OOPSLA '92
Vancouver, British Columbia, Canada
5 – 10 October 1992

Addendum to the Proceedings

Workshop Report— Metrics for Object-Oriented Software Development

Reported by:

Teri Roberts

Organizers:

Sam Adams
Knowledge Systems Corporation

Teri Roberts
Object International, Inc.

Rajiv Tewari
Temple University

Student Volunteer:

Ralph Jackson

1. Introduction

Use of the object-oriented paradigm for software solutions has gained momentum and popularity. The metrics selection activity has been incubating and is ready to hatch. We are interested in nurturing the growth and use of a variety of metrics for the object-oriented paradigm. Described herein are the activities and results of the first OOPSLA workshop on metrics. Interest and participation will grow as the object-oriented paradigm matures through it's practical application.

2. Workshop Objectives, Goals, and Format

The objective, as stated in the advance program, was to propose metrics for estimating cost and schedule, for evaluating productivity of object-oriented techniques and to gauge improvement in software quality and programmer productivity. In spite of limited response we forged ahead with a goal of reaching some level of consensus about useful metrics based on a format of presenting and discussing participant experiences.

After the welcoming "administrivia" Teri Roberts suggested a product metrics versus process metrics point of view. By the end of the workshop we agreed that we had a better handle on product metrics than on process metrics. Contact the individual participants (see Section 8) for a copy of their presentation materials.

3. Recent Directions

Sam Adams started the day by sharing his angle on metrics derived from recent work. He proposed 4 measurement levels: method, class, component/application and project management. The first 3 belong to the product camp, while the last is in the process camp. He discussed 33 method metrics, combined the class and component levels for a total of 13 metrics, and presented 8 project management metrics. He also cited 4 issues to consider for metric usage.

4. Participant Presentation Summaries

David Tegarden's work is focused on structural complexity and perceptual complexity. He is trying to map perception to structure. Both of these aspects are based on the system complexity, which he sees as part problem complexity and part technique complexity.

He introduced 4 levels of product metrics: system, object, variable and method. At each level he identified a set of graph-theoretic and lexically-based measures. He showed a 3 dimensional measurement framework where these metric levels are mapped against analysis, design, programming, and maintenance activities and against generalization, aggregation, association, inheritance, and message passing relationships.

This was followed by a cognitive map of perceptions of object-oriented system complexity derived from a pilot study involving 7 graduate students who were asked, "What contributes to the complexity of an object-oriented system?". The students proposed 10 categories which were rated for importance in controlling complexity. Four of them (class design, structure, method design, and message passing) were mapped back to the structural complexity aspect using the 4 levels (system, object, variable and method). Most of the mappings were certain, however, some were questionable or weak. The students then built linkages between their 10 categories in the cognitive map via pair-wise comparisons to describe the effect of increasing the complexity of one category on the complexity of the other categories.

Sallie Henry has worked for years on metrics and concludes that traditional complexity metrics like lexical token count, control flow complexity of procedures or functions, and inter-connectivity of statements or system components are just not suitable for the object oriented paradigm. The mind set and goals are different.

Her research objectives are to define and validate object-oriented software complexity metrics, to understand maintenance activities in object-oriented systems and to propose metric instrumentation in an object-oriented software life cycle. She presented 6 questions and hypotheses, gave the results, then backed them up with data gathered from 2 large software systems, one for user interface management and on for quality evaluation.

Sallie's first validation model started with the set of metrics proposed by Chidamber and Kemerer last year at OOPSLA. She had automation problems with the coupling between objects (CBO) metric as defined by Chidamber and substituted 3 different metrics for coupling complexity and added 1 public interface metric. A second validation model used two size metrics from Dick Nance and Dennis Kafura.

Her objectives were focused around maintenance activities and the data was hard to get. After a valiant effort to find maintenance data, multiple linear regression and cross validation here are the conclusions:

- Size metrics are important but are not sole predictors and they were incorporated into the other metrics.
- Prediction of maintenance is possible with the full set of Chidamber based metrics.

