The Influence Reuse Will Have on Software Measurement

Duane W. Hybertson

MITRE Corp.
1120 NASA Road 1
Houston, TX 77058
Tel: (713) 333-0932
Email: dhyberts@mitre.org

David A. Eichmann

Software Engineering Program
University of Houston - Clear Lake
Box 113, 2700 Bay Area Blvd.
Houston, TX 77058
Tel: (713) 283-3875
Email: eichmann@rbse.jsc.nasa.gov

Abstract

For reuse to mature into an institutional context, we must move beyond anecdotal support to cost and estimation models supported by legitimate measures. This implies that the existing measures and models will need to be adapted to properly distinguish and credit reuse when it occurs and when it can occur. Furthermore, the *focus* of those measures and models may change.

Keywords: maturity, measurement, metrics

Workshop Goals: Assess the acceptance of measurement-based software reuse; understand how others measure reuse activities; work towards consensual standards of assessment.

Working Groups: Useful and collectible metrics, reuse management, organization and economics, reuse maturity models.

1 Background

As part of our relationship with the Information Systems Directorate at NASA's Johnson Space Center, we are currently involved in the support of the Defense Information Systems Agency / Joint Interoperability Engineering Organization / Center for Information Management in their work in institutionalizing reuse within the DoD community. In particular, we are members of both the reuse metrics team, responsible for defining and executing a measurement plan to assess reuse activity within the Defense Software Repository System (DSRS) and the pilot projects supported by the DSRS, and the Software Reuse Roadmap team, responsible for assessing the current state of reuse in government, industry and academia, and laying out a strategy for the DoD to best leverage its investment in research.

Eichmann is also currently the Director of Research and Development for the Repository Based Software Engineering (RBSE) program, a NASA supported reuse initiative, of which the AdaNET software repository is a part, and a member of the AIAA Software Reuse Standards Working Group, an effort to propose standards for repository interoperability protocols. RBSE is currently supporting Rockwell in a pilot project to reengineer the Space Shuttle flight analysis systems from legacy FORTRAN into an object-oriented implementation designed for reuse on other NASA/JSC shuttle systems.

2 Position

2.1 The Reuse Perspective

As reuse becomes a fundamental part of software engineering, it changes the way we view software. (Software is defined here to be all software content objects, including requirements, architectures, designs, code artifacts at both the system and component level, as well as the relationships among them.) Before reuse, software was only of temporary interest, to be created, delivered, and forgotten - even though it was frequently maintained for decades. The software that was produced got no respect from the industry as an object of inquiry; it was considered to be a special case, one-time effort. The generic methods and processes of producing it were considered to be of much greater interest and importance. The concept of reuse comes from the recognition that, within application domains, software problems and solutions begin to coalesce and stabilize around a certain conventional agreed-upon set of architectures. Design becomes variation on a standard theme. As the problem/solution set is refined and matures, the view of software changes from something that is transient and special case to something that is timeless, enduring, worthy of respect and inquiry. When this point is reached, the software industry is well on its way to becoming a traditional engineering discipline.

2.2 General Influence on Software Measurement

The change in perspective described above brings about a change in the focus and importance of software measurement. Before reuse, measurement was not viewed as particularly critical (except by a few disciplined organizations) and the focus was on development projects: the process of developing software (estimating effort to produce a system of a certain estimated size) and, to some extent, the quality of the developed product, especially its external behavior.

Reuse magnifies the importance of measurement, particularly the measurement of software content and the creation and utilization of that content. Now it is useful to know much more about the qualities and characteristics of the software itself because people need to communicate about it – it will be used by many people in many different contexts. There are new things to measure, such as distinctions between producing/discovering/articulating standard solutions in the form of artifacts and knowledge, managing the artifacts and knowledge, and using them to build new systems.

There is also the possibility that somewhat different attributes need to be measured in different domains, or at least the focus might differ. The domains themselves need to be measured, to assess maturation rates and artifact coverage.

2.3 Near-Term Effects

In the short term, the focus of reuse measurement will be on cost benefits and return on investment of reuse (compared to no reuse), and certification of parts. Reuse programs are frequently initiated based upon anecdotal claims, with little understanding of the context from which those anecdotes arise. A primary short-term goal is agreement within the reuse community on what comprises reuse of an artifact, allowing reasonable comparisons between projects and organizations. Differentiating between verbatim and adaptive reuse and accounting for these distinctions in cost models to adequately estimate effort for design with reuse need to be addressed quickly so that managers can clearly see the benefits derived from reuse.

We see near-term results primarily in what to count, how to count it, and how to assign cost and savings to what is counted. These results will be achieved on software artifacts that generally resemble the traditional products of the software life cycle, rather than on the results of domain engineering efforts currently underway. The deployment of domain architectures in practical application systems is only now becoming clear.

