D. Patel, I. Choudhury, S. Patel
and S. de Cesare (Eds)

2000 OOIS

6th International Conference on Object Oriented Information Systems

Also Available:

OOIS'98
1998 International Conference on Object Oriented Information Systems,
9-11 September 1998, Paris
Edited by Colette Rolland and Georges Grosz

OOIS'97
1997 International Conference on Object Oriented Information Systems,
10-12 November 1997, Brisbane
Edited by Maria E Orlowska and Roberto Zicari

Dilip Patel
School of Computing, Information Systems and Mathematics,
South Bank University, 103 Borough Road, London, UK

Islam Choudhury
School of Computing, London Guildhall University, 100 Minories, London, UK

Shushma Patel
School of Computing, Information Systems and Mathematics,
South Bank University, 103 Borough Road, London, UK

Sergio de Cesare
Department of Information Systems and Computing, Brunel University,
Uxbridge, London, UK

ISBN 1-85233-420-7 Springer-Verlag London Berlin Heidelberg

British Library Cataloguing in Publication Data
A catalog record for this book is available from the British Library.

Library of Congress Cataloging-in-Publication Data
A catalog record for this book is available from the Library of Congress.

Printed in Great Britain

Typesetting: Camera ready by contributors
Printed and bound by the Athenæum press Ltd., Gateshead, Tyne & Wear
34/3830-543210 Printed on acid-free paper SPIN 10790885

Programme Committee

Programme Committee
General Chair

Keith Jeffrey (UK)

Programme Co-Chairs

Dilip Patel (UK)
Islam Choudhury (UK)

Programme Committee

Motoei Azuma (Japan)
Peter Bielkowicz (UK)
Fang Fang Cai (UK)
Francesco Capozza (Italy)
Akmal Chaudri (UK)
Islam Choudhury (UK)
Quan Dang (UK)
Alessandro d'Atri (Italy)
Sergio de Cesare (UK)
Tharam Dillon (Australia)
Mohamed Fayad (USA)
Ian Graham (UK)
Jane Grimson (Ireland)
Tushar Hazra (USA)
Gurdeep Singh Hura (USA)
Brian Henderson-Sellers (Australia)
Keith Jeffrey (UK)
Robin Laney (UK)
Jeff Naylor (UK)
Dilip Patel (UK)
Shushma Patel (UK)
D Janaki Ram (India)
Colette Rolland (France)
Jeff Sutherland (USA)
Yuan Sun (UK)
Bhuvan Unhelkar (Australia)
Yingxu Wang (Sweden)
Alan Wills (UK)
Rebecca Wirf-Brock (USA)
Roberto Zicari (Germany)

Organising Chairs

Sergio de Cesare (UK)
Fang Fang Cai (UK)

Additional Reviewers

Costas Calcanis (UK)
Adil Dhalaboy (UK)
Maria Carmina di Camillo (ITALY
Kay Dudman (UK)
Kevin Lu (UK)
Nazmul Mannan (UK)
George Ubakanma (UK)

XML AND CORBA ISSUES II

Quality is a multidimensional concept, composed of different characteristics such as functionality, reliability, usability, efficiency, maintainability and portability (ISO 9126, 1999). However, the definition of the different characteristics that compose the concept of "quality" is not enough on its own in order to ensure quality in practice as people will generally make different interpretations of the same concept. Software measurement plays and important role in this sense because metrics provide a valuable and objective insight into specific ways of enhancing each of the software quality characteristics. Measurement data can be gathered and analysed using various quality models to assess current product quality, to predict future quality, and to drive quality improvement initiatives (Tian, 1999).

Most the external quality attributes proposed in the ISO 9126 (1999), such as maintainability, reliability, etc. can only be measured late in the OOIS life cycle. So it is necessary to find early indicators of such qualities based, for example, on the structural properties of class diagrams (Briand, 1999b).

Early availability of measures is a key factor in the successful management of software development, since it allows for (Briand et al., 1999a):

1. the early detection of problems in the artifacts produced in the initial phases of the life cycle (specification and design documents) and, therefore, reduction of the cost of change-late identification and correction of the problems are much more costly than early ones;
2. better software quality monitoring from the early phases of the life cycle;
3. quantitative comparison of techniques and empirical refinement of the processes to which there are applied;
4. more accurate planning of resource allocation, based upon the predicted quality of the system and its constitutent parts.

Within the field of software engineering a plethora of metrics have been proposed for measuring OO software products, even though most of them are related on products obtained from advanced design and implementation phases (Chidamber and Kemerer, 1994, Lorenz and Kidd, 1994; Brito e Abreu and Melo, 1996; Henderson-Sellers, 1996). Genero et al. (1999) have proposed some metrics for measuring OMT class diagrams. Few works have been done specifically about measures applied to UML class diagrams (Marchesi, 1998; Genero et al., 2000).

