Quantitative Approaches in Object-Oriented Software Engineering

Fernando Brito e Abreu[1], Horst Zuse[2], Houari Sahraoui[3], and Walcelio Melo[4]

[1] Faculdade de Ciências e Tecnologia / INESC, Portugal
fba@di.fct.unl.pt - http://www.fct.unl.pt
[2] Technische Universitat Berlin, Germany
zuse@cs.tu-berlin.de - http://www.tu-berlin.de
[3] Centre de Recherche Informatique de Montréal , Canada
hsahraou@crim.ca - http://www.crim.ca
[4] Oracle, Brazil
wmelo@acm.org

Abstract. This full-day workshop was organized in four sessions. The first three were thematic technical sessions dedicated to the presentation of the recent research results of participants. Seven, out of eleven accepted submissions were orally presented during these three sessions. The first session also included a metrics collection tool demonstration. The themes of the sessions were, respectively, "Metrics Definition and Collection", "Quality Assessment" and "Metrics Validation". The last session was dedicated to the discussion of a set of topics selected by the participants.

1. Introduction

Quantitative Methods in the Object Oriented (OO) field is an active research area. This workshop aimed to shed some light on recent research results and future directions that might interest not only the academic community but also industry. The latter is eagerly launching software process improvement initiatives but is often insecure on how to assess the corresponding results both at the product and process levels. Measures of software structural attributes have been extensively used to help software managers, customers and users to characterize, assess, and improve the quality of software products. Many large software companies have intensively adopted software measures to increase their understanding of how and how much software product internal attributes affect the overall software quality. Estimation models have successfully been used to perform risk analysis, to assess software maintain-ability and error-proneness, mainly on what we call today "legacy technology". Also, large software companies have taken advantage of software measures to built up useful cost models and delivery schedules. By doing so, they have been able to improve their software development processes by producing realistic deadlines and allocating adequate resources. OO paradigm provides powerful design mechanisms which have not been fully or adequately quantified by the existing software product measures. Much work is yet to be done to investigate analytically or empirically the relationship between OO

A. Moreira and S. Demeyer (Eds.): ECOOP'99 Workshops, LNCS 1743, pp. 326–337, 1999.

design mechanisms, e.g., inheritance, polymorphism, encapsulation, usage etc., and different aspects of software quality, e.g., modularity, modifiability, understandability or extensibility. Emerging concepts and technologies, such as OO frameworks, OO Analysis/Design Patterns, Web technology and Component-based development, take advantage of OO design mechanisms. To understand their pros and cons we must be able to assess the quality of the base technology on which they are built upon.

Several workshops on similar topics have occurred in previous years, such as the following:

- "Object-Oriented Product Metrics for Software Quality Assessment" - ECOOP'98
- "Object-Oriented Design Quality" - OOPSLA'97
- "Object-Oriented Product Metrics" - OOPSLA'96
- "Quantitative Methods for Object-Oriented Systems Development" - ECOOP'95
- "OO Process and Metrics for Effort Estimation" - OOPSLA'95
- "Pragmatic and Theoretical Directions in OO Software Metrics" - OOPSLA'94

This workshop took place on the 15th June 1999 at the Faculty of Sciences of the Lisbon University. The workshop started by the mutual presentation of participants. All organizers except Walcelio Melo attended the workshop. Authors of all submissions selected for oral presentation were present, so no changes had to be operated. The workshop organic details were explained to participants and then the technical sessions begun.

The structure of this report mirrors the actual workshop program layout. The workshop was organized into 3 technical sessions and a roundtable discussion on elected topics. The technical sessions were thematic:

- Session 1 - Metrics Definition and Collection
- Session 2 - Quality Assessment
- Session 3 - Metrics Validation

The complete set of accepted submissions is available online at:
`http://www.esw.inesc.pt/ftp/pub/esw/mood/ecoop99`

2. Technical Presentations

In this section we will present an abstracted version of the presentations done during the three technical sessions.

2.1. Session 1 - Metrics Definition and Collection

This session included three communications and a metrics collection tool demonstration.

2.1.1. Modeling and Measuring Object-Oriented Software Attributes with Proximity Structures (Geert Poels and Guido Dedene)

Geert Poels from the Department of Applied Economic Sciences of the Catholic University of Leuven, in Belgium, presented this communication.

The authors advocated an approach to model software attributes with proximity structures. These are empirical relational structures that describe the concept of dissimilarity or conceptual distance. Measurement theory is used to formally validate software measures as measures of distance.