2.4 Long-Term Effects

In the long term, the focus will be on characterizing and measuring software content and its quality, not just the external behavioral view but structural views as well. There will be less emphasis on cost benefits of reuse versus no reuse because using standard solutions will have become the way of doing business and will not have to be justified. There will be less emphasis on source lines of code (SLOC), especially as the basis of productivity, because more useful and important metrics of the functionality and quality of software will be available. The deemphasis on SLOC will reflect the increased emphasis on products derived from earlier activities in the life cycle, particularly design quality assessment and the general state of domain development, effectively comprising a domain maturity measure. The quantification of the notion of variations on a theme (i.e., domain architectures) mentioned above will be a more important estimate of effort than of the total size of the system delivered.

3 Comparison

There are a number of recent additions to the literature relating to the position we elaborate here. Concerning maturity and its assessment, the recent special issue of *IEEE Software* (particularly

[1]), Pfleeger's paper on process maturity and metrics [2], and the reuse maturity model formulated in [3] relate the kind of activity we see being directed eventually not just toward processes and organizations, but toward domains as well.

Progress toward our near-term goals is indicated by reports such as those by Daskalantonakis [4] and Pfleeger [5]. Recent efforts by DoD and SEI on the derivation of a standard set of core metrics will lead to more uniform reporting and comparison of such efforts.

Finally, and perhaps more critically, cost models are beginning to be discussed in the literature, indicating a new phase in the maturation of reuse: [6], [7], [8]. Such models typically involve a single organization and the benefits derived from reuse activities. Their relevance to multiple contractor contexts, particularly government projects such as those done by DoD and NASA, have yet to be established.

It important to note that the work described in these references commonly starts from widely varying premises. There is demonstrable benefit derived from adaptive reuse, and we provide different weightings for verbatim reuse and adaptive reuse in the cost models that we are exploring. Poulin, however, counts only verbatim reuse in the calculation of return on investment [7]. The variations in results make comparison difficult. The reuse community needs to arrive at a consensual cost model approach that allows proper comparison, so that the benefits of a mature reuse program are clear and unarguable. Only then will institutionalization of reuse become a reality.

References

[1] R. Dion, "Process Improvement and the Corporate Balance Sheet," *IEEE Software*, vol. 10, pp. 28–35, July 1993.

[2] S. L. Pfleeger, "Software Metrics in a Process Maturity Framework," *Journal of Systems and Software*, pp. 255–261, September 1990.

[3] SPC, "Reuse Adoption Guidebook," Tech. Rep. SPC-92051-CMC, Version 01.00.03, Software Productivity Consortium, 1992.

[4] M. K. Daskalantonakis, "A Practical View of Software Measurement and Implementation Experiences within Motorola," *IEEE Transactions on Software Engineering*, vol. 18, pp. 998–1010, November 1992.

[5] S. L. Pfleeger, "Lessons Learned in Building a Corporate Metrics Program," *IEEE Software*, vol. 10, pp. 67–74, May 1993.

[6] J. Margono and T. E. Rhoads, "Software Reuse Economics: Cost-Benefit Analysis on a Large Scale Ada Project," in *Proceedings Fourteenth International Conference on Software Engineering*, pp. 338–348, May 11-15 1992.

[7] J. Poulin, "A Reuse Metrics and Return on Investment Model," in *Proceedings Second Workshop on Software Reusability*, March 24-26 1993.

[8] J. Poulin, "Issues in the Development and Application of Reuse Metrics," in *Proceedings Fifth International Conference on Software Engineering and Knowledge Engineering*, pp. 152–159, June 1993 1993.

4 Biography

Duane Hybertson is a Member of the Technical Staff at the MITRE Corporation in Houston, Texas. He investigates software engineering and technology issues for the Software Technology Branch in NASA Johnson Space Center's Information Systems Directorate. He is currently working on two software reuse tasks related to DoD's Software Reuse Initiative and originating from the Defense Information Systems Agency. He was previously a senior engineer with Lockheed on the Space Station Freedom Software Support Environment. Prior to that, he had an appointment at the Software Productivity Consortium in Virginia, where he developed part of the original SPC reuse synthesis system prototype. He received the B. A. in Math from Northwest Nazarene College in 1970, the Ph.D. in Educational Research Methods and Statistics from New Mexico State University in 1974, and the M.S. in Computer Science from the Johns Hopkins University in 1985.

David Eichmann is an Assistant Professor of Software at the University of Houston - Clear Lake. He is Director of Research and Development for the Repository Based Software Engineering Program, a NASA software reuse initiative, and lead for the metrics team supporting the Defense Information Systems Agency / Center for Information Management. He was previously on the faculty at West Virginia University, where he headed the SoRReL group. He received the B.S. in 1978, the M.S. in 1983 and the Ph.D. in 1989, all from the University of Iowa in computer science.