The goal of this work is to propose a set of metrics in order to measure the complexity of UML class diagrams (section 2) focusing specially in the different UML relationships, such as associations, aggregations, generalisations and dependencies. We also put the proposed metrics under empirical validation in order to provide empirical support to their practical significance and usefulness (section 3). Lastly, section 4 summarises the paper, draws our conclusions, and presents future trends in metrics for object modelling using UML.

2. A proposal of metrics for UML class diagrams

In this section we will propose a set of closed-ended metrics (Lethbridge, 1998) for assessing the complexity of UML class diagrams at the initial phase of the OOIS life cycle. A closed-ended metric is where measurements can only fall within a particular range, and where it is impossible for them to fall outside that range (most of our metrics fall in the range [0,1]). As the aim of this work is simplify class diagrams as much as possible, our goal will be minimise the metric values. We consider the worth case value when the metric value tends to 1, and the best case when the metric value tends to 0.

△ in the class diagram

2.5 M_{GH} metric

The goal of Generalisation Hierarchy metric is to evaluate the complexity of class diagrams due to generalisation hierarchies. For this we take into account some of the factors that influence in the hierarchy structure (number of classes, number of levels and the use of multiple inheritance).

We define this metric as follow:

1. If the class diagram has no generalisation hierarchies: $M_{GH}= 0$
2. If the class diagram has generalisation hierarchies, M_{GH} is defined as: $M_{GH} = \sum_{i=1}^{n} CJ_i$, where CJ_i is the complexity of the ith generalisation hierarchy and n is the number of generalisation hierarchy within a class diagram

In order to calculate CJ_i we combine two factors: The first factor is the number of classes that are leaves of the hierarchy. This factor called $FLEAF_i$, is calculated thus: $FLEAF_i = \frac{N_i^{LEAF}}{N_i^C}$, where N_i^{LEAF} is the number of leaf classes in the i^{th} generalisation hierarchy and N_i^C is the number of classes in the i^{th} hierarchy.

Figure 1 shows different generalisation hierarchies with their values of $FLEAF_i$. $FLEAF_i$ approaches to 0.5 when the generalisation hierarchy is a binary tree (figure 1, parts c and d). It approaches zero in the ridiculous case of a unary "tree" with just a single superclass-subclass chain (figure 1, part b). And it approaches one if every superclass is an immediate subclass of the root class (part a).
On its own, $FLEAF_i$ has the undesirable property that for a very shallow hierarchy (e.g. just two or three levels) with a high branching factor it gives a measurement that is unreasonably high, from a subjective standpoint (part a of fig. 1 illustrate this). To correct this problem with $FLEAF_i$, an additional factor is used in the calculation of CJ_i, : the average number of direct and indirect superclasses per non-root class $ALLSUP_i$ (the root class of the generalisation hierarchy is not counted since it cannot have parents). This second factor is related to hierarchy depth but depends to some extent on the amount of multiple inheritance.

CJ_i is thus calculated using the following formula: $CJ_i = FLEAF_i - \frac{FLEAF_i}{ALLSUP_i}$

$AvsC = \left(\frac{N^A}{N^A + N^C} \right)^2$	N^A is the number of attributes in an UML class diagram. N^C is the number of classes in an UML class diagram. Where $N^A + N^C > 0$.

2.8 MEvsC metric

The Methods vs. Classes metric measures the relation that exists between the number of methods and the number of classes in an UML class diagram.

We define this metric as follows:

$MEvsC = \left(\frac{N^{ME}}{N^{ME} + N^C} \right)^2$	N^{ME} is the number of methods in an UML class diagram. N^C is the number of classes in an UML class diagram. Where $N^{ME} + N^C > 0$.

3. Our metrics in practice

The software measures have an important role on management and qualification of OOIS, at the initial phase of their life cycle. Nevertheless, it is well known that the definition of new metrics is not enough to cover this purpose. Another fundamental aspect in this aim rests on the empirical validation and correct interpretation of the observed measures (Fenton, 1994; Zuse, 1998). According to this purpose we have studied the behaviour of the proposed metrics in a set of assets (reusable software elements) stored in the GIRO repository.

The GIRO repository was founding as part of a research work on reuse, and it includes assets coming from different stages of software development and from various software paradigms, basically object oriented and classical (García, 2000).

3.1 Data collection and analysis

For the moment, the supplier of GIRO repository is the membership of GIRO and students from the University of Valladolid, who are developing their final project in Computer Technical Engineering. The asset measured belong to three product lines: 7 of Image (I), 7 of Disabled (D) and 13 of Optic (O). All of them were developed following the OO paradigm and UML, using the RATIONAL ROSE case tool; DELPHY was the implementation language.