Their "distance-based" approach was presented as a constructive five-step measurement procedure that hides the complexity of the underlying measurement theoretic constructs from the user. Special attention was paid to an intuitive hierarchy of assumptions on which the constructive procedure is based. They also gave a brief overview of some results regarding the distance-based measurement of object-oriented enterprise models.

The distance-based approach presents an alternative way of modelling and measuring software attributes. On the one hand, the approach has firm measurement theoretic foundations. On the other hand, the complexity of the measurement theoretic constructs involved is largely hidden from the user by means of a constructive, five-step procedure. They paid special attention to the assumptions underlying the approach as they determine its successful use. They found proximity structures especially useful in the context of object-oriented enterprise modelling measurement. They do acknowledge, however, that further experiences with their approach are needed to draw definite conclusions about its usefulness for (object-oriented) software measurement in general. Further research must also focus on the relations between alternative models of software attributes (e.g., proximity structures versus belief structures), their underlying assumptions, and their implications regarding uniqueness and meaningfulness of measurement scales.

2.1.2. The GOODLY Design Language for MOOD2 Metrics Collection (Fernando Brito e Abreu, Luis Ochoa, and Miguel Goulão)

Fernando Brito e Abreu from the Faculty of Sciences and Technology of the Lisbon New University and INESC, both in Portugal, presented the GOODLY language that is being used by his team in the production of a new generation of the MOODKIT tools that allow the MOOD2 metrics extraction. The MOOD2 design metrics set is an extension of the original MOOD set. A detailed definition of the MOOD2 set is available as an INESC internal report. The GOODLY language, whose syntax and semantics were briefly described, allows specifying the design of systems built according to the Object Oriented paradigm. This language, whose features are fully described in the paper submitted to this workshop, allows expressing the most relevant OO design information, such as the class structure with corresponding inheritance relations, class parameterization, uses relationships, message exchanges and information hiding. The presentation

also described the architectural evolution of MOODKIT, along with its rationale. It was evident how the GOODLY design language played a fundamental role in this effort.

Miguel Goulão then made a presentation of the MOODKIT G2 tool. This tool, in its current version, allows parsing Eiffel and Smalltalk code (C++ and Java are also sought), as well as object models expressed in OMT and UML (using the ParadigmPlus tool), to produce GOODLY code. The tool has a linking facility for identification of missing components and besides generating the MOOD2 metrics, it also produces a hypertext version of GOODLY code (HT-GOODLY) with traceability features. Miguel showed how a large software system specification expressed in GOODLY could be navigated with a normal browser supporting frames. The software industry interest on the MOOD metrics set was evidenced by the availability of several tools that support the collection of this metric set, such as Cantata++, a tool for C++ and Ada95 projects (`http://www.iplbath.com/tools/`) and Krakatau, a tool for Java and C++ projects (`http://www.power-soft.co.uk/english/kr/`).

Fernando ended the presentation with a call for cooperation to other research teams in the OO metrics area. His team is specially interested in finding partners with process data that can be used along with OO product metrics for building validation experiments. The web site of the MOOD project can be used for browsing a large set of systems expressed in HT-GOODLY, to see the original code from which they were generated or to get papers produced within the MOOD team. (`http://www.esw.inesc.pt/ftp/pub/esw/mood`)

2.1.3. Software Metrics and Object-Oriented Systems (Horst Zuse)

Next presentation, after the morning coffee break, was from Horst Zuse of the Technical University of Berlin, Germany.

He focused on the validation of software measures in order to predict an external variable. Such external variables can be costs of maintenance, time to repair a module, etc. Validation of software measures is a very important task, but not an easy one. Mostly, correlation coefficients or regression analysis for the validation of software measures are used. He pointed out that measures for the objected-oriented paradigm and imperative languages are very differently related to the prediction of maintenance effort.

2.1.4. Demo of a Tool for OO Metrics Collection (Brian Henderson-Sellers)

Brian Henderson-Sellers from the School of Computing Sciences of the University of Technology in Sydney, Australia, presented a tool for OO metrics collection. This tool, named "IT 903 Metrics Tool" was developed by a group of five students (Peter Bonifacio, John Cain, Mimi Cheong, Goran Duspara and Tracey Richards) from the Master of Information Technology at Swinburne University of Technology, Australia. The available user manual indicated that the last version was 1.1 and its revision date was from November 1998.

