and modifiability (see table 2).

	NC	NA	NM	NAssoc	NAgg	NDep	NGen	NAggH	NGenH	MaxHagg	MaxDIT
Understandability	0.961	0.941	0.929	0.753	0.813	0.518	0.876	0.714	0.902	0.728	0.749
Analysability	0.966	0.940	0.916	0.733	0.822	0.534	0.868	0.720	0.921	0.722	0.738
Modifiability	0.950	0.924	0.908	0.733	0.818	0.522	0.865	0.719	0.888	0.725	0.751

Table 2. Spearman's correlation between UML class diagrams structural complexity metrics and understandability, analysability and modifiability

Analysing the Spearman's correlation coefficients shown in table 4, 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 and Romero, 2000; Olivas, 2000), which lead us to the construction of fuzzy prototypes (Zadeh, 1982) that characterise the maintainability of UML class diagrams. The FPKD is a fuzzy extension of the traditional Knowledge Discovery in Databases (KDD) (Fayyad, 1996).

The prototypes obtained from the FPKD 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 allows us to achieve better and more understandable results, concerning patterns and prediction 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 (see Appendix A). 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 subjects'rating. The application of Repertory Grids Analysis Algorithm returns a graphic which reflects each prototype (easy, medium and difficult to maintain), and the class diagrams which pertain to them (see figure1).

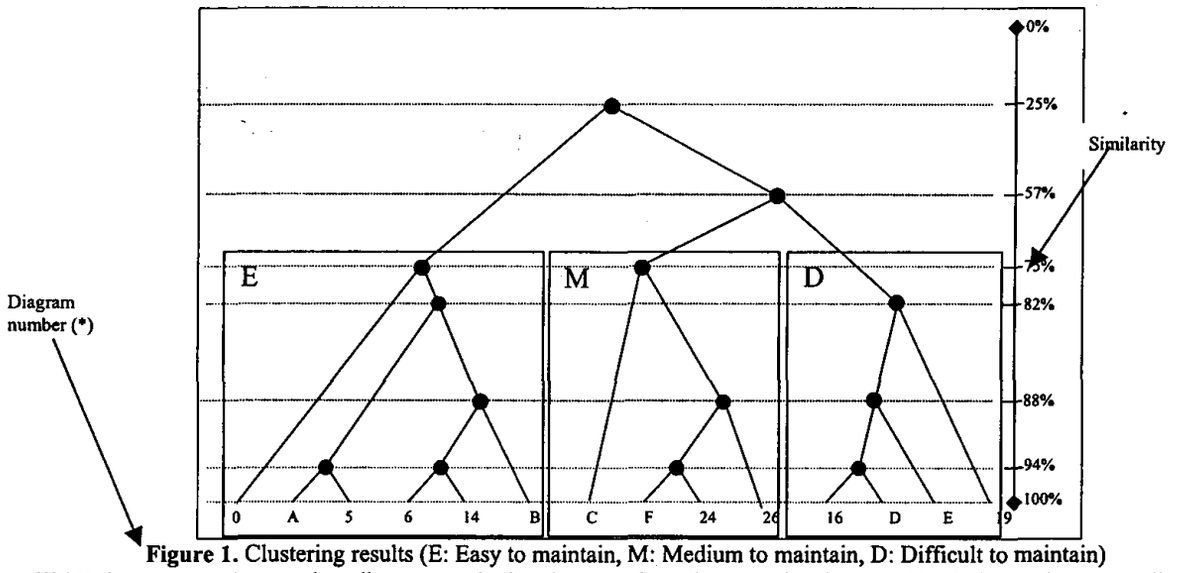

Figure 1. Clustering results (E: Easy to maintain, M: Medium to maintain, D: Difficult to maintain)
(*) We have grouped some class diagrams assigning them one letter because they have 100% similarity (see appendix A)

- DATA MINING. The selected algorithm for the data mining process was summarise functions. Table 3 shows the parametric definition of the prototypes.

	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

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

- 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).