To each asset we calculate the proposed metrics. Each asset has those values as part of its quality documentation.

The methodology followed in this research is thus:

1. Extract relevant information from a summary of data.
2. Analyse the data in order to find differences between product lines.
   The null hypotheses to test is: H^1o: *there is no difference between metric X in line1 and metric X in line2*

"-" There are few observations							
	ASvsC	AGvsC	GEvsC	DEPvsC	AvsC	MEvsC	M_{GH}
I/D	**0.0087**	-	0.9350	-	0.2013	0.7983	-
I/O	**0.0383**	**0.0066**	-	-	**0.0383**	0.4273	-
D/O	0.1303	-	-	-	**0.0015**	0.5253	-

Table 2. P-Value (p) of Mann-Whitney Test

Considering Table 2, we can conclude that:

- The ASvsC can differentiate between product lines I and D.
- Furthermore I is different of O, when ASvsC, AGvsC and AvsC are considered.
- The AvsC can differentiate between product lines D and O.

3.4 Relationships among metrics

Every metric considered should quantify a distinct feature of the software products. To achieve this goal they need to be independent from each other, we can express the hypothesis H^2o and H^3o *as:* $r_{xy}= 0$

Tables 3, 4 and 5 summarise the results achieved applying the Spearman correlation test. The cells contain the p-value and the Spearman correlation coefficient. If there are insufficient observations the cell contains "-".

From table 3, we can conclude that MEvsC could have positive correlation with AGvsC(p= 0.0326) and DEPvsC(p=0.0663) in product line I.

From table 4, we have not observed any correlation in product line D, (p > 0.1 in all cases).

From table 5, we can conclude that GEvsC could have negative correlation with MEvsC(p= 0.0154) in the product line O.

When we studied the metric dependency in all the sample (n = 27) we only observed positive correlation between ASvsC and DEPvsC (p =0.0598), the relationships detected into product lines disappeared.

Image	ASvsC	AGvsC	GEvsC	DEPvsC	AvsC	MEvsC
ASvsC		0.7875 -0.1101	0.95815 -0.0263	-	0.7575 7 -01261	0.40187 -0.3424
AGvsC			0.1945 -0.6489	-	0.1089 0.6547	**0.0326** **0.8729**
GEvsC				-	0.4727 0.3591	0.8375 -0.1026
DEPvsC					-	-
AvsC						**0.0663** **0.7500**

Table 3. Matrix of Spearman correlation in product line Image

Disabled	ASvsC	AGvsC	GEvsC	DEPvsC	AvsC	MEvsC
ASvsC		-	-	0.6274 0.1982	0.1154 0.6429	0.7931 0.1071
AGvsC			-	-	-	-
GEvsC				-	-	-
DEPvsC					0.3316 0.3964	0.7575 0.1261
AvsC						0.7931 -0.1071

Table 4 Matrix of Spearman correlation in product line Disabled

Optic	ASvsC	AGvsC	GEvsC	DEPvsC	AvsC	MEvsC
ASvsC		0.2413 0.3354	0.8493 -0.0633	-	0.2156 0.3574	0.1507 0.4148
AGvsC			0.8067 -0.0815	-	0.16193 -0.4038	0.9068 -0.0338
GEvsC				-	0.5828 0.1831	**0.0154** **-0.8074**
DEPvsC					-	-
AvsC						0.7366 -0.0971

Table 5. Matrix of Spearman correlation in product line Optic

In conclusion, the complexity metrics ASvsC, AGvsC and AvsC could have different behaviour by product lines, so if we ignore it we can have confounding (Kleinbaum, 1987). Furthermore, some metrics, such as MEvsC wtih AGvsC, AvsC and GEvsC could be correlated in different product lines, and so we must consider this relationship in future models. We cannot forget that the sample is small, so the conclusions must be corroborated with other larger samples.

4. Conclusions and future work

Due to the growing complexity of OOIS, continuous attention to and assessment of object models are necessary to produce quality software systems. The fact that UML has emerged is a great step forward in object modelling. Even though this does not guarantee the quality of the models produced through the IS life cycle. Therefore, it is necessary to have metrics in order to evaluate their quality from the early phases in the OOIS development process.

In this work we have presented a set of metrics for assessing the complexity of UML class diagrams, obtained at early phases of the OOIS life cycle. It is widely accepted that the greater complex an UML class diagram, the greater complex the OOIS which is finally implemented, and therefore more effort is needed to develop and maintain it. So that the proposed metrics could be very fruitful, because they will allow OOIS designers to assess the complexity of their designs, and compare between design alternatives, from the early phases of OOIS life cycle.