According to the authors this tool runs on Windows 95, requires a 80386 processor or above, 8 Mbytes of RAM, 10 Mb of hard disk space and a standard file archiving utility such as, for example, WinZip, for installation purposes. The tool uses activeX technology for displaying charts. The tool's main features are:

- extraction of around 30 distinct OO metrics from source code in Java
- comparison among projects or different versions of the same project
- metrics collection at four distinct abstraction levels: method, class, package (module / subsystem) or system
- definition of two warning levels for each metric; these levels are used, for instance, to select values to display
- metrics display in several graphical display formats (area charts, horizontal and vertical bars, line, mark, pie and fit to curve charts)
- several report options
- metrics data export facility

The tool had an appealing interface and seemed to be user friendly. The origin of the supported metrics was not identified, although, among others, we could distinguish the set proposed by Chidamber and Kemerer from the Sloan School of Management at the MIT. Brian informed the audience that to try this tool one should contact their authors, since it is the Swinburne University of Technology policy that students keep the copyright of their academic assignments' outcome.

2.2. Session 2 - Quality Assessment

After the lunch break we restarted the workshop with session 2, which included two communications.

2.2.1. Towards the Measurement of Reuse by Inheritance in Legacy Systems (Radu Marinescu)

Next presentation was done by Radu Marinescu from the Faculty of Automatics and Computer Science / "Politehnica" University of Timisoara, Romenia.

He argued that the possibility of using object-oriented metrics to support the re-engineering of legacy systems is still put under question. Radu believes that the cause of this fact does not lie in an intrinsic incapacity of metrics to help in re-engineering, but in applying metrics that are inadequate for this purpose. The use of metrics that take full advantage of the information available for a legacy system, instead of using simple, high-level design metrics, increases the chances of successfully applying metrics to re-engineering. The paper presents guidelines for defining such metrics and proposes a multi-layered system of metrics that measures the real reuse of a base class in its descendants. It also reports the first results and conclusions of applying these metrics on three legacy systems. The author believes that the future use of such systems of metrics will offer a systematic and flexible manner of dealing with metrics, increasing the scope of their usability.

2.2.2. Using Metrics for Refactoring (Houari A. Sahraoui, T. Miceli, and R. Godin)

Houari A. Sahraoui from the Computer Science Center of Montreal (CRIM) and the University of Montreal in Canada, presented the following communication.

The authors proposed a technique that aims at detecting and correcting design flaws. This technique uses quality estimation models which are based on the correlation between quality characteristics (e.g. maintainability) and quantitative attributes of software (metrics) and software transformations. The idea behind this work is to relate potential transformations with symptomatic situations. To do that, a four-step

process is proposed. First, a set of transformations that can be applied to improve the quality of a system is chosen. Then, a set of metrics is selected under the basis that they can be good indicators of design anomalies. Third, a study of the impact of the transformations on the metrics in term of variation is done. Finally rules are designed to correct the anomalies using these variations.

The speaker mentioned that the approach was applied to C++ classes. In the majority of the cases, the suggested transformations were adopted. The authors recognize that even if the first results were very satisfactory, the limited number of the studied transformations does not allow to measure in a precise way the impact of the technique. The attendees agreed that further experiences are needed to draw a definite conclusion.

2.3. Session 3 - Metrics Validation

The last technical session, with two additional papers, started after the afternoon coffee break.

2.3.1. An Hypothesis-Based Evaluation of an Object-Oriented Inheritance Metric (Steve Counsell, P. Newson, and K. Mannock)

Next speaker was Steve Counsell from the Department of Computer Science at the Birkbeck College, University of London in the United Kingdom.

He started by observing that various Object-Oriented (OO) metrics have been proposed to capture features of a systems' inheritance hierarchy. Some of these metrics appear more useful than others. In this communication the authors used a single inheritance metric - the Number Of Descendants per class (NOD) to highlight a relationship with the number of friends found in five C++ Systems. Classes containing friends in each of the five systems were found to have considerably less descendants than other classes, indicating that to obtain the benefits of extra functionality, friends will tend to be found in classes deep in an inheritance hierarchy. Their analysis highlights the importance of choosing an inheritance metric that allows hypotheses of these sort to be tested. As a result they compared the applicability of the NOD metric with two other frequently used inheritance metrics.

