SAM 2004

Mario Piattini and
Manuel Serrano(Eds.)

Software Audit and Metrics

**Proceedings of the
1st International Workshop on
Software Audit and Metrics,
SAM 2004**
In conjunction with ICEIS 2004
Porto, Portugal, April 2004

Manuel Serrano and
Mario Piattini (Eds.)

Software Audits and Metrics

Proceedings of the
1st International Workshop on
Software Audits and Metrics,
SAM 2004
In conjunction with ICEIS 2004
Porto, Portugal, April 2004

INSTICC PRESS
Portugal

Volume Editors

Manuel Serrano
Mario Piattini
University of Castilla-La Mancha
Spain.

Proceedings of the 1st International Workshop on Software Audits and Metrics – (SAM 2004)
Porto, Portugal, April 2004.
Manuel Serrano and
Mario Piattini (Eds.)

Printed in Portugal

ISBN 972-8865-04-X

Foreword

These proceedings include the papers accepted for the First International Workshop on Software Audit and Metrics (SAM´2004), which was held in Porto, Portugal, on April 13th 2004.

Modern societies demand high quality products and services, which rely more and more on software. Software quality has become a key issue for software development organizations, which try to improve software processes and products. Audits and metrics could help to ensure that software meets all the client requirements and needs, and to find anomalies as early and as inexpensively as possible. Audit and metrics also provide visibility into the various areas in the organization, which is necessary to understand and improve software processes.

The workshop intends to provide a forum for researchers and practitioners working on approaches, methods, techniques, guidelines, and tools for auditing, measuring, predicting, evaluating, controlling, assuring and improving the quality of software processes and products.

Submission were invited, but not limited, to the following topics: Standards and procedures for software audits, Software audit automatic support, Quality metrics, Standards for the collection and comparison of metrics, Validation of software metrics, Testing and inspection metrics, Metrics and CASE tools, Assessment of software processes, Software reliability, Estimation models for software development, Certification of software processes and products, and Training and qualification in audit and metrics.

We would like to thank all members of the Program Committee and the reviewers for their work in reviewing the papers that appear in this volume. We appreciated the outstanding invited talk: "Quantifying the Unquantified: How quantification of key performance attributes can improve project audit, process audit and project management", by Tom Gilb. Special thanks go to the steering and organizing committees of the ICEIS, especially to Joaquim Filipe and Vitor Pedrosa.

April 2004

Manuel Serrano
Mario Piattini

Workshop Chairs

Manuel Serrano
Mario Piattini
University of Castilla-La Mancha
Spain.

Program Committee

Fernando Brito e Abreu, Universidade Nova de Lisboa (Portugal)
Coral Calero, University of Castilla-La Mancha (Spain)
Giovanni Cantone, University of Rome (Italy)
Marcela Genero, University of Castilla-La Mancha (Spain)
Elixabete Ostolaza, European Software Institute
Geert Poels, Ghent University (Belgium)
Houari Sahraoui, Université de Montréal (Canada)
Jeffrey Voas, Cigital (USA)

Additional Reviewers

Yann-Gaël Guéheneuc, Université de Montréal (Canada)
Félix García, University of Castilla-La Mancha (Spain)

Table of Contents

Quantifying the Unquantified

How quantification of key performance attributes can improve project audit, process audit and project management.

Tom Gilb
Tom@Gilb.com

Abstract. 'Scales of measure' are fundamental to the definition of all scalar system attributes; that is, to all the performance attributes (such as reliability, usability and adaptability), and to all the resource attributes (such as financial budget and time). A defined scale of measure, allows you to numerically quantify such attributes.

'Scales of measure' form a central part of Planguage, a specification language and set of methods, which I have developed over many years.

This paper describes how you can develop your own tailored scales of measure for the specific system attributes, which are important to your organization or system. You cannot rely on being 'given the answer' about how to quantify. You will lose control over your current vital system performance concerns if you cannot, or do not, quantify your critical attributes. Better quantification is obviously a key to better analysis and management of project audit, process audit, and project management.

Scales of Measure and Meters

Scales of measure (Scales) are essential to *quantify* system attributes. The 'system' in question can be a project, a development or life-cycle process, or a product. A Scale specifies an operational definition of 'what' is being measured and it states the units of measure. All estimates or measurements are made with reference to the Scale.