As in other aspects of Software Engineering, proposing techniques and metrics is not enough, validation is critical to the success of software measurement (Kitchenham et al., 1995; Fenton and Pflegeer, 1997; Schneidewind, 1992; Basili et al.,1999). It is also necessary for them under theoretical and empirical validation, in order to assure their utility. Furthermore, it is important to understand their behaviour in order to define suitable models and appropriate design of the experiments (Montgomery, 1991, Kleinbaum et al., 1987), i.e. the quality model changes if we need consider blocks (product lines in this case), or if we have variables which are not independent (MEvsC and AGvsC).

With regard to empirical validation, we are carrying out some experimentation not only with controlled experiments but also with "real" cases taken from some enterprises, with the objective of assessing these metrics as predictors of maintenance efforts, and therefore, determine whether they can be used as early quality indicators.

In future work, we will focus our research on measuring other quality factors like those proposed in the ISO 9126 (1999), which not only tackle class diagrams, but also evaluate other UML diagrams, such as use-case diagrams, state diagrams, etc. To our knowledge, few works have been done towards measuring dynamic and functional models (Derr, 1995; Poels, 1999; Poels 2000). As is quoted in (Brito e Abreu et al., 1999) this is an area which lacks in depth investigation. In addition further empirical validation of the proposed metrics is needed to research their usefulness in the development software process. Furthermore we need a sample bigger to confirm the tracks find in this study and a quality model which includes a set of interest variables.

We will extend our metric, called MANTICA (created to measures data models), in order to provide support for collecting, analysing and visualising metric values applied to UML class diagrams.

Acknowledgements

This research is part of the MANTICA project, partially supported by CICYT and the European Union (1FD97-0168), and MENHIR project, supported by CICYT TIC97-0593-C05-05.

References

(Basili et al., 1999): V. Basili, F. Shull and F. Lanubile. *Building knowledge through families of experiments.* IEEE Transactions on Software Engineering, 25(4), pages 435--437, 1999.

(Briand et al., 1999a): L. Briand, S. Morasca and V. Basili. *Defining and Validating Measures for Object-Based high-level design.* IEEE Transactions on Software Engineering. 25(5), pages 722--743, 1999.

(Briand et al., 1999b): L. Briand, S. Arisholm, F. Counsell, F. Houdek, F. and Thévenod-Fosse. *Empirical Studies of Object-Oriented Artifacts, Methods, and Processes: State of the Art and Future Directions.* Technical Report IESE 037.99/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany, 1999.

(Brito e Abreu et al., 1996): F. Brito e Abreu and W. Melo. *Evaluating the Impact of Object-Oriented Design on Software Quality.* Proceedings of 3rd International Metric Symposium, 1996.

-437, 1999.

(Briand et al., 1999a): L. Briand, S. Morasca and V. Basili. *Defining and Validating Measures for Object-Based high-level design.* IEEE Transactions on Software Engineering. 25(5), pages 722--743, 1999.

(Briand et al., 1999b): L. Briand, S. Arisholm, F. Counsell, F. Houdek, F. and Thévenod-Fosse. *Empirical Studies of Object-Oriented Artifacts, Methods, and Processes: State of the Art and Future Directions.* Technical Report IESE 037.99/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany, 1999.

(Brito e Abreu et al., 1996): F. Brito e Abreu and W. Melo. *Evaluating the Impact of Object-Oriented Design on Software Quality.* Proceedings of 3rd International Metric Symposium, 1996.

(Brito e Abreu et al., 1999): F. Brito e Abreu, H. Zuse, H. Sahraoui W. and Melo. *Quantitative Approaches in Object-Oriented Software Engineering.* Object-Oriented technology: ECOOP´99 Workshop Reader, Lecture Notes in Computer Science 1743, Springer-Verlag, pages 326--337, 1999.

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

(Derr, 1995): K. Derr. *Applying OMT.* SIGS Books, New York, 1995.

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

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

(García, 2000): F. J. García. *Modelo de Reutilización Soportado por Estructuras Complejas de Reutilización Denominadas Mecanos.* PHD Thesis, University of Salamanca, 2000.

(Genero et al., 1999): M. Genero, Mª E. Manso, M. Piattini and F. J. García. *Assessing the Quality and the Complexity of OMT Models.* 2nd European Software Measurement Conference - FESMA 99, Amsterdam,The Netherlands, pages 99—109, 1999.

(Generoet al., 2000): M. Genero, M. Piattini and C. Calero (2000). *Una Propuesta para Medir la Calidad de los Diagramas de Clases en UML.* IDEAS´2000, Cancún,