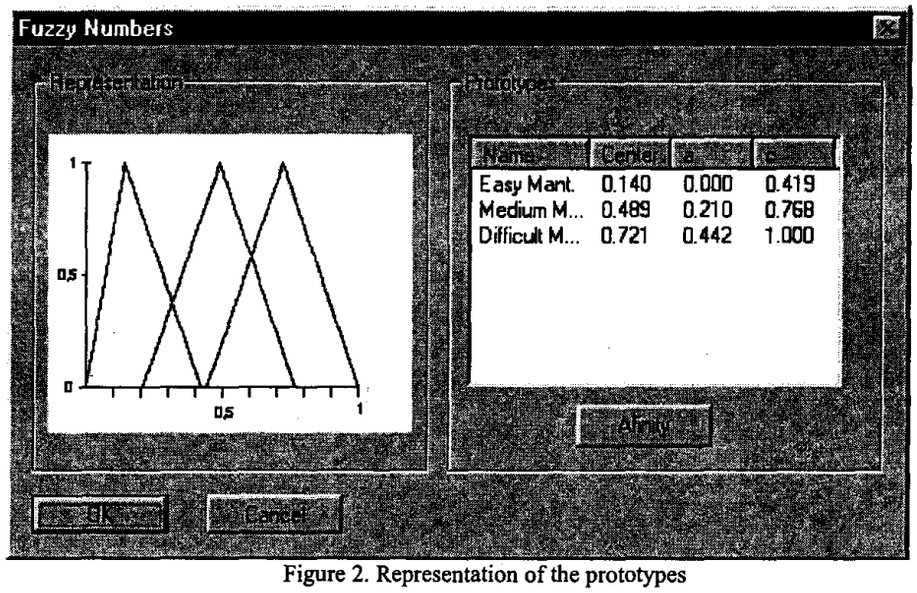

Figure 2. Representation of the prototypes

5. Example of prediction of UML class diagram maintainability

Using Fuzzy Deformable Prototypes (Olivas and Romero, 2000; Olivas, 2000), we can deform the most similar prototype to a new class diagram, and define the factors for a new situation, using a linear combination with the degrees of membership as coefficients. 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 0.69. The affinity with the prototypes is shown in figure 3.

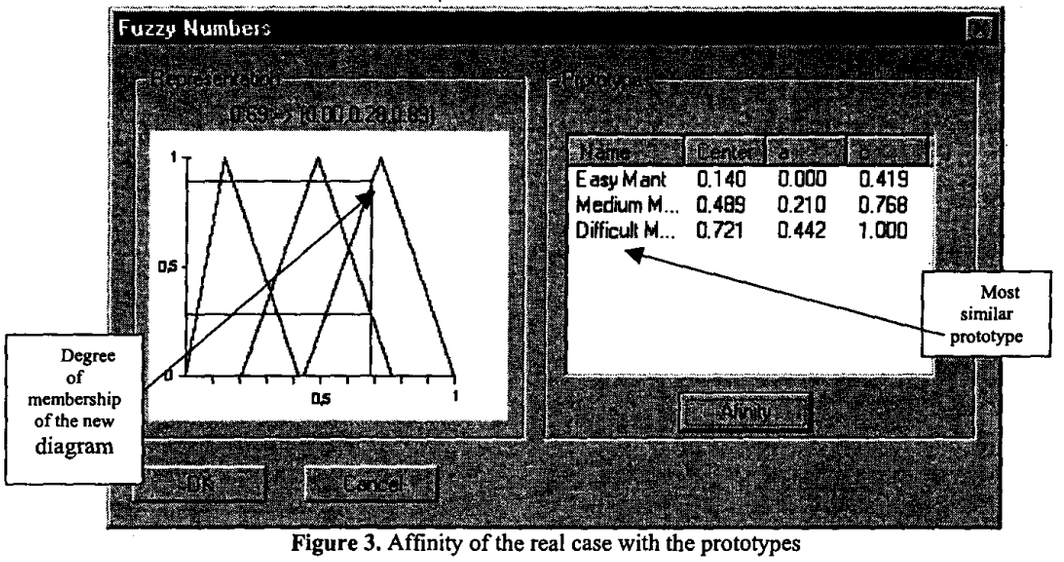

Figure 3. Affinity of the real case with the prototypes

The most similar prototype for this new class diagram is "Difficult to maintain", with a degree of membership of 0.89. Then, the prediction is:

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

6. Conclusions and future work

In this paper we have presented Genero et al.'s metrics (Genero et al., 2000), which are defined to assess the structural complexity of UML class diagrams obtained at high level design stage. These metrics allow OO designers:

1. a quantitative comparison of design alternatives, and therefore an objective selection among several class diagram alternatives with equivalent semantic content.
2. a prediction of external quality characteristics, like maintainability in the initial phases of the OOIS life cycle and a better resource allocation based on these predictions.

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. Analysing the data collected using Spearman 's correlation we have concluded that most of the proposed metrics are highly correlated with the maintainability characteristics such as: understandability, analysability and modifiability.

