EMPIRICAL VALIDATION OF METRICS FOR UML STATECHART DIAGRAMS

David Miranda, Marcela Genero and Mario Piattini
ALARCOS Research Group
Department of Computer Science, University of Castilla - La Mancha,
Paseo de la Universidad, 4, 13071 - Ciudad Real, Spain
Email: dmiranda@proyectos.inf-cr.uclm.es, {Marcela.Genero, Mario.Piattini}@uclm.es

Keywords: Object Oriented Software Systems maintainability, UML behavioural diagrams, UML statechart diagrams, Structural complexity metrics, Empirical validation, Experiment replica

Abstract: It is widely recognised that the quality of Object Oriented Software Systems (OOSS) must be assessed from the early stages of their development. OO Conceptual models are key artifacts produced at these early phases, which cover not only static aspects but also dynamic aspects. Therefore, focusing on quality aspects of conceptual models could contribute to produce better quality OOSS. While quality aspects of structural diagrams, such as class diagrams, have being widely researched, the quality of behavioural diagrams such as statechart diagrams have been neglected. This fact leaded us to define a set of metrics for measuring their structural complexity. In order to gather empirical evidence that the structural complexity of statechart diagrams are related with their understandability we carried out a controlled experiment in a previous work. The aim of this paper is to present a replication of that experiment. The findings obtained in the replication corroborate the results of the first experiment in the sense that at some extent, the number of transitions, the number of states and the number of activities influence statechart diagrams understandability.

1 INTRODUCTION

Nowadays the idea that "measuring quality is the key to developing high-quality OO software" is gaining relevance (Schneidewind, 2002), and it is widely recognised that the quality of OOSS must be assessed from the early stages of their development. Conceptual modelling has become a key task of those early stages because it provides the solid foundation for OOSS design and implementation. The proof of this is that most of approaches towards OOSS development, like OMT, Catalysis, Rational Unified Process, OPEN, etc. have considered conceptual modelling as an integral part.

As early available, key analysis artifacts the quality of the conceptual models are crucial to the success of OOSS development. Therefore, the quality of conceptual models could have a significant impact on the quality of OOSS, which is ultimately implemented and must be evaluated, and if needed improved.

Modelling OOSS with the Unified Modeling Language (UML) (OMG, 1999) the static aspects at conceptual level are mainly represented in structural diagrams such as class diagrams, whilst dynamic aspects are represented in behavioural diagrams such as statechart diagrams, activity diagrams, sequence diagrams and collaboration diagrams. In this work we only focus on quality aspects of statechart diagrams.

Maintainability has become one of the software product quality characteristics (ISO 9126, 2001) that software development organisations are more concerned about since it is the major resource consumer of the whole software life cycle. But we are aware that maintainability is an "external quality attribute" that can only be evaluated once the product is finished or nearly finished. Therefore, it is necessary to have early indicators of such qualities based, for example, on the structural properties of statechart diagrams (Briand et al., 1999), such as their structural complexity.

Cant et al. (1995) have proposed a general cognitive model of software complexity that elaborates on the impact of structure on understandability. In the core of this model is a human memory model and there is some belief within the software engineering community that this model is a reasonable point of departure for

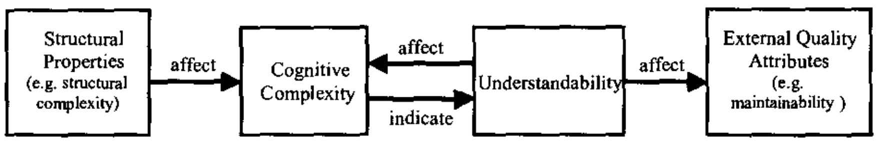
Structural Properties (e.g. structural complexity)
affect
Cognitive Complexity
affect
indicate
Understandability
affect
External Quality Attributes (e.g. maintainability)

Figure 1: The complexity model for system development artifacts of (Briand et al., 1998).

understanding the impact of structural properties on understandability. The theoretical basis for developing quantitative models relating structural complexity and external quality attributes has been provided by (Briand et al., 1998). It is the basis for much empirical research in the area of software artifact structural properties (El-Emam and Melo, 1999; El-Emam, 2001; Poels and Dedene, 2000b). We assume in this work a similar representation to hold for statechart diagrams. We implement the relationship between the structural complexity on the one hand, and external quality attributes on the other hand (see figure 1). We hypothesized that the structural properties (such as structural complexity) of a UML statechart diagram have an impact on its cognitive complexity. By cognitive complexity we mean the mental burden of the persons who have to deal with the artifact (e.g. developers, testers, maintainers). High cognitive complexity leads to an artifact reduce their understandability and this conduce undesirable external qualities, such as decreased maintainability.