2.3.2. Validation of Metrics for Object-Relational Databases (Coral Calero, Mario Piattini, Francisco Ruiz, and Macario Polo)

The last speaker was Mario Piattini from the ALARCOS Group at E.S. Informática / U.C.L.M., Ciudad Real in Spain. He presented a work in the databases area which, he argued, the metrics community has neglected. Nowadays a new generation of object-relational databases is coming out. These database systems will have a big impact. They propose a set of metrics for object-relational database systems, characterizing them in a well known formal metric framework.

3. Discussion

The participants selected cooperatively a set of topics for discussion. The selection process included a voting process based on individual scientific interest. The discussion around those topics then took place, ordered by decreasing votes obtained. The topics discussed and the ideas, or even questions, that emerged about them, were the following:

Many New Metrics Continue to be Proposed Every Year in the Literature. Isn't It Time to Start Reducing the Set?

- Horst Zuse observed that more than three hundred OO metrics have already been proposed.
- Reducing sets implies that authors reach an agreement. Under which assumptions?
- Most proposed metrics are static! Don't we need dynamic ones?

Which Are the Most Relevant Obstacles to Metrics Validation?

- Scarce availability of process data.
- No public repository (this rises the question of how could it be managed so that some non-disclosure problems could be withdrawn).
- Need for standard data sets with different recognized variations.

How Can We Use Existing Metrics to Predict Software Quality Attributes?

- As pointed out in the talk of Horst Zuse, the prediction of external variables, like effort, is still a difficult task. The calculation of correlation or the use of regression analysis is only one way to do it and it is not sufficient.
- Statistical based validations with significant samples must be carried on.
- A quality model for software is needed here. The ISO9126 model can be a good starting point. However, there has been a very slow progress within the

ISO JTC1/SC7 towards choosing appropriate metrics to quantify the proposed quality characteristics. Horst Zuse claimed that the reference model itself took 6 years to be published.

What about Metrics in the OO Databases Field?

- We need to define new metrics for Object-Relational Databases.
- Can existing OO metrics be applied to OO database schemes?
- We need to consider both the schema level (DDL) and the manipulation one (DML)

How to "Sell" Metrics to Industry?

- Industry must be convinced that every euro/dollar spent on quality has a return.
- We need product metrics standards, even if not "perfect"!
- Lack of validation is still an obstacle.
- Software industry is reasonably well convinced to use metrics to increase software quality.
- Introduction of a metrics programme is the most difficult task.

Which "Niches" for Research?

- Bridging design-patterns (qualitative) with OO metrics (quantitative) research work.
- Metrics for Use-cases
- Metrics for components technology
- Metrics for dynamic OO design modelling

4. Other Accepted Submissions

The following papers were accepted but not presented orally:

Representation of High Level Models for Reuse (Soliman M. Algeri and Ian Finch)

This paper presents the reuse engineering paradigm of the conceptual framework for reuse(CFR). The framework consists of reuse management and reuse engineering. The reuse engineering paradigm describes a chain of activity that addresses reuse-related software development. The reuse engineering paradigm

consists of three processes: model creation, model management, and model utilization. Reuse initially started at the low level (code reuse), but current research is interested in the broader picture of a high level reuse (design and requirement). In the mean time, attention has been paid to the model creation process of high level models (such as use cases and design patterns) and less on the model management and model utilization processes. Their research interest is focused on the model management process. This management process consists of three processes: acquisition, representation, and retrieval of the domain models. The work presented in their paper focused on the representation of high level models such as use cases and design patterns. The objective was to facilitate the production of a complete system design (a design instance) by integrating both models. The use case model and design patterns have been chosen because both are in line with the object-oriented paradigm.

Measuring the Impact of Migration to an Object Oriented Paradigm (David John Leigh, Colin J. Theaker, Neil Blackwood, and Robert Mason)

This paper describes the practical application of metric measurements to an evolving project. The initial implementation in a C-based non OO environment has been assessed using a number of techniques. The migration to a Java-based OO implementation is then discussed.

Towards Techniques for Improved OO Design Inspections (Forrest Shull, Guilherme H. Travassos, and Victor Basili)

This paper is about OO Design Inspections. Software inspections aim to guarantee that a particular software artifact is complete, consistent, unambiguous, and correct enough to effectively support further system development. For instance, inspections have been used to improve the quality of a systems design and code. Typically, inspections require individuals to review a particular artifact, then meet as a team to discuss and record defects, which are then sent to the document's author to be corrected. The position paper discussed some issues regarding the definition and application of reading techniques that can be used to read requirements, use-cases and design artifacts within a domain in order to identify defects among them.