The practical ability to *measure* where you are on a Scale (that is to be able to establish the numeric level) is also important. A Meter (sometimes known as a 'Test') is a practical method for measuring. A Scale can have several Meters.

Finding and Developing Scales of Measure and Meters

The basic advice for identifying and developing scales of measure (Scales) and meters (Meters) for scalar attributes is as follows:

1. Try to re-use previously defined Scales and Meters.

2. Try to modify previously defined Scales and Meters.

3. If no existing Scale or Meter can be reused or modified, use common sense to develop innovative, homegrown quantification ideas.

4 Conclusion

Semantic metrics provide a mechanism for assessing the quality of software in the design or implementation phases. While assessing software in the implementation phase is important, being able to assess earlier provides added value in the ability to correct mistakes or potential problems earlier in the software development lifecycle, when changes are less expensive to make. Whereas most of the previous semantic metrics required a conceptual graph-based knowledge base, the proposed set of metrics can be calculated using any knowledge base that associates classes with ideas. Thus, the proposed set of metrics can be computed using a broader range of knowledge bases than could be used with previously existing semantic metrics.

6 Acknowledgements

The research in this paper was partially supported by NASA grants NAG5-12725 and NCC8-200.

References

1. Etzkorn, L., Delugach, H.: Towards a Semantic Metrics Suite for Object-Oriented Design. Proceedings of the 34th International Conference on Technology of Object-Oriented Languages and Systems (2000) 71-80.
2. Etzkorn, L., Gholston, S., Hughes, W.: A Semantic Entropy Metric. Journal of Software Maintenance and Evolution, Vol. 14, No. 4 (July/August 2002) 293-310.
3. Curtis, B., Carleton, A.: Seven ± Two Software Measurement Conundrums. Proceedings of the 2nd International Metrics Symposium (1994) 96-105.
4. Park, R.: Software Size Measurement: A framework for Counting Source Statements. Technical Report SEI-92-TR-20. Software Engineering Institute, Pittsburgh (1992) 136-137.
5. Kitchenham, B., Pfleeger, S., Fenton, N.: Towards a Framework for Software Measurement Validation. IEEE Transactions on Software Engineering, Vol. 21, No. 12 (Dec. 1995) 929-944.
6. Briand, L., Morasca, S., Basili, V.: Property-Based Software Engineering Measurement. IEEE Transactions on Software Engineering, Vol. 22, No. 1 (Jan. 1996) 68-86.

Validating Metrics for OCL Expressions Expressed within UML/OCL models

Luis Reynoso[1], Marcela Genero[2] and Mario Piattini[2]

[1] Department of Computer Science, University of Comahue,
Buenos Aires 1400, 8300, Neuquén, Argentina.
lreynoso@uncoma.edu.ar
[2] Department of Computer Science, University of Castilla-La Mancha.
Paseo de la Universidad, 4, 13071, Ciudad Real, Spain.
{Marcela.Genero,Mario.Piattini}@uclm.es

Abstract. Measuring quality is the key to developing high-quality software, and it is widely acknowledged that quality assurance of software products must be guaranteed from the early stages of development, assessing through metrics the quality of early models such as UML diagrams. There exists several proposals of metrics to UML diagrams, such as class diagrams, use case diagrams, etc. But, even though the incorporation of OCL to UML diagrams improves software quality and software correctness, there are no metrics for OCL expressions. In a previous work we have defined and theoretically validated a set of metrics that can be applied to OCL expressions expressed within UML/OCL combined models. The main goal of this paper is to show how we carried out a controlled experiment to ascertain the empirical validity of the proposed metrics as early indicators of OCL expressions understandability and modifiability.

1 Introduction

The huge amount of metrics existing in the literature that can be applied to Unified Modelling Language (UML) [21] diagrams [1], [8], [15] reveals a great effort for improving software quality from early stages of their development. Most of the existing studies are focused on the measurement of internal quality attributes of UML diagrams, such as structural complexity, coupling, size, etc. However, none of the proposed metrics take into account the added complexity involved when diagrams are complemented by expressions written in the Object Constraint Language (OCL) [20], that is a UML/OCL combined model. OCL was defined as a textual add-on to the UML diagrams. Its main elements are OCL expressions that represent declarative and side effect-free textual descriptions that are associated to different features of UML diagrams [16]. OCL expressions add precision to UML models beyond the capabilities of the graphical diagrams of UML [23], [25]. Moreover, OCL is essential in building consistent and coherent platform-independent models (PIM) and helping to raise the level of maturity of the software process [25].