To assess the structural complexity of statechart diagrams in an objective and quantitative way it is necessary to dispose of metrics, avoiding thus bias in the quality evaluation process.

After having thoroughly reviewed existing works about metrics, which measure quality aspects, for UML diagrams we found several works related to metrics for structural diagrams such as class diagrams (Brito e Abreu and Carapuça, 1994; Chidamber and Kemerer, 1994; Fenton and Neil, 2000; Genero, 2002; Lorenz and Kidd, 1994; Marchesi, 1998). However, there is little reference to metrics for behavioural diagrams such as statechart diagrams in the existing literature. One of the first approaches towards the definition of metrics for behavioural diagrams can be found in (Derr, 1995), where metrics were applied to statechart diagrams developed with OMT (Rumbaugh et al., 1991). Yacoub et al. (1998) proposed structural complexity and coupling metrics for measuring the quality of dynamic executions. Metrics were defined basing in concepts as Petri Net and McCabe's cyclomatic structural complexity and were applied to simulated scenarios in Real-Time Object Modelling (ROOM) (Selic et al., 1994). Poels and Dedene (2000b) defined structural complexity metrics for event-driven OO conceptual models using MERODE (Snoeck, 1995). These proposals of metrics have not gone beyond the definition step. As far as we know, there is no published works related to their theoretical an empirical validation (except Poels and Dedene who performed the theoretical validation). Therefore, and as was pointed out in (Brito e Abreu et al., 1999, 2000, 2002; Poels and Dedene, 2000b) the definition of metrics for diagrams that capture dynamics aspects of OOSS it is an interesting topic for future investigation. This fact motivated us to define metrics for behavioural diagrams, starting with metrics for measuring the structural complexity of UML statechart diagrams.

According some suggestions about "how to define valid metrics" (Briand et al., 1997; Calero et al., 2001) we have followed a process consisting of three main tasks: metric definition, theoretical validation and empirical validation.

For the definition and the theoretical validation of the metrics (Genero et al., 2002) we followed the DISTANCE framework (Poels and Dedene, 1999, 2000a) which assures the theoretical validity of the defined metrics, i.e., that they measure the attribute they intend to measure. By means of the usage of the DISTANCE framework we could also assure that the proposed metrics are characterised by the ratio scale type, which as Zuse (1998) pointed out it is a main concern when defining metrics for software artifacts.

Empirical validation is crucial for the success of any software measurement project (Basili et al., 1999; Fenton and Pfleeger, 1997; Kitchenham et al., 1995; Schneidewind, 1992). A proposal of metrics has not value if their practical use is not demonstrated empirically, either by means of case studies taken from real projects or by controlled experiments. Therefore, our main motivation is to investigate, through experimentation, if the metrics we proposed in (Genero et al., 2002) for UML statechart diagram structural complexity are related to statechart diagrams understandability. We performed a previous controlled experiment (Genero et al., 2002), pursuing a similar objective. However simple studies rarely provide definite answers and it is also necessary to accumulate the material of many studies (Basili et al., 1999; Miller, 2000), to obtain stronger findings. Following this purpose the main objective of this paper is to show how we carried out a replication of that previous experiment.

Table 1: Metrics for UML statechart diagram structural complexity.

Metric Name	**Metric definition**
NUMBER OF ENTRY ACTIONS (NEntryA)	The total number of entry actions, i.e. the actions performed each time a state is entered.
NUMBER OF EXIT ACTIONS (NExitA)	The total number of exit actions, i.e. the actions performed each time a state is left.
NUMBER OF ACTIVITIES (NA)	The total number of activities (do/activity) in the statechart diagram.
NUMBER OF STATES (NS)	The total number of simple states, considering also the simple states within the composites states.
NUMBER OF TRANSITIONS (NT)	The total number of transitions, considering common transitions (the source and the target states are different), the initial and final transitions, self-transitions (the source and the target state is the same), internal transitions (transitions inside a state that responds to an event but without leaving the state).

