Component-Based Software Engineering: a Quantitative Approach

Miguel Goulão
Departamento de Informática
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
2829-516, Caparica, Portugal
+351212948536
miguel.goulao@di.fct.unl.pt

1. DESCRIPTION OF PURPOSE

1.1 Problem Description

Component-based development (CBD) claims include reductions on the development costs and time to market, along with an improvement of the developed system's overall quality [1]. These benefits come with a price: existing components have to be evaluated, selected and, often, adapted to suit the particular needs of the component-based software being assembled. In CBD, those tasks must be performed without access to the component's source code, since it is usually not available. This constrains our ability to assess such components and differentiates assessment activities from those performed with white-box software reuse.

The quality evaluation of software components is carried out, at least to some extent, in an ad-hoc fashion. Evaluations are not independently and consistently replicable, and their success depends highly on the assessor's expertise.

1.2 Limitations of the current state of the art

The bulk of research in component-based software has been devoted to the functionality and composability of software components, but the area of quality assessment for CBD remains fairly unexplored. For instance, there is no widely accepted quality model suited for CBD assessment [2], although there are some attempts to adapt the ISO9126 quality model to CBD [3].

The ability to predict the final system's properties from the properties of reused components is a growing concern for the research community [4]. Some proposals aim at the development of prediction-enabled component specifications to facilitate automated prediction of properties [5, 6]. Their focus is on the analysis of run-time quality attributes.

A complementary research niche, where we are conducting our research, is the one of static analysis of quality attributes such as reusability or maintainability, using software metrics. Several authors contributed with proposals for the evaluation of component interfaces and dependencies [7-9], with a particular concern on their reusability. Others proposed metrics to assess component integration density as an indirect measure of the component's internal complexity [10]. All of these proposals take a component-centric approach on their evaluation since they assess components in isolation.

As argued by Wallnau and Stafford, it is probably more beneficial to perform the evaluation on assemblies, rather than on individual components [11]. We are fundamentally interested in selecting the components that maximize the overall system quality. In this assembly-focused view, individual component assessment may be performed as part of the component assembly evaluation, but the focus is on the overall best solution. Examples of metrics following this view can be found at [10, 12].

The existing quality models and metrics proposals include, to some extent, informal specifications. Even when the metrics are specified through mathematical formulae, the elements in those formulae are usually specified in natural language. This often creates ambiguity, as there are several plausible interpretations for such definitions. Ambiguity is an obstacle when conducting independent validation efforts. Such efforts are essential to a sound validation of quality models and metrics proposals. Another common obstacle is that tool support for metrics collection is usually unavailable, as their proponents either did not build collection tools or do not provide access to them.

With the exception of [9], most proposals went through scarce validation. Frequently, authors present metrics using toy examples that help clarifying their rationale and collection process, but are insufficient to their validation, as discussed in [13].

1.3 Significance of the problem

Currently, component assessment frequently requires experts' opinion. Based on their experience and on an implicit quality model which is dependent on the personal view, assessors make judgments that are at least partially informal, subjective and hard to replicate. Moreover, experts may not be available to perform the assessment. In such event, practitioners are left with an essentially blind choice, with respect to the quality of components. With the growing usage of reusable components in software construction, a blind choice is a major threat to the success of CBD projects.

2. GOAL STATEMENT

2.1 Main Contributions

A possible way of circumventing the problems identified in the previous sections is to evolve the integrated development environments (IDEs) so that they include functionalities to facilitate assessment in CBD. An analogy can be made to the automated refactoring functionalities of some IDEs.

However, the validation of metrics to assess components and assemblies in the context of a quality model is a pre-requisite for such evolution, in what concerns the quality attributes covered by

metrics-based static analysis. The integration of such help in IDEs is essential, if a widespread adoption of a quantitative approach to CBD is sought.

With our research, we plan to demonstrate the feasibility of a formalized approach to CBD assessment that combines rigor with simplicity, is replicable, and can be integrated with current development environments, to provide automated advice to practitioners involved in CBD assessment. Our main contributions to facilitate the achievement of this goal include:

i. the proposal of a quality model for CBD;

ii. a proof of concept concerning the applicability of a metamodel-based approach to metrics collection in the scope of CBD;

iii. the formalization of metrics for CBD available in the literature and their independent empirical assessment;

iv. the development and formal definition of a metrics set for the assessment of individual components and component assemblies that fulfill the measurement needs of the model referred in i;

v. the validation of those metrics and of the quality model, through the conduction of controlled experiments;

vi. the production of heuristics based on those metrics and the quality model;

vii. the development of adequate tool support that integrates well with current development environments; the created tool support will be made available to the community, to foster independent validation studies, both of our and other proposals.

2.2 Contributions impact

In this section, we will trace the expected impacts to the claimed contributions, by using their reference ids (i, ii, …, vii).