Having in mind the importance of OCL in software development, in a previous work we have defined and theoretically validated a set of metrics that can be applied

to a OCL expression within UML/OCL models. We have started our definition focusing on a particular UML diagram: the class diagram, because it constitutes "the most important diagram in the model" [25], since many other diagrams are structured and developed around it. A new effort looking forward the improvement of class diagram quality is our definition of OCL metrics attached to UML class diagrams we published in [22]. In that work we have defined a set of metrics for measuring structural properties of OCL expressions. These metrics were also theoretically validated according to the Briand et al.'s framework [5], [6], [7]. But as many authors remarked [2], [14], [18] the practical utility of the metrics must be demonstrated to empirical validation in order the metrics can be accepted in the software engineering field. For that reason, the main goal of this paper is to present a controlled experiment we carried out in order to ascertain if our metrics could be used as indicators of OCL expressions understandability and modifiability.

In relation to our aim we start in the following section describing how we have defined a set of metrics for OCL expressions whilst all the proper information related to the controlled experiment is presented in section 3. Finally, in section 4 some conclusions are drawn and future work is outlined.

2. A proposal of metrics for OCL expressions

Briand and Wüst [8] have been the mentors in providing the theoretical basis for developing quantitative models relating to structural properties and external quality attributes [17]. Their theory hypothesizes that the structural properties of a software component have an impact on its cognitive complexity [13]. In this work we assume that a similar representation holds for OCL expressions. We implement the relationship between the structural properties on one hand, and external quality attributes on the other hand [17]. We hypothesize that the structural properties of an OCL expression have an impact on its cognitive complexity[1]. High cognitive complexity leads to a reduction in the understandability of an artifact –in this case, the OCL expressions-, and provokes undesirable external qualities, such as decreased maintainability.

For defining the metrics in a disciplined manner we have applied a method for metric definition based on [9] and [12], which will allow us to obtain valid and reliable metrics.

Moreover, for defining the metrics we have considered the two following issues:

- Structural properties of OCL expressions: In order to analyze the structural properties of an OCL expression we have considered the OCL concepts described in the OCL metamodel [20].
- Cognitive aspects. Ideally, we should also be able to explain the influence of the values of the metrics from a cognitive point of view. Cant et al. [10]; [11] argue in their Cognitive Complexity Model (CCM model) that measuring complexity should affect attributes of human comprehension since complexity is relative to human cognitive characteristics. Therefore we have considered the cognitive techniques applied by modelers when they try to understand an OCL expression (considered in our study as a single mental abstraction: a chunk). These techniques are: "chunking" and "tracing" [10]; [11], [13]. We have defined metrics related to these cognitive techniques.

[1] By cognitive complexity we mean the mental burden of the persons who have to deal with the artifact (e.g. modelers, designers, maintainers) [13].

Fenton [14] suggested that it is not advisable to define a single measure for capturing different structural properties. For that reason we have defined a set of metrics, each of which captures different structural properties of an OCL expression, related to the cognitive techniques of the CCM model, such as the "tracing" and "chunking" techniques. These techniques are concurrently and synergistically applied in problem solving [19]. "Chunking" involves the recognition of a set of declaration and extracting information from them, which is remembered as a chunk, whereas "tracing" involves scanning, either forward or backwards, in order to identify relevant "chunks". The whole set of metrics defined for each group can be found in [22].

As chunking and tracing techniques are concurrently applied, we cannot plan an experiment considering only a set of OCL metrics related to only one cognitive technique. We have selected, for the empirical validation, some metrics that are related to those OCL concepts, which are more commonly used in simple OCL expressions.

- Metrics related to "chunking": NKW (Number of OCL Keywords), NES (Number of Explicit Self), NBO "Chunking" (Number of Boolean Operators), NCO (Number of Comparison Operators), NEI (Number of Explicit Iterator variables), NAS (Number of Attributes belonging to the classifier that Self represents).
- Metrics related to "tracing": NNR (Number of Navigated Relationships), NAN (Number of Attributes referred through Navigations), NNC (Number of Navigated Classes), WNN (Weighted Number of Navigations), DN (Depth of Navigations), WNCO (Weighted Number of Collection Operations).