Also we have used a fuzzy extension of the traditional KDD process, the FPKD (Olivas and Romero, 2000; Olivas, 2000) for building the maintainability prototypes which serve as the basis of the prediction model for the sub-characteristics that affect class diagram maintainability.

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.

The prediction model was built using the FPKD process, which is a fuzzy extension of the traditional KDD. The FPKD 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, financial analysis or medical diagnosis, obtaining satisfactory results.

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 (1999; 2000b), which is in our knowledge the most appropriate for OO measurements.

Regarding empirical validation we are refining this experiment in order to replicate it. For example we have found out that it is not necessary to include 28 diagrams, but it would be possible to take only the most representative. We are also designing a new experiment, in which we will give the subjects several class diagrams and some new requirements to be added. In this case the independent variable will be measured by the time spent in modification tasks, which is more objective than subjects' rating.

We also 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 fruitful to software designers and developers. However the scarcity of such data continues to be a great problem which we must tackle to validate metrics. Brito e Abreu et al. (1999) suggested the necessity of a public repository of measurement experiences, which we think could be a good step towards achieving success in all the work done related to software measurement.

Once the proposed metrics are refined (i.e. validated or discarded) we have the plan to embed them into an OO CASE tool, for helping OO designers to take better decisions in their design tasks, which is the most important goal of any measurement proposal that aims to be useful (Fenton and Neil, 2000).

In future work, we will also tackle the measurement of other quality factors like those proposed in the ISO 9126 (1999), 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 (Poels and Dedene, 2000a). As is quoted in Brito e Abreu et al. (1999) 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).

References

- Basili V. and Weiss D. (1984). A Methodology for Collecting Valid Software Engineering Data, *IEEE Transactions on Software Engineering*, 10, 728-738.
- Basili V. and Rombach H. (1988). The TAME Project: Towards Improvement-Oriented Software Environments. *IEEE Transactions on Software Engineering* 14, 758-773.
- Basili V., Shull F. and Lanubile F. (1999). Building knowledge through families of experiments. *IEEE Transactions on Software Engineering*, 25(4), 435-437.
- Briand L., Bunse C. and Daly J. A Controlled Experiment for evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. *Technical Report IESE 002.99/E, Fraunhofer Institute for Experimental Software Engineering*, Kaiserslautern, Germany, (1999).
- Brito e Abreu F. and Carapaçua R. (1994). Object-Oriented Software Engineering: Measuring and controlling the development process. *4th Int Conference on Software Quality*, Mc Lean, Va, USA.
- Brito e Abreu F., Zuse H., Sahraoui H. and Melo W. (1999). Quantitative Approaches in Object-Oriented Software Engineering. *Object-Oriented technology: ECOOP'99 Workshop Reader*, Lecture Notes in Computer Science 1743, Springer-Verlag, 326-337.
- Calero C., Piattini M. and Genero M. (2001). Metrics for controlling database complexity. In *Developing Quality Complex Databases*. Shirley Becker Ed. Idea Group Publishing
- Chidamber S. and Kemerer C. (1994). A Metrics Suite for Object Oriented Design. *IEEE Transactions on Software Engineering*. 20(6), 476-493.
- Fayyad U., Piatetsky-Shapiro G. and Smyth P. (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39(11), 27 – 34.
- Fenton N. and Neil, M. (2000). Software Metrics: a Roadmap. *Future of Software Engineering*. Ed:Anthony Finkelstein, ACM, 359-370.
- Fenton N. and Pfleeger S. (1997). *Software Metrics: A Rigorous Approach*. 2nd. edition. London, Chapman & Hall.
- Genero, M., Piattini, M. and Calero, C. (2000). Early Measures For UML class diagrams. *L'Objet*. 6(4), Hermes Science Publications, 489-515.
- ISO/IEC 9126-1.2. (1999). Information technology- Software product quality – Part 1: Quality model.
- Henderson-Sellers B. (1996). *Object-Oriented Metrics - Measures of complexity*. Prentice-Hall, Upper Saddle River, New Jersey.
- Kitchenham, B., Pflegger, S. and Fenton, N. Towards a Framework for Software Measurement Validation. *IEEE Transactions of Software Engineering*, 21(12), (1995) 929-943.
- Lorenz M. and Kidd J. (1994). *Object-Oriented Software Metrics: A Practical Guide*. Prentice Hall, Englewood Cliffs, New Jersey.
- Marchesi M. (1998). OOA Metrics for the Unified Modeling Language. *Proceedings of the 2nd Euromicro Conference on Software Maintenance and Reengineering*, 67-73.
- Melton, A. (ed.) (1996). *Software Measurement*. London. International Thomson Computer Press.