Two of the major problems of current quality models is that either they are not suited for CBD, or have not been thoroughly validated. By proposing a quality model tailored for CBD (i), along with suitable metrics to assess components and assemblies with respect to that quality model (iv), and by validating both the metrics and the quality model (v) we can mitigate both these problems.

The quantitative assessment of component and component assemblies requires a rigorous approach. Different assessors evaluating the same component or component assembly in different locations must be able to replicate the assessment conditions and get the same results. This requirement of replicability is generic to any scientific experiment and crucial if we want to make independent assessments of components.

The metamodel-based approach to metrics collection (ii) deals with the limitation of previous approaches, where the elements being measured and their relationships are informally specified. This shortcoming is perhaps subtle, but its effects are overwhelming, as it leads to ambiguous and therefore conflicting metrics definitions: two independent assessors using (apparently) the same metric may obtain different results, if their perception of the artifacts being measured is not exactly the same. Expressing metrics definitions formally (iii, iv) upon a metamodel completes our rigorous approach for building up a measurement framework suitable to independent assessments.

Contributions (vi) and (vii) have an impact on the packaging of our quantitative approach to practitioners. While the heuristics based approach is expected to be the most visible outcome of the automated assessment, the inclusion of an assessment tool in common IDEs is essential to foster its usage. This aspect will be further discussed in the next section.

3. APPROACH

3.1 Approach Outline

We expect to develop a quality model suitable to the specific needs of CBD, in the near future. A Goal-Question-Metric approach [14] will follow, to determine which metrics should be collected to assess components and assemblies within the scope of the quality model to be proposed.

Most of our work so far has been devoted to the research and development of an infrastructure to support formal metrics specification and metrics collection from component assemblies' repositories. The framework uses UML meta-class diagrams where operation semantics is formalized with OCL clauses. The OCL language provides the required formality without sacrificing understandability, since it was conceived with usability in mind for UML practitioners.

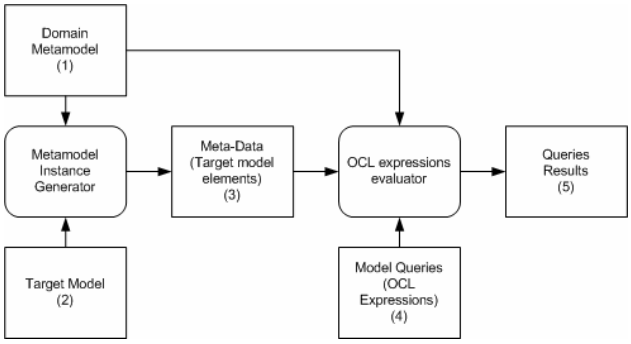

Fig. 1 Metrics collection process overview

Figure 1 outlines the metrics collection process. We use a domain metamodel (1) to express the basic concepts from which we want to extract relevant properties. Then, we populate the metamodel with an instantiation (a model) that represents the target model to be assessed (2). This instantiation (3) is a graph where the nodes are meta-objects and the edges are meta-links. We use OCL expressions which define the metrics to be collected (4) to traverse the graph and compute the metrics. Heuristics definitions may also be defined in OCL, at this stage. Finally, we analyze the results of the OCL expressions - both metrics and heuristics (5).

The approach itself is generic, both in what concerns the underlying component model and the evaluation target (individual components, or component assemblies). We have used it successfully to assess components and component assemblies expressed upon the UML 2.0 metamodel (the original components were reverse engineered from JavaBeans to UML 2.0) [15, 16] and the CORBA Component Model metamodel [17, 18].

The approach is also flexible: adding a new metric, requires defining a new OCL expression that specifies how the metric should be computed. Heuristics may also be defined using the same technique, typically through the specification of OCL predicates that check for metrics values beyond their expected range [15, 16].

Finally, the approach is open, in the sense that the metrics are defined using standard OCL clauses, and it essentially requires a common UML tool with OCL support, upon which we can load a model (the domain metamodel) and populate it with the appropriate instances.

We are currently using the USE tool [19] for this purpose, but this kind of computational support is likely to be available in many UML tools in the near future, as they become "OCL-enabled". Those tools will support the UML 2.0 component metamodel, either internally (on their data dictionary) or at least on their external interface (using some UML 2.0-compliant XMI version) For other component metamodels, we will still have to go on developing instances generators, or, in alternative, to use UML profiles, such as the one for EJB. We do not regard this as a major drawback, due to the current momentum of MDA.

3.2 Validation

The evaluation of our proposals is being carried out in three different ways:

First, most of the work done so far has been devoted to setting up the assessment infrastructure and exercise it with different underlying metamodels and metrics for different purposes, exercising different aspects of components interfaces and interaction mechanisms. By doing so, we were able to assess the flexibility of our approach.