3. A controlled experiment

In this section we describe a controlled experiment we have carried out to empirically validate the proposed measures as early OCL expressions understandability indicators. To describe the experiment we use (with only minor changes) the format proposed by Wohlin et al. [26].

3.1 Definition

Using the GQM [2], [24] template for goal definition, the goal pursued in this experiment is: Analyze <<Metrics for OCL expressions within UML/OCL models>>; for the purpose of <<Evaluating>>; with respect to <<The capability to be used as understandability and modifiability indicators of OCL expressions >>; from the point of view of <<OO Software modellers>>: in the context of <<Undergraduate Computer Science enrolled in a course related to OCL, of the Department of Computer Science at the National University of Comahue>>.

3.2 Planning

After the definition of the experiment, the planning phase took place. It prepares for *how* the experiment is conducted, including the following six steps:

Context selection. The context of the experiment is a group of undergraduate students who had agreed to take part in a course on OCL, and hence the experiment is run off-line (not in an industrial software development environment). The subjects were twenty-nine students enrolled in the third and fourth-year of Computer Science at the Department of Computer Science at the National University of Comahue in Argentina.

The experiment is specific since it is focused on twelve metrics for OCL expression within UML/OCL combined models. The experiment addresses a real problem, i.e., which indicators can be used for the understandability and modifiability of OCL expressions? With this end in view it investigates the relationship between metrics and the time spent in understandability and modifiability tasks.

Selection of subjects. According to [26] we have applied a probability sampling technique: a convenience sampling. The nearest persons we could choose were undergraduate students who had, in average, one year of experience in the development of OO systems using UML, and by the time the experiment took place they were taking a course of OCL.

Variable selection. The independent variable is the structural complexity of OCL expressions. The dependent variables are the understandability and modifiability of OCL expressions.

Instrumentation. The objects were four UML/OCL combined models, having each of them one OCL expressions. The independent variable was measured through the selected metrics. The dependent variables were measured according to:

- The time each subject carried out the understandability and modifiability tasks.
- The subjects'ratings of understandability or modifiability.
- We have also used as indicators of understandability and modifiability the following measures proposed in [4]
  - Understandability Correctness = Number of correct answers/ Number of answers., which represents the correctness of the understanding questionnaire, i.e. the number of questions correctly answered. The number of correct answers is a reasonable measure of the understanding since all the tests have the same design, it has the same quantity of questions.
  - Modifiability correctness = Number of correct modifications/ Number of Modifications applied.
  - Modifiability completeness = Number of correct Modifications /Number of modifications required

Hypothesis formulation. We wish to test the following hypotheses (two hypotheses for each measure for the dependent variables)

1) $H_{0,1}$: There is no significant correlation between the OCL metrics and the understandability and modifiability time. // $H_{1,1}$: $\neg H_{0,1}$
2) $H_{0,2}$: There is no significant correlation between the OCL metrics and understandability/modifiability correctness/completeness. // $H_{1,2}$: $\neg H_{0,2}$

3) $H_{0,3}$: There is no significant correlation between the OCL metrics and the subjective understandability/modifiability. // $H_{1,3}$: $\neg H_{0,3}$
4) $H_{0,4}$: There is no significant correlation between the understandability/modifiability time and the subjective understandability/modifiability. // $H_{1,4}$: $\neg H_{0,4}$

Experiment design. We selected a within-subject and balanced design experiment, i.e., all the tests (experimental tasks) had to be solved by each of the subjects. The tests were put in a different order for each subject for alleviating learning effects.

3.3 Operation

The operational phase is divided into three steps: preparation, execution and data validation.

Preparation. We have selected as experimental subjects a group of students who have taken a semester class on System Analysis. In this course the student had learnt the use of UML. The students were motivated to take a course on OCL , they were informed that OCL is an expressive language used for formally expressing additional and necessary information about a model specified in UML. Later, the students were asked to participate in the course, 29 subjects agreed to take part, so they are volunteers. They were motivated to take a training session on OCL language and to do some practical exercises as part of the session, but it was not mentioned that these exercises are constituent of an experiment. The subjects were not aware of what aspects we intended to study. Neither were they aware of the actual hypothesis stated.