Dirk Meyerhoff presented an overview of the Metrics Education ToolKIT (METKIT) Computer Aided Instruction (CAI) System. Its main goal is to support the use of measurement in software engineering by providing required conceptual knowledge about software measurement on the computer. Hypertext and graphical browsers are combined to present textual information on concepts and a graphic visualization of the relationships between them.

The system can be used by authors or experts to define and update knowledge about concepts and their relationships, and by readers or learners to browse the defined knowledge.

Dirk showed how concepts that are specific to measuring of Object Oriented Systems can be included in the system such that a learner can find, for instance, the definition of some measure, the entities and attributes it is supposed to measure, and tools that could be used for the actual measurement.

Ross Huitt shared his concern for metrics that focused on maintenance and understandability (readability). A series of bar charts were used to graphically illustrate that OO has shifted the complexity in systems. If we use traditional metrics we won't be measuring where the complexity is in OO systems, but rather where it used to be in structured systems. Function size and cyclomatic complexity aren't measuring the more significant aspect of complexity in OO systems.

He thought that the Chidamber and Kemerer metrics were a fairly good start, but also had some argument with how depth of inheritance tree (DIT), coupling between objects (CBO), and response for a class (RFC) were defined. He uses automated support from a Metrics for Object-Oriented Software Engineering (MOOSE) tool based on modified Chidamber metrics.

Ross has plans to fully instrument and measure a large scale project next year and the results will be published.

Pedro Inacio shared his project experience with the OBLOG (OBject LOGical) Workbench and made a plea for metrics at the design and analysis levels, rather than at the implementation level. He works with high level specification languages and plans to generate code. He is looking for patterns that can be measured with validated metrics so he can keep his managers interested in the object-oriented paradigm. There is a real need for some process oriented metrics.

Steven Bilow also came in search of metrics that can be applied prior to design and implementation. His goal is to quantify design complexity early in the product lifecycle thereby reducing implementation and maintenance complexity and effort.

He presented 2 unlabelled graphs of roughly the same complexity and indicated that they were remarkable similar. He then identified these graphs as being a control flow diagram from McCabe's paper on cyclomatic complexity, and a state model from Shlaer and Mellor's book "Object Lifecycles." He noted that there is a strong structural relationship between models of control flow and those of object state and inferred that graph theoretical measures (such as variations on cyclomatic complexity) may be able to quantify the complexity of the state models of object-oriented systems. Each of the 2 graphs shown were fully connected. Since graph theory provides a way to describe the complexity of any fully connected graph, similar techniques could be used as complexity metrics for both control flow and object state. Since McCabe's metrics for control flow is well accepted, similar state model metrics should be further explored. Steven concludes:

- Graph theory is useful in evaluating the complexity of object-oriented state models.
- Some object-oriented information models can be characterized, but work is incomplete.
- Chidamber & Kemerer had better metrics for single inheritance hierarchy and encapsulation. But even in these cases we can learn a lot about a system by looking at which states are visible from the outside versus those that are hidden internally. So even in the case of encapsulation, state model metrics may prove a useful method of analysis.

5. Group Discussion and Consensus

The group agreed that process metrics are important, but without good product metrics first, resource consumption measurements (the basis of process metrics) are not possible. We have to start with what is available right now. And even though it is scant, there is more data for validating product metrics than process metrics. Managers are not ready for object-oriented process metrics yet. The right mind set for using these metrics is very important and isn't very evident.

So we narrowed our discussion to product metrics for analysis, design and implementation. If source code (implementation) metrics can predict maintainability, perhaps the lower level metrics can be abstracted up into the design level and into the analysis level. But representation plays a big role. There is no common agreement about what constitutes the output of object-oriented analysis and design activities and this complicates the selection of appropriate metrics for these levels. If you want to work from the top down, rather than bottom up, you have to choose your methodology first and let it guide your selection of metrics. The point is, what you choose should map from one level to the next. Choose what you think is important and what you can measure. 6–12 metrics should be sufficient. Be careful if you take a large number of metrics then condense them down into a few categories ... you can lie with statistics. It is harder to say what is important and easier to say what can be measured.