Combining Metrics, Principles, and Guidelines for Object Oriented Design Complexity Reduction (Guilherme H. Travassos and Renata da Silva Andrade)

In their paper, Guilherme Travassos and Renata Andrade argue that the need for large and more complex software systems is ever increasing. The object-oriented paradigm is one of the options that developers have to develop complex

software systems, providing them with flexibility to decide about software design and architecture. But, associated to OO software construction, structural complexity is still an issue that must be addressed. This position paper describes results of a work that identified object-oriented standards, guidelines and metrics that can be combined to help developers to observe and reduce OO design structural complexity.

Annex. Participants List with Contacts

Geert Poels (Geert.Poels@econ.kuleuven.ac.be)
Department of Applied Economic Sciences, Katholieke Universiteit Leuven, Belgium.

Guido Dedene (Guido.Dedene@econ.kuleuven.ac.be)
Department of Applied Economic Sciences, Katholieke Universiteit Leuven, Belgium.

Radu Marinescu (radum@cs.utt.ro / marinesc@fzi.de)
Faculty of Automatics and Computer Science, "Politehnica" University of Timisoara, Romenia.

Coral Calero (ccalero@inf-cr.uclm.es)
Grupo ALARCOS. E.S. Informática. U.C.L.M., Ciudad Real, Spain.

Mario Piattini (mpiattin@inf-cr.uclm.es)
Grupo ALARCOS. E.S. Informática. U.C.L.M., Ciudad Real, Spain.

Francisco Ruiz (fruiz@inf-cr.uclm.es)
Grupo ALARCOS. E.S. Informática. U.C.L.M., Ciudad Real, Spain.

Macario Polo (mpolo@inf-cr.uclm.es)
Grupo ALARCOS. E.S. Informática. U.C.L.M., Ciudad Real, Spain.

Horst Zuse (zuse@tubvm.cs.tu-berlin.de)
Technische Universitat Berlin, Germany.

Houari A. Sahraoui (hsahraou@crim.ca)
Centre de Recherche de Informatique du Montréal (CRIM)
University of Québec, Canada.

T. Miceli
University of Québec, Canada.

R. Godin
University of Québec, Canada.

Fernando Brito e Abreu (fba@di.fct.unl.pt / fba@inesc.pt)
Faculdade de Ciências e Tecnologia / Universidade Nova de Lisboa, Portugal.
INESC, Portugal.

Luis Ochoa (luis.ochoa@link.pt)
Link, Portugal.

Miguel Goulão (miguel.goulao@inesc.pt)
Faculdade de Ciências e Tecnologia / Universidade Nova de Lisboa, Portugal.
INESC, Portugal.

Steve Counsell (steve@dcs.bbk.ac.uk)
Department of Computer Science, Birkbeck College, University of London, United Kingdom.

P. Newson
Department of Computer Science, Birkbeck College, University of London, United Kingdom.

K. Mannock
Department of Computer Science, Birkbeck College, University of London, United Kingdom.

Soliman M. Algeri (algeri@csc.liv.ac.uk)
Department of Computer Science, University of Liverpool, United Kingdom.

Ian Finch (ian@connect.org.uk)
Department of Computer Science, University of Liverpool, United Kingdom

David John Leigh (D.J.Leigh@soc.staffs.ac.uk)
Staffordshire University, Stafford, United Kingdom

Colin J Theaker (C.J.Theaker@soc.staffs.ac.uk)
Staffordshire University, Stafford, United Kingdom

Neil Blackwood (nb@terrafix.co.uk)
Terrafix Limited, Stoke-on-Trent, United Kingdom

Robert Mason (R.Mason@terrafix.co.uk)
Terrafix Limited, Stoke-on-Trent, United Kingdom

Forrest Shull (fshull@cs.umd.edu)
Experimental Software Engineering Group, Department of Computer Science, University of Maryland, USA

Guilherme H. Travassos (travasso@cs.umd.edu / ght@cos.ufrj.br)
Experimental Software Engineering Group, Department of Computer Science, University of Maryland, USA
Computer Science and System Engineering Department, Federal University of Rio de Janeiro, Brasil

Victor Basili (basili@cs.umd.edu)
Experimental Software Engineering Group, Department of Computer Science, University of Maryland, USA

Renata da Silva Andrade (randrade@cos.ufrj.br)
Computer Science and System Engineering Department, Federal University of Rio de Janeiro, Brasil