Table 1. Metric values for each UML/OCL model

Test	NNR	NAN	WNN	WNCO	NAS	NEI	NCO	NBO	NES	NKW	NNC	DN
1	2	1	2	1	1	0	3	2	4	4	2	1
2	4	2	5	4	0	0	3	4	5	6	4	1
3	3	1	4	3	0	1	2	3	5	5	2	3
4	4	2	2	3	2	0	3	2	4	4	4	3

We prepared the material handed to the subjects, consisting of four UML/OCL models. These diagrams were related to different universes of discourse that were easy enough to be understood by each of the subjects, and some of them were obtained from the existent OCL literature [20]. The structural complexity of each model is different as it is revealed from the metrics values of each UCL/OCL model (see Table 1). Before running the experiment we performed a pilot experiment. We asked a researcher who has experience on OCL to carry out the experimental tasks. All the modifications she suggested were considered.

Each UML/OCL model had a test enclosed that included two types of tasks:

Understandability tasks:

- The subjects had to answer four questions about the meaning of the OCL expression within the UML/OCL model. These questions had the purpose to test if the subjects had understood each expression. The first question was related to naviga-

tions concepts, meanwhile the last three questions were a multiple choice about the meaning of the OCL expressions. Each question has three options, being only one option the correct answer. They also had to note how long it took to answer the questions. The "understandability time", expressed in minutes and seconds, was obtained from that.

- The subjects had to rate the understandability tasks using a scale consisting of five linguistic labels (Very difficult to understand, A bit difficult to understand, Neither difficult nor easy to understand, A bit easy to understand and Very easy to understand). We called this measure "subjective understandability".

Modifiability tasks:

- Each UML/OCL model used by the subjects in the understandability task had also enclosed three new requirements for the OCL expression. Each subject had to modify the OCL expression according to the new requirements. The modifications to each test were similar, including defining new navigations, attributes referred through navigations, etc.
- They also had to write down the time when they started to do the modifications and when they finished them. This time was called "modifiability time".
- We have also used a scale consisting of five linguistic labels similar to the one used for understandability, so that the subject could rate the modifiability tasks. We called this measure "subjective modifiability".

Moreover, we prepared a debriefing questionnaire. This questionnaire included personal details and experience.

Execution. In the lecture before the experiment was carried out, the subjects were asked to bring a watch in the next lecture. Those subjects who did not bring a watch were able to use a clock rendered with a multimedia projector. The subjects were given all the materials described in the previous paragraph. We explained to them how to carry out the test, asking for carrying out the test alone, and using unlimited time to solve it. There was an instructor who supervised the experiment and any doubt could be asked to him. We collected all the data, including subjects' rating obtained from the responses of the experiment.

Data validation. Analyzing the debriefing questionnaire, we can corroborate that the subjects had approximately the same degree of experience in modelling with UML, the profile of the subject is the following: their average age is 24 years old, they have an average of 4 years programming experience, and one year in modelling UML class diagrams. Taking into account their profile, we consider their subjective evaluation reliable.

Most of the answers for the modifiability part of the four tests were not correctly answered, only the understandability part of the four tests had an optimal rate of answers. We think that the reason is that the experiment was carried out after two lectures of 2 hours each, and this period of time was enough for the students to understand OCL expressions but they did not have enough practice in OCL expressions modification. We think we exposed the students prematurely to do OCL expression modification. For that reason we consider in this paper only the understandability part of the experiment. Regarding this part, three tests were studied as outliers and 12 tests were separated because they have a correctness below 75%. Finally, we had 101 data sets to be analyzed.

3.4 Analysis and Interpretation

We had analysed the experiment data in order to test the hypotheses formulated in section 3.2. First we had to check the normality of the data obtained. If the data was normal, the best option in our case was to use parametric tests because they are more efficient than non-parametric tests. Then, we applied the Kolmogorov-Smirnov test to ascertain if the distribution of the data collected was normal. As the data was 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 of at least 95%, which is statistically acceptable). Each of the metrics was correlated separately to the mean of the subjects' understandability time (see Table 2). For a sample size of 101 and $\alpha = 0.05$, the Spearman cut-off for accepting H_0 is 0.1956. Hence, after analyzing table 2, we can conclude that:

- There is a significant correlation between WNN, WNCO, NEI, NCO, NBO, NES, NKW and DN metrics and subjects' understandability time.
- There is a significant correlation between NEI, NCO and NES metrics and correctness and completeness.
- There is a significant correlation between the NEI, NCO and DN metrics and the subjective understandability.