- Object Management Group. (1999). *UML Revision Task Force. OMG Unified Modeling Language Specification, v. 1.3. document ad/99-06-08.*
- Olivas J. A. and Romero F. P. (2000). FPKD. Fuzzy Prototypical Knowledge Discovery. Application to Forest Fire Prediction. *Proceedings of the SEKE'2000*, Knowledge Systems Institute, Chicago, Ill. USA, 47 – 54.
- Olivas J. A. (2000). Contribution to the Experimental Study of the Prediction based on Fuzzy Deformable Categories, PhD Thesis, University of Castilla-La Mancha, Spain.
- Perry D., Porte A. and Votta L. (2000). Empirical Studies of Software Engineering: A Roadmap. *Future of Software Engineering*. Ed. Anthony Finkelstein, ACM, 345-355.
- 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.
- 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.
- 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.
- Schneidewind N. (1992). Methodology For Validating Software Metrics. *IEEE Transactions of Software Engineering*, 18(5), 410-422.
- Van Solingen R. and Berghout E. (1999). *The Goal/Question/Metric Method: A practical guide for quality improvement of software development.* McGraw-Hill.
- Wohlin C., Runeson P., Höst M., Ohlson M., Regnell B. and Wesslén A. (2000) *Experimentation in Software Engineering: An Introduction.* Kluwer Academic Publishers.
- Zadeh L. (1982). A note on prototype set theory and fuzzy sets. *Cognition* 12, 291 – 297.
- Zuse H. (1998). *A Framework of Software Measurement.* Berlin, Walter de Gruyter.

Appendix A

The following table shows a summary of the data collected in the experiment explained in section 3. The first column shows the class diagram number, the following eleven columns show the metrics values, and the last one shows the mean of the subject's rating of understandability, analysability and modifiability. Attached to some class diagram numbers appears a letter. The diagrams which have the same letter mean that they have 100% similarity.

Class diagram number	NC	NA	NM	NAssoc	NAgg	NDep	NGen	NAggH	NGenH	Max Hagg	Max DIT	Understandability	Analisability	Modifiability
D0	2	4	8	1	0	0	0	0	0	0	0	1	1	1
D1 (A)	3	6	12	1	1	0	0	1	0	1	0	2	2	2
D2 (A)	4	9	15	1	2	0	0	1	0	2	0	2	2	2
D3 (A)	3	7	12	3	0	0	0	0	0	0	0	2	2	2
D4 (A)	5	14	21	1	3	0	0	2	0	2	0	2	2	2
D5	3	6	12	2	0	0	0	0	0	0	0	2	2	2
D6	4	8	12	3	0	1	0	0	0	0	0	2	3	3
D7 (B)	6	10	14	2	2	0	2	1	1	2	1	3	3	3
D8 (A)	3	9	12	1	0	1	0	0	0	0	0	2	2	2
D9 (B)	7	14	20	2	3	0	2	1	1	2	1	3	3	3
D10 (B)	9	18	26	2	3	0	4	1	2	3	1	3	3	3
D11 (B)	7	18	37	3	3	0	2	1	1	3	1	3	3	3
D12 (B)	8	22	35	3	2	1	2	1	1	2	1	3	3	3
D13 (A)	5	9	26	0	0	0	4	0	1	0	2	2	2	2
D14	8	12	30	0	0	0	10	0	1	0	3	2	3	3
D15 (C)	11	17	38	0	0	0	18	0	1	0	4	4	4	4
D16	20	42	76	10	6	2	10	2	3	2	2	6	6	6
D17 (D)	23	41	88	10	6	2	16	2	3	4	3	6	6	6
D18 (E)	21	45	94	6	6	1	20	2	2	4	4	6	5	6

D19	29	56	98	12	7	3	24	3	4	4	4	6	6	7
D20 (B)	9	28	47	1	5	0	2	2	1	4	1	3	3	3
D21 (F)	18	30	65	3	5	0	19	1	2	3	4	5	5	5
D22 (D)	26	44	79	11	6	0	21	2	5	4	3	6	6	6
D23 (F)	17	32	69	1	5	0	19	1	1	2	5	5	5	5
D24	23	50	73	9	7	2	11	3	4	4	1	5	6	5
D25 (E)	22	42	84	14	4	4	16	2	3	2	3	6	5	6
D26	14	34	77	4	9	0	7	2	2	3	4	4	5	5
D27(C)	17	34	47	6	6	0	11	3	2	2	2	4	4	4