This paper is organized in the following way: Section 2 presents a proposal of metrics for the structural complexity of UML statechart diagrams.

A replication of a previous controlled experiment for empirically validating the proposed metrics is presented in section 3. The comparison of the results of the previous experiment with this replication is presented in Section 4. Finally section 5 shows the conclusions and some lines, which are still open for further investigation.

2 A PROPOSAL OF METRICS FOR UML STATECHART DIAGRAMS STRUCTURAL COMPLEXITY

The statechart diagrams structural complexity is determined by the different elements that compose it, such as states, transitions, activities, etc. As Fenton (1994) points, it is not advisable to define a single metric for the structural complexity of UML statechart diagrams since a single metric of structural complexity cannot capture all possible aspects or viewpoints of structural complexity. So several metrics were defined (Genero et al., 2002), each one focusing on a different statechart diagram element (see table 1).

A relevant property of the proposed metrics is that they are simple and ease to automate, which are, as Fenton and Neil (2000) remark in a paper related to the future of software metrics, desirables properties for software metrics.

3 REPLICATION OF A CONTROLLED EXPERIMENT

In this section we describe a replication of an experiment we have carried out to empirically validate the proposed metrics as early understandability indicators (Genero et al., 2002). We have followed some suggestions provided in (Briand et al., 1999; Kitchenham et al., 2002; Perry et al., 2000; Wohlin et al., 2000) on how to perform controlled experiments and have used (with only minor changes) the format proposed by Wohlin et al. (2000) to describe it.

Analyse	*UML statechart diagrams structural complexity metrics*
For the purpose of	*Evaluating*
With respect to	*The capability of being used as understandability indicators of the UML statechart diagrams*
From the point of view of	*OOSS designers*
In the context of	*Undergraduate students in the third year of Computer Science in the University of Castilla-La Mancha*

Table 2: Goal of the experiment.

	NEntryA	NexitA	NA	NS	NT	Mean of the understandability time (seconds)
D01	1	1	0	3	7	129.8
D02	1	0	3	4	7	124.4
D03	2	0	2	4	7	261.7
D04	0	0	2	4	11	195.4
D05	3	2	2	4	13	129.4
D06	6	6	0	6	13	134.8
D07	1	0	1	5	11	169.6
D08	1	0	3	5	13	135.8
D09	0	1	4	5	10	155.4
D10	2	1	0	4	6	108.9
D11	1	2	1	6	17	185.7
D12	1	1	1	3	5	115.3
D13	2	1	0	2	4	93.3
D14	1	1	2	3	8	128.5
D15	1	0	4	9	13	181.3
D16	0	0	5	9	24	162.9
D17	2	0	1	5	8	149.4
D18	2	0	1	12	24	166.5
D19	0	1	0	2	6	108.9
D20	0	0	0	5	12	99.5

Table 3: Metric values for each UML statechart diagram.

3.1 Definition

Using the Goal-Question-Metric (GQM) template (Basili and Weiss, 1984; Basili and Rombach, 1988) for goal definition, the goal of the experiment is shown in table 2.

3.2 Planning

After the definition of the experiments (why the experiment is conducted), the planning took place. The planning is preparation for how the experiment is conducted and includes the following activities:

- **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 in an industrial software development environment).
  The subjects were twenty students enrolled in the third-year of Computer Science at the Department of Computer Science in the University of Castilla-La Mancha in Spain.
  The experiment is specific since it focuses on UML statechart diagram structural complexity metrics. The ability to generalise from this specific context is further elaborated below when we discuss threats to the experiment. The experiment addresses a real problem, i.e., which indicators can be used to assess the understandability of UML statechart diagram? To this end it investigates the correlation between metrics and understandability.
- **Selection of subjects.** The subjects are chosen for convenience, i.e. the subjects are students that have medium experience in the design and development of OOSS.
- **Variables selection.** The independent variable is UML statechart diagram structural complexity. The dependent variable is UML statechart diagram understandability.
- **Instrumentation.** The objects used in the experiment were 20 UML statechart diagrams. The independent variable was measured by the metrics presented in section 2. The dependent variable was measured by the time the subject spent answering the questionnaire attached to each diagram. We called that time "understandability time".
- **Hypothesis formulation.** An important aspect of experiments is to know and to state in a clear and formal way what we intend to evaluate in the experiment. This leads us to the formulation of the following hypotheses:
  Null hypothesis, H_0: There is no significant correlation between the UML statechart diagrams structural complexity metrics and the understandability time.
  Alternative hypothesis, H_1: There is a significant correlation between the UML statechart

diagrams structural complexity metrics and the understandability time.