Table 2. Spearman's correlation between metrics and understandability time

		NNR	NAN	WNN	WNCO	NAS	NEI	NCO	NBO	NES	NKW	NNC	DN
Und. Time	Scc	.162	.020	**.207**	.223	-.172	**.348**	**-.348**	**.207**	**.282**	**.207**	.020	**.324**
	p-value	.105	.840	**.038**	**.025**	.086	**.000**	**.000**	**.038**	**.004**	**.038**	.840	**.001**
Corr.	Scc	-.073	.016	-.180	-.151	.173	**-.229**	**.229**	-.180	**-.224**	-.180	.016	**-.147**
	p-value	.465	.877	.071	.131	.084	**.021**	**.021**	.071	**.025**	.071	.877	**.143**
Sub. Und.	Scc	.093	-.026	.084	.108	-.076	**.308**	**-.308**	.084	.166	.084	-.026	**.326**
	p-value	.357	.794	.403	.283	.450	**.002**	**.002**	.403	.097	.403	.794	**.001**

Table 3. Spearman's correlation between metrics and subjective understandability

		Correctness	Subjective Und.
Understandability Time	Scc	-.102	**.349**
	p-value	.310	**.001**

Moreover, after analyzing Table 3 we can conclude that there is a significant correlation between the understandability time and the subjective understandability.

Nevertheless, these encouraging findings must be considered as preliminaries. More experimentation would be necessary in order to obtain more conclusive results.

3.5 Validity evaluation

Next we will discuss the empirical study's various threats to validity 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 which is perhaps not enough for

non-parametric statistic tests. We are aware of this, so we will consider the results of the experiment only as preliminary findings.

Threats to construct validity. We proposed an objective measure for the dependent variable, the understandability time, i.e., the time each subject spent answering the questions related to each UML/OCL model, that it is considered the time they need to understand it. We also proposed subjective metrics for them (using linguistic variables), based on the judgment of the subjects. As the subjects involved in this experiment have medium experience in OO system design using UML we think their ratings could be considered significant. The construct validity of the metrics used for the independent variables is guaranteed by Briand et al.'s framework [5], [6], [7] used to validate them.

Threats to Internal Validity. 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 time, understandability correctness and subjective understandability. Such statistical relationships do not demonstrate per se a causal relationship. They only provide empirical evidence of it. We tried to alleviate some threats: differences among subjects, Knowledge of the universe of discourse among UML/OCL combined models, accuracy of subjects responses, learning effects, fatigue effects, subject motivation, plagiarism, etc.

Threats to external validity. The greater the external validity, the more the results of an empirical study can be generalized to actual software engineering practice. Two threats of validity have been identified which limit the possibility of applying any such generalization:

- Materials and tasks used. In the experiment we have used UML/OCL models, which can be representative of real cases. Related to the tasks, the judgment of the subjects is to some extent subjective, and does not represent a real task.
- Subjects. To solve the difficulty of obtaining professional subjects, we worked with students from software engineering courses. We are aware that more experiments with practitioners and professionals must be carried out in order to be able to generalize these results. However, in this case, the tasks to be performed do not require high levels of industrial experience, so, experiments with students could be appropriate [3].

4. Conclusions and future work

The quality of OO software systems is highly dependent on decisions made early in its development, so many measurement researches have been developed focusing on the quality of artifacts produced at the initial stages of the software systems lifecycle. Important efforts were carried out for measuring UML diagrams. But many design decisions, constraints and essential aspects of software systems cannot be expressed in a UML diagram using only diagrammatic notations. This implies that the metrics defined until now will not be able to capture those design decisions that cannot be expressed using graphical notations. The quality of UML/OCL model should be also considered. A first work in this direction was the definition and theoretical validation of a set of metrics for an OCL expression within UML/OCL diagrams [22].

Although OCL is considered [20] a formal language easy to read and write, the misuse of the language can lead to complicated written OCL expressions. Warmer and Kleppe [25] recognize that the way OCL expressions are defined has a large impact on readability, maintainability and the complexity of the associated diagrams. Since empirical validation of metrics is crucial in defining reliable metrics we have presented in this paper a controlled experiment.