Complexity is in communication links and dynamic dispatch. Simple access (read/write/initialize) of variables (1 or 2 lines of code—no side effect external to the object) can't really be considered to add complexity. This introduces the concept of intra and inter communication levels. Complexity is NOT productivity, NOR quality.

Concurrent and distributed systems have another kind of complexity—concurrency of objects and distribution of objects separately and concurrent distribution of objects collectively. We have no idea how to measure these highly dynamic systems. Our static metrics are barely off the ground.

Automation is a necessity, but what do you automate? Here are our suggestions:

CLASSES but—

We also need a larger grained measure (for categories, packages, building blocks, mechanisms, patterns, clusters—whatever you care to call them) and tools that operate on these meaningful sets. Inheritance really throws a curve into things and we don't agree on the best way to measure that yet. Problem domain will make a difference for calibrating the chosen set. We must be careful that semantics don't get lost in the packaging.

COUPLING but—at units larger than a single class. This was seen as one of the weaknesses of the Chidamber metrics—the level is too low.

COHESION but—at units larger than a single class. For the same reason.

STRUCTURE which includes inheritance, aggregation and association.

6. Future Directions

After we get product metrics nailed down and validated, then we can move on to process metrics. We need more experience and data from projects. We want to have a workshop next year and invite interested participants to focus on the product metrics we have recommended and help us validate them.

7. Participant Papers

"Object-Oriented System Complexity: An Integrated Model of Structure and Perceptions"

David P. Tegarden
Information and Decision Sciences
School of Business and Public Administration
California State University, San Bernardino
San Bernardino, CA 92407-2397
tegarden@gallium.csusb.edu

Steven D. Sheetz
Graduate School of Business
University of Colorado
Boulder, CO 80309-0419
sheetz_s@cubldr.colorado.edu

"Metrics for Object Oriented Systems"

Sallie Henry
Wei Li
Computer Science Department
Virginia Tech
Blacksburg,VA 24062
henry@vtopus.cs.vt.edu

"Metrics for Object-Oriented Software Development—Workshop Position Paper"

Ross Huitt
Bellcore
RRC-1H206
444 Hoes Lane
Piscataway, NJ 08854
bytor@ctt.bellcore.com

"Metrics for Object-Oriented Software Development: the OBLOG Workbench Project Experience"

Pedro Barros Inacio (*pbi@solo.inesc.pt*)
Peter Hartel (*pet@solo.inesc.pt*)
Espirito Santo Data Informatica SA.
Avenida Avlares Cabral 41-5
1200 Lisbon, Portugal

"Making Knowledge about Software Measurement Available on a Computer: The METKIT CAI System"

D. Meyerhoff
M. Müllerburg
GMD
Schloss Birlinghoven
D-2505 Sankt Augustin 1
Germany
mdirk@gmdzi.gmd.de

"Borrowing from McCabe: What Object-Oriented Methodologists Can Learn from Cyclomatic Complexity"

Steven C. Bilow
Network Displays Division
Tektronix, Inc.
P.O. Box 1000 60-646
Wilsonville, OR 97070-1000
steveb@orca.wv.tek.com

Contact information:

Sam Adams
Knowledge Systems Corporation
114 MacKenan Drive, Suite 100
Cary, NC 27511
Email: *ksc@cup.portal.com*

Teri Roberts
Object International, Inc.
8140 N. MoPac Expwy, Bldg 4 - Suite 200
Austin, TX 78759
Email:
71210.3642@compuserve.com
-or-
coad@applelink.apple.com

Rajiv Tewari
Temple University
CIS Department
Philadelphia, PA 19122
Email: *tewari@astro.ocis.temple.edu*