- **Experiment design.** We selected a within-subject design experiment, i.e. all the questionnaires had to be solved by each of the subjects. The subjects were given the tests in different order.

3.3 Operation

It is in this phase where measurements are collected including the following activities:

- **Preparation.** At the time the experiment was done all of the students had taken a course on Software Engineering, in which they learnt in depth how to design OOSS using UML. Moreover, the subjects were given an intensive training session before the experiment took placing. However, the subjects were not aware of what aspects we intended to study. Neither they were informed about the actual hypotheses stated.
  We prepared the material we handed to the subjects, which consisted of a guide explaining the UML statechart notation and 20 UML statechart diagrams. These diagrams were related to different universes of discourse that were easy enough to be understood by each of the subjects. The structural complexity of each diagram is different, covering a broad range of the metrics values (see table 3). Each diagram had a test enclosed, which includes a questionnaire in order to evaluate if the subjects really understand the content of the UML statechart diagrams. Each questionnaire contained exactly the same number of questions (four) and the questions were conceptually similar and were written in identical order. Each subject had to write down the time he started answering the questionnaire and at the time they finished. The difference between the two is what we called the understandability time (expressed in seconds).
- **Execution.** The subjects were given all of the material described in the previous paragraph. We explained to them how to carry out the experiment. Each subject had to carry out the test alone and there was a teacher supervising the execution of the experiment. We collected all of the data with the understandability time calculated from the responses of the experiments.
- **Data Validation.** Once the data were collected, we controlled if the tests were completed and if the questions have been answered correctly. All the tests were considered valid because all the questions were correctly answered.

3.4 Analysis and Interpretation

First we summarized the data collected for each diagram. We had the metric values and we calculated the mean of the subjects' understandability time for each statechart diagram (see table 3). We want to analyse these data in order to test the hypotheses formulated in section 3.2. For this purpose we used the Statistical Package for Social Science (SPSS).

We applied the Kolmogrov-Smirnov test to ascertain if the distribution of the data collected was normal. As the data were non-normal we decided to use a non-parametric test like Spearman's correlation coefficient, with a level of significance $\alpha = 0.05$, which means the level of confidence is 95% (i.e. the probability that we reject H_0 when H_0 is false is at least 95%, which is statistically acceptable). Each of the metrics was correlated separately to the mean of the subjects' understandability time (see table 4).

For a sample size of 20 (mean values for each diagram) and $\alpha = 0.05$, the Spearman cut-off for accepting H_0 is 0.44 (Briand et al., 1997; CUHK, 2002). Because the computed Spearman's correlation coefficients for metrics NA, NS and NT (see table 4), are above this cut-off and the p-value < 0.05, the null hypothesis H_0, is rejected. Hence, we can conclude that there is a significant correlation between NA, NS and NT metrics and subjects' understandability time.

3.5 Validity evaluation

We will discuss the various issues that threaten the validity of the empirical study and the way we attempted to alleviate them:

- **Threats to Conclusion Validity.** The only issue that could affect the statistical validity of this study is the size of the sample data (20 values), which perhaps are not enough for both parametric and non-parametric statistic test

	NEntryA	NExitA	NA	NS	NT
Understandability time	- 0.046 p=0.85	- 0.346 p=0.14	**0.517** **p=0.02**	**0.575** **p=0.01**	**0.550** **p=0.01**

Table 4: Spearman's correlation between metrics and understandability time.

(Briand et al., 1997). We are aware of this, so we will try to obtain bigger sample data through more experimentation.