The experiment reveals that there is a strong correlation between the subjective understandability rating and the understandability time. The findings we have obtained are: (1) only the set of metrics composed by WNN (Weighted Number of Navigations), WNCO (Weighted Number of Collection Operations), NEI (Number of Explicit Iterator variables), NCO (Number of Comparison Operators), NBO (Number of Boolean Operators), NES (Number of Explicit *Self*), NKW (Number of OCL Keywords) and DN (Depth of Navigations) is related with the understandability time. (2) A subset of this set of metrics, that is composed of NEI; NCO, NES and DN has an impact on the subjective cognitive understandability of subjects, and, although the rating is subjective we have also corroborated that almost the same subset of metrics (with exception of NES although it has a low p-value) is also correlated to the completeness of the experimental tests, giving the second finding more significance.

These findings should be considered as preliminaries. We are aware that further validation is necessary in order to assess if the presented metrics could be used as early indicators of OCL expressions understandability.

Acknowledgements

This work has been partially funded by the MESSENGER project (PCC-03-003-1) financed by "Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla - La Mancha", the network VII-J-RITOS2 financed by CYTED, and the UNComa 04/E048 Research Group financed by "Subsecretaría de Investigación" of Comahue University. Luis Reynoso enjoys a postgraduate grant from the agreement between the Goverment of Neuquén (Argentina) and YPF-Repsol.

References

1. J. Bansiya and C. G. Davis. A Hierarchical Model for Object-Oriented Design Quality Assessment, IEEE Trans. on Soft. Eng., Vol. 28 N° 1, 2002, 4-17.
2. V. R. Basili, and H. D. Rombach. The TAME project: towards improvement-oriented software environments. IEEE Trans. on Soft. Eng.. Vol. 14 N° 6, 1998, 758-773.
3. V. R. Basili, F. Shull and F. Lanubile. Building knowledge through families of experiments. IEEE Trans. on Soft. Eng., Vol. 25 N° 4, 1999, 456-473.
4. L. C. Briand L., C. Bunse and J. W. Daly. A Controlled Experiment for evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. IEEE Trans. on Soft. Eng., Vol. 27 N° 6, 2001, 513-530.
5. L. C. Briand, S. Morasca, and V. R. Basili. Defining and Validating Measures for Object-based High Level Design. IEEE Trans. on Soft. Eng. Vol. 25 N° 5, 1999, 722-743.