Second, the peer review of our work in the context of international scientific forums is providing us with an external assessment of the soundness of our proposals.

Third, the primary source of validation for our quality model and metrics proposals will be the controlled experiments. So far, we have done so only in a small scale, with metrics proposed by other authors [15, 16]. We plan to use open source component based repositories as subjects for the evaluation of our proposals. The option for this kind of repositories is of a practical nature: we have no access to industrial CBD projects upon which we can collect the metrics and assess the quality model to be defined. To simulate the black-box access to components, we will restrain from using in our metrics definitions information that would be normally hidden in commercial components, as this would be a threat to the validity of the obtained results.

4. REFERENCES

[1] Szyperski, C., Gruntz, D., and Murer, S., *Component Software: Beyond Object-Oriented Programming*, 2nd ed. New York: ACM Press - Addison Wesley, 2002.

[2] Simão, R. P. S. and Belchior, A. D., "Quality Characteristics for Software Components: Hierarchy and Quality Guides", in *Component-Based Software Quality: Methods and Techniques*, *LNCS 2693*, A. Cechich, M. Piattini, and A. Vallecillo, Eds.: Springer, pp. 184-206, 2003.

[3] Bertoa, M. and Vallecillo, A., "Quality Attributes for COTS Components", *6th International Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'2002)*, Málaga, Spain, 2002.

[4] Crnkovic, I., Schmidt, H., Stafford, J. A., and Wallnau, K., "6th ICSE Workshop on Component-Based Software Engineering: Automated Reasoning and Prediction", *ACM SIGSOFT Software Engineering Notes*, vol. 29, pp. 1-7, 2004.

[5] Wallnau, K. and Ivers, J., "Snapshot of CCL: A Language for Predictable Assembly", Carnegie Mellon, Software Engineering Institute, Technical Note, CMU/SEI-2003-TN-025, June 2003.

[6] Larsson, M., "Predicting Quality Attributes in Component-based Software Systems", PhD Thesis, Department of Computer Science and Engineering, Mälardalen University, Västeras, Sweden, 2004.

[7] Boxall, M. A. S. and Araban, S., "Interface Metrics for Reusability Analysis of Components", *Australian Software Engineering Conference (ASWEC'2004)*, Melbourne, Australia, 2004.

[8] Gill, N. S. and Grover, P. S., "Few Important Considerations for Deriving Interface Complexity Metric for Component-Based Software", *Software Engineering Notes*, vol. 29, pp. 4-4, 2004.

[9] Washizaki, H., Yamamoto, H., and Fukazawa, Y., "A Metrics Suite for Measuring Reusability of Software Components", *9th IEEE International Software Metrics Symposium (METRICS 2003)*, Sydney, Australia, 2003.

[10] Narasimhan, V. L. and Hendradjaya, B., "A New Suite of Metrics for the Integration of Software Components", *The First International Workshop on Object Systems and Software Architectures (WOSSA'2004)*, South Australia, Australia, 2004.

[11] Wallnau, K. and Stafford, J. A., "Dispelling the Myth of Component Evaluation", in *Building Reliable Component-Based Software Systems*, I. Crnkovic and Larsson, Eds. Boston, London: Artech House, pp. 157-177, 2002.

[12] Hoek, A. v. d., Dincel, E., and Medvidovic, N., "Using Service Utilization Metrics to Assess and Improve Product Line Architectures", *9th IEEE International Software Metrics Symposium (Metrics'2003)*, Sydney, Australia, 2003.

[13] Goulão, M. and Abreu, F. B., "Software Components Evaluation: an Overview", *5ª Conferência da APSI*, Lisbon, October, 2004.

[14] Basili, V. R., Caldiera, G., and Rombach, D. H., "Goal Question Metric Paradigm", in *Encyclopedia of Software Engineering*, vol. 1, J. J. Marciniak, Ed.: John Wiley & Sons, pp. 469-476, 1994.

[15] Goulão, M. and Abreu, F. B., "Cross-Validation of a Component Metrics Suite", *IX Jornadas de Ingeniería del Software y Bases de Datos*, Málaga, Spain, 2004.

[16] Goulão, M. and Abreu, F. B., "Validação Cruzada de Métricas para Componentes", *IEEE Transactions Latin America*, vol. 3, 2005.

[17] Goulão, M. and Abreu, F. B., "Formal Definition of Metrics upon the CORBA Component Model", First International Conference on the Quality of Software Architectures (QoSA'2005), Erfurt, Germany, September, 2005.

[18] Goulão, M. and Abreu, F. B., "Composition Assessment Metrics for CBSE", *31st Euromicro Conference - Component-Based Software Engineering Track*, Porto, Portugal, September, 2005.

[19] Richters, M., "A UML-based Specification Environment". http://www.db.informatik.uni-bremen.de/projects/USE: University of Bremen, 2004.