- **Threats to Construct Validity.** The construct validity is the degree to which the independent and the dependent variables are accurately measured by the measurement instruments used in the experiment. For the dependent variable we use the understandability time, i.e., the time each subject spent answering the questions related to each diagram, that it is considered the time they need to understand it. It is an objetive measure so we consider the understandability time could be considered a measure constructively valid. The construct validity of the metrics used for the independent variable is guaranteed by Poels and Dedene's framework (Poels and Dedene, 1999, 2000a), used for their theoretical validation (Genero et al., 2002).
- **Threats to Internal Validity.** The internal validity is the degree of confidence in a cause-effect relationship between factors of interest and the observed results. The analysis performed here is correlational in nature. We have demonstrated that several of the metrics investigated had a statistically and practically significant relationship with understandability. Such statistical relationship do not demonstrate per se a causal relationship. They only provide empirical evidence of it. Only controlled experiments, where the metrics would be varied in a controlled manner and all other factors would be held constant, could really demonstrate causality. However, such a controlled experiment would be difficult to run since varying structural complexity in a system, while preserving its functionality, is difficult in practice. On the other hand, it is difficult to imagine what could be alternative explanations for our results besides a relationship between structural complexity and understandability. The following issues have also been dealt with: Differences among subjects, Knowledge of the universe of discourse among class diagrams, Precision in the time values, Learning effects, Fatigue effects, Persistence effects, Subject motivation, Plagiarism and Influence between students.
- **Threats to External Validity.** External validity is the degree to which the research results can be generalised to the population under study and other research settings. The greater the external validity, the more the results of an empirical study can be generalised to actual software engineering practice. Two threats to validity have been identified which limit the ability to apply any such generalisation: Materials and tasks used and Selection of subjects.

3.6 Presentation and Package

As the diffusion of experimental data is important for the external replication of the experiments (Brooks et al., 1996) we have put all of the material of this experiment onto the website http://alarcos.inf-cr.uclm.es.

4 COMPARISON WITH THE RESULTS OF A PREVIOUS EXPERIMENT

The most important difference between the previous experiment we carried out and this replication is the subjects who participate in them. In the first experiment the subjects were teachers and advanced students in the final year of computer science; whilst the subjects that performed this replication were third-year students.

Comparing the findings of both experiments (see tables 4 and 5) we realized that they are similar. This means that the metrics NA, NS and NT are to some extent correlated with the understandability time of UML statechart diagrams.

Nevertheless, despite the encouraging results obtained we still consider them as preliminaries. Further replication, both internal and external (Brooks et al., 1996), is of course necessary and also new experiments must be carried out with practitioners who work in software development organizations. Only after performing a family of experiments you can build an adequate body of knowledge to extract useful measurement conclusions regarding the use of OO design metrics to be applied in real measurement projects (Basili et al., 1999; Briand et al., 1998).

To improve our empirical studies it is necessary to increase the difference between the values of the metrics. This option could lead to more conclusive results about the metrics and their relationship with the factor we are trying to control.

Another way to enhance the validity of the

	NEntryA	NExitA	NA	NS	NT
Understandability time	0.215 p=0.45	0.285 p=0.28	**0.483** **p=0.03**	**0.500** **p=0.02**	**0.605** **p=0**

Table 5: Spearman's correlation between metrics and understandability time in the previous experiment.

results is by working with data obtained from industrial environments, for gathering real evidence that the metrics we presented can be used as early statechart diagram understandability indicators. However, the scarcity of such data continues to be a great problem so we must find other ways to tackle validating metrics. Brito e Abreu et al. (1999, 2000, 2002) suggested the necessity of a public repository of measurement experiences, which we think would be a good step towards the success of all the work done on software measurement. It will possible to do that when more "real data" on systems developed using UML is available, which is the challenge of most of the researchers in this area.

5 CONCLUSIONS AND FUTURE WORK

With the hypothesis that the structural complexity of UML statechart diagrams may influence their understandability (and therefore in their maintainability), we defined a set of metrics for the structural complexity of UML statechart diagrams (Genero et al., 2002). This hypothesis was empirically corroborated by a controlled experiment we carried out before. In order to emphasise these results we carried out a replication, which was thoroughly described in the current work.

As a result of all the experimental work, we can conclude that the metrics NA, NS and NT seem to be highly correlated with the understandability of UML statechart diagrams. Despite of this, we are aware that further empirical validation of these metrics would be necessary to assess if they could be really used as early understandability indicators of the UML statechart diagrams.