6. C. Briand, S. Morasca, and V. R. Basili. Property-based Soft. Eng. Measurement. IEEE Trans. on Soft. Eng., Vol. 22 N° 1, January 1996, 68-86.
7. L. C. Briand, S. Morasca and V. R. Basili. Response to: Comments on 'Property-Based Soft. Eng. Measurement': Refining the Additivity Properties. IEEE Trans. on Soft. Eng., Vol. 23 N° 3, March 1997, 196-197.
8. L. C. Briand. and J. Wüst. Modeling Development Effort in Object-Oriented Systems Using Design Properties. IEEE Trans. on Soft. Eng., Vol. 27 N° 11, November 2001, 963-986.
9. C. Calero, M. Piattini, and M. Genero. Method for Obtaining Correct Metrics. Proc. of the 3rd Int. Conf. on Enterprise and Information Systems (ICEIS'2001), 2001, 779-784.
10. S. N. Cant, B. Henderson-Sellers, and D. R. Jeffery. Application of Cognitive Complexity Metrics to Object-Oriented Programs. Journal of Object-Oriented Programming, Vol. 7 N° 4, 1994, 52-63.
11. S. N. Cant, D. R. Jeffery and B. Henderson-Seller. A Conceptual Model of Cognitive Complexity of Elements of the Programming Process. Information and Software Technology, Vol. 37 N° 7, 1995, 351-362.
12. G. Cantone, and P. Donzelli. Production and maintenance of software measurement models. Journal of Soft. Eng. and Knowledge Engineering, Vol. 5, 2000, 605-626.
13. K. El-Eman. Object-Oriented Metrics: A Review of Theory and Practice. National Research Council Canada. Institute for Information Technology. March 2001.
14. N. E. Fenton, and S. L. Pfleeger. Software Metrics: A Rigorous and Practical Approach. Chapman & Hall, London, 2nd Edition. International Thomson Publishing Inc. 1997.
15. M. Genero. Defining and Validating Metrics for Conceptual Models, PhD Thesis, University of Castilla-La Mancha. 2002.
16. R. Hennicker, H. Hussmann, and M. Bidoit. On the Precise Meaning of OCL Constraints. Tony Clark and Jos Warmer (eds), Advances in Object Modelling with the OCL, Springer, Berlin, LNCS 2263, 2001, 69-84.
17. ISO/IEC 9126. Software Product Evaluation-Quality Characteristics and Guidelines for their Use. Geneva.
18. B. Kitchenham, S. Pflegger, and N. Fenton. Towards a Framework for Software Measurement Validation. IEEE Transactions of Soft. Eng., Vol. 21 N° 12, 1995, 929-944.
19. T. Klemola. A Cognitive Model for Complexity Metrics. 4th Int. ECOOP Workshop on Quantitative Approaches in OO Soft. Eng.. France, 2000.
20. Object Management Group. UML 2.0 OCL 2nd revised submission. OMG Document ad/2003-01-07. [On-line] Available: http://www.omg.org/cgi-bin/doc?ad/2003-01-07.
21. Object Management Group. UML Specification Version 1.5, OMG Document formal/03-03-01. [On-line] Available: http://www.omg.org/cgi-bin/doc?formal/03-03-01.
22. L. Reynoso, M. Genero and M. Piattini. Measuring OCL Expressions: An approach based on Cognitive Techniques. Piattini M., Genero M. and Calero C. (eds), Imperial College Press, UK (to appear), 2004.
23. M. Richters. A Precise Approach to Validating UML Models and OCL Constraints. Biss Monographs Vol. 14. Gogolla, M., Kreowski, H.J., Krieg-Brückner, B., Peleska, J., Schlingloff, B.H. (eds.). Logos Verlag. Berlin, 2002.
24. R. Van Solingen, and E. Berghout. The Goal/Question/Metric Method: A practical guide for quality improvement of software development. McGraw-Hill, January 1999.
25. J. Warmer and A. Kleppe. The Object Constraint Language. Second Edition. Getting Your Models Ready for MDA. Object Technology Series. Addison-Wesley, Massachusetts, August 2003.
26. C. Wohlin, P. Runeson, M. Höst, M. Ohlson, B. Regnell, and A. Wesslén. Experimentation in Soft. Eng.: An Introduction, Kluwer Academic Publishers, March 2000.

Analysis of Software Measures Using Metrology Concepts – ISO 19761 Case Study

Alain Abran and Asma Sellami

École de technologie supérieure – ETS
Université du Québec, Montréal, Québec, Canada
aabran@ele.etsmtl.ca
asma.sellami.1@ens.etsmtl.ca

Abstract. To help identify the strengths of proposed software measurement methods, this paper proposes an analytical approach based on metrology concepts documented in the ISO International Vocabulary of Basic and General Terms in Metrology. This approach is illustrated with a case study using one specific functional size measurement method recognized as an ISO standard: COSMIC-FFP (ISO 19761). The case study documents the metrology concepts addressed in this ISO standard, either in the design of this measurement method or in some of its practical uses. It illustrates, for instance, that the design of COSMIC-FFP encompasses a large number of related metrology concepts. It is suggested that such a review using metrology criteria be used to analyze other software functional size measurement methods, as well as other software measures suggested to industry.

1 Introduction

Hundreds of software measures have been defined in the software engineering domain and proposed to industry. However, only the following have successfully undergone the rigor of international standardization: the quality measures in the ISO 9126 series [8], and three functional size measurement methods, among them ISO 19761 – COSMIC-FFP [10]. Software functional size measures are used in particular to compare the productivity of software projects (internally or across organizations), for project effort estimation and for the control of functional changes over a project life cycle. The use of such standardized measures is important to ensure comparability of measurement results between projects and between organizations; indeed, it would not be relevant to compare numbers based on distinct (and not standardized) measurement methods. Without the use of standards, ideally those officially recognized internationally, software agreements between customers and suppliers are prone to a variety of interpretations and, often, to conflicts.

A large number of software measures are defined based on the intuition of their authors. When subjected to the scrutiny of researchers, they are often investigated only from the perspective referred to as "measurement theory" (i.e. their mathematical properties) [5, 6, 11]. However, in other science and engineering disciplines, it is the domain of knowledge referred to as "metrology" that is the foundation for the devel-