Once we obtained stronger results in this line, we think the metrics we proposed could also be used for allowing OOSS designers a quantitative comparison of design alternatives, and therefore, an objective selection among several statechart diagram alternatives with equivalent semantic content, and predicting external quality characteristic, like understandability in the initial stages of the OOSS life cycle and a better resource allocation based on these predictions. In this sense we plan to build a prediction model (based on the metrics values) using advanced techniques borrowed from artificial intelligence such as Fuzzy Classification and Regression Trees (Delgado et al., 2001) and Fuzzy Prototypical Knowledge Discovery (Olivas and Romero, 2000), which have been used for prediction purposes in others domains obtaining accurate predictions.

Finally, another research line of interest would be to evaluate the influence of the structural complexity of the UML statechart diagrams on other maintainability factors such as modifiability and analysability.

ACKNOWLEDGEMENTS

David Miranda is supported by a FPI grant. This research is part of the DOLMEN project supported by Subdirección General de Proyectos de Investigación - Ministerio de Ciencia y Tecnología (TIC 2000-1673-C06-06).

REFERENCES

Basili, V. and Rombach, H., 1988. The TAME project: towards improvement-oriented software environments. In *IEEE Transactions on Software Engineering 14(6) 758-773.*

Basili, V. and Weiss, D., 1984. A Methodology for Collecting Valid Software Engineering Data. In *IEEE Transactions on Software Engineering 10(6) 728-738.*

Briand, L., Arisholm, S., Counsell, F., Houdek, F. and Thévenod-Fosse, P., 1999. Empirical Studies of Object-Oriented Artifacts, Methods, and Processes: State of the Art and Future Directions. In *Empirical Software Engineering 4(4) 387-404.*

Briand, L., Bunse, C., Daly, J. and Differding, C., 1997. An Experimental Comparison of the Maintainability of Object-Oriented and Structured Design Documents. In *Empirical Software Engineering 2(3).*

Briand., L., Wüst, J. and Lounis, H., 1998. A Comprehensive Investigation of Quality Factors in Object-Oriented Designs: an Industrial Case Study. In *Technical Report ISERN-98-29, International Software Engineering Research Network.*

Brito e Abreu, F. and Carapuça, R., 1994. Object-Oriented Software Engineering: Measuring and controlling the development process. In *4th International Conference on Software Quality.*

Brito e Abreu, F., Henderson-Sellers, B., Piattini, M., Poels, G. and Sahraoui, H., 2002. In *Quantitative Approaches in Object-Oriented Software Engineering. ECOOP'01 Workshop Reader,* LNCS Vol. 2323, Springer-Verlag 174-183.

Brito e Abreu, F., Poels, G., Sahraoui, H. and Zuse, H., 2000. In *Quantitative Approaches in Object-Oriented Software Engineering. ECOOP'2000 Workshop Reader,* LNCS Vol. 1964, Springer-Verlag 93-103.

Brito e Abreu, F., Zuse, H., Sahraoui, H. and Melo, W., 1999. In *Quantitative Approaches in Object-Oriented*

Software Engineering. ECOOP'99 Workshops Reader, LNCS Vol. 1743, Springer-Verlag 326-337.

Brooks, A., Daly, J., Miller, J., Roper, M. and Wood, M., 1996. Replication of experimental results in software engineering. In *Technical report ISERN-96-10, International Software Engineering Research Network.*

Calero, C., Piattini, M. and Genero, M., 2001. Empirical validation of referential integrity metrics. In *Information and Software Technology 43 949-957.*

Cant, S., Jeffery, R. and Henderson-Sellers, B., 1995. A Conceptual Model of Cognitive Complexity of Elements of the Programming Process. In *Information and Software Technology 7 351-362.*

Chidamber, S. and Kemerer, C., 1994. A Metrics Suite for Object Oriented Design. In *IEEE Transactions on Software Engineering 20(6) 476-493.*

CUHK - Chinese University of Hong Kong - Department of Obstetrics and Gynaecology – http://department.obg.cuhk.edu.hk/ResearchSupport/Minimum_correlation.asp (Last visited on July 22nd, 2002).

Delgado, M., Gómez-Skarmeta, A. and Jiménez, L., 2001. A regression methodology to induce a fuzzy model. In *International Journal of Intelligent Systems 16(2) 169-190.*

Derr, K., 1995. *Applying OMT,SIGS Books*, Prentice Hall. New York.

El-Emam, K. and Melo, W., 1999. The Prediction of Faulty Classes Using Object-Oriented Design Metrics. In *NRC/ERB1064*, National Research Council. Canada.

El-Emam, K., 2001. Object-Oriented Metrics: A Review on Theory and Practice. In *NRC/ERB 1085*, National Research Council. Canada.

Fenton, N. and Neil, M., 2000. *Software Metrics: a Roadmap. Future of Software Engineering,* Anthony Finkelstein Ed., ACM 359-370.

Fenton, N. and Pfleeger, S., 1997. *Software Metrics: A Rigorous Approach*, Chapman & Hall. London, 2nd. edition.

Fenton, N., 1994. Software Measurement: A Necessary Scientific Basis. In *IEEE Transactions on Software Engineering 20(3) 199-206.*

Genero, M., 2002: Defining and Validating Metrics for Conceptual Model, *Ph.D. thesis*. University of Castilla-La Mancha.

Genero, M., Miranda, D. and Piattini, M., 2002. Defining and Validating Metrics for UML Statechart Diagrams. In 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'2002) 120-136.

ISO 9126, 2001. Software Product Evaluation-Quality Characteristics and Guidelines for their Use, ISO/IEC Standard 9126. Geneva.

Kitchenham, B., Pflegger, S. and Fenton, N., 1995 Towards a Framework for Software Measurement Validation. In *IEEE Transactions of Software Engineering 21(12) 929-943.*

Kitchenham, B., Pflegger, S., Pickard, L., Jones, P., Hoaglin, D., El-Emam, K. and Rosenberg, J., 2002. Preliminary Guidelines for Empirical Research in Software Engineering. In *IEEE Transactions of Software Engineering 28(8) 721-734.*

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. In 2nd Euromicro Conference on Software Maintenance and Reengineering 67-73.

Miller, J., 2000. Applying Meta-Analytical Procedures to Software Engineering Experiments. In *Journal of Systems and Software Vol. 54 29-39.*

Object Management Group, 1999. UML Revision Task Force, OMG Unified Modeling Language Specification, v. 1.3, document ad/99-06-08.

Olivas, J. and Romero, F., 2000. FPKD. Fuzzy Prototypical Knowledge Discovery. Application to Forest Fire Prediction. In *SEKE'2000, Knowledge Systems Institute, Chicago, Ill. USA 47-54.*

Perry, D., Porter, A. and Votta, L., 2000. Empirical Studies of Software Engineering: A Roadmap. In *Future of Software Engineering. Anthony Finkelstein Ed., ACM 345-355.*

Poels, G. and Dedene, G., 1999. DISTANCE: A Framework for Software Measure Construction. In *Research Report DTEW9937, Dept. Applied Economics, Katholiek Universiteit Leuven, Belgium.*

Poels, G. and Dedene, G., 2000a. Distance-Based software measurement: necessary and sufficient properties for software measures. In *Information and Software Technology 42(1) 35-46.*

Poels, G. and Dedene, G., 2000b. Measures for Assessing Dynamic Complexity Aspects of Object-Oriented Conceptual Schemes. In *19th International Conference on Conceptual Modelling (ER 2000) 499-512.*

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W., 1991. *Object- Oriented Modelling and Design,* Prentice Hall. USA.

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

Schneidewind, N., 2002. Body of Knowledge for Software Quality Measurement. In *IEEE Computer 35(2) 77-83.*

Selic, B., Gullekson, G. and Ward P., 1994. *Real-Time Object Oriented Modelling*, John Wiley & Sons, Inc.

Snoeck, M., 1995: On a process algebra approach for the construction and analysis of M.E.R.O.D.E. - based conceptual models, *Ph.D.* Katholieke Universiteit Leuven.

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

Yacoub, S., Ammar, H. and Robinson, T., 1998. Dynamic Metrics for Object Oriented Designs. In *Sixth IEEE International Symposium on Software Metrics*.

Zuse, H., 1998. A Framework of Software Measurement, Walter de Gruyter. Berlin.