Assessing the Quality of Object-Oriented Designs

Ralf Reißing
Institute of Computer Science, University of Stuttgart
Breitwiesenstr. 20-22
70565 Stuttgart, Germany
+49 711 7816 338
reissing@informatik.uni-stuttgart.de

ABSTRACT

Object-oriented design plays a pivotal role in software development because it determines the structure of the software solution. Once the design has been implemented, it is difficult and expensive to change. Therefore high design quality is vital for reducing software cost, and quality assurance in the design stage has a high return on investment. Unfortunately, it is mostly unclear what design quality really is.

This thesis wants to clarify the general notion of design quality and to make design quality measurable. The approach is as follows: The criteria for design quality and their relationships are identified. Then objective and subjective metrics for each criteria are introduced. Together the criteria and the metrics form a quality model for object-oriented design. This model can be used for design assessment both in comparing design alternatives and in design improvement.

Keywords

object-oriented design, design quality, design quality model, design assessment, design quality assurance

1. PROBLEM

Design is an important cost driver in software development, for it does not only cause the cost of its own creation, but it also heavily influences the cost of the following phases, i.e. implementation and maintenance. The design phase only takes 5-10% of the total effort (over the whole software life-cycle), but a large part (up to 80%) of the total effort goes into correcting bad design decisions [1]. If bad design is not fixed in the design phase, the cost for fixing it after delivery of the software is between 5 and 100 times higher [3].

But even if the initial design is good enough for the moment, there may be difficulties when trying to make changes and extensions in the maintenance phase. The characteristics of the design, e.g. changeability, heavily influence the ease of maintenance. As at least 50% of the total life-cycle cost goes into maintenance [2], it is very cost effective to have high design quality early – and to maintain it throughout the life-cycle.

2. GOALS

In order to create and maintain a high quality design, quality assurance in the form of design assessment and review is needed. It is important and useful to measure design quality early in software development. Strong correlations between design metrics (e.g. modularity metrics) and the maintainability of systems have been identified [7], so design *measurement* is useful for quality assurance. Early design measurement means, of course, that it should not depend on detailed design information or even on code.

Design measurement in itself, i.e. without a purpose, is useless because the actual measurements would have no meaning. Therefore the metrics have to be associated with a *quality model*. The quality model determines the interpretation of the measurements and thus defines the notion of design quality.

Design assessment can be done by experts using checklists which are based on the quality model and its metrics. However, typical designs are so big that quality assurance becomes a time-consuming activity. Therefore it is more efficient to use a *tool* for design assessment. Even though such a tool can never replace a human expert, it can help him (or her) to identify components of a design that are (potentially) troublesome. The tool can also help to compare design alternatives by evaluating each alternative and comparing the results.

3. APPROACH

The approach to design assessment is depicted in figure 1. The design is supposed to include a UML model, an instance of the UML metamodel [4]. UML is used here because it is the standard notation for object-oriented design. UML artifacts are available early in the design stage and suitable for design assessment.

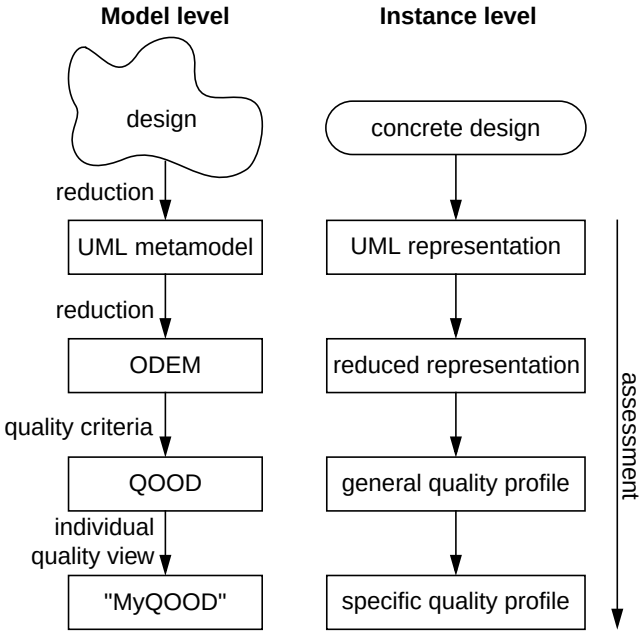

Figure 1: Deriving a specific quality profile from a design

From the UML model the design artifacts that are relevant for design assessment are selected. The result is a reduced design model that is an instance of the metamodel ODEM (Object-Oriented Design Model; for details see [6]). A general quality profile is derived from the reduced design model by using the quality model QOOD (Quality of Object-Oriented Design). Because qual-

ity requirements differ from project to project, the quality model has to be adaptable. Therefore QOOD is a general, generic quality model that serves as a basis for deriving specific quality models ("MyQOOD"). A specific quality model yields a specific quality profile when applied to the general quality profile. The specific quality profile is the input for the final design assessment by a reviewer. Specific quality models incorporate the specific quality requirements and an individual quality view, e.g. the view of the customer or a maintenance programmer. These parameters decide which aspects of the general model are relevant and how important the aspects are in relation to each other (resulting in weights).

In order to create the general design quality model QOOD, first the relevant characteristics for the quality of object-oriented design are identified (quality criteria). Then the criteria are connected with (measurable) attributes of the design, and metrics for these attributes are defined. To avoid the common problem of vague definitions of metrics, the metric definition are based on ODEM as a formal reference model of OOD.

Some criteria, e.g. understandability, cannot really be measured objectively. Therefore, additional subjective metrics are needed. In order to make it easier for a reviewer to determine these metrics, for each criterion questionnaires are given. The questionnaires ask for certain characteristics a design should (and should not) have. This approach also enhances reproducibility and repeatability of subjective measurement. Even if objective metrics are available, they often do not capture all possible aspects. Therefore the ultimate quality metric for each criterion is a subjective metric that is derived by combining the results from the objective metrics and from the questionnaires.

In order to make the application of the model easier, it is supported by a measurement tool. The tool calculates all the metrics needed (as far as they can be automatically determined) on the UML model. The UML model standard file format is XMI (XML Metadata Interchange; see [5]), so the input to the tool is an XMI file. Currently a prototype of this tool is developed in a master's thesis project.

The general quality model is validated by applying it to various design alternatives for a medium sized project and comparing the results of the model with expert opinions on the design alternatives. The validated model is then used in case studies to assess the effects of the application of various refactorings and design patterns on design quality. This way it can be determined whether the promised benefits of patterns and refactoring are real.

4. STATUS

Criteria, metrics and checklists for the model have been selected and combined to form the general quality model. Relationships between the criteria have been identified and added to the model. The model is supposed to be ready for application and validation by the end of July. The validation and assessment activities for the model will last until October. From this date, the main activity will shift to writing up the thesis.

5. REFERENCES

[1] Bell, G.; Morrey, I.; Pugh, J.: Software Engineering: A Programming Approach. Prentice Hall, New York, 1987.

[2] Boehm, B. Software Engineering. IEEE Transactions on Computers 25, 12, 1976, 1226-1241.

[3] Boehm, B., and Basili, V. Software Defect Reduction Top 10 List. IEEE Computer 34, 1, 2001, 135-137.

[4] Object Management Group. Unified Modeling Language (UML) Specification. http://www.omg.org/technology/documents/formal/unified_modeling_language.htm

[5] Object Management Group. XML Metadata Interchange (XMI) Specification. http://www.omg.org/technology/documents/formal/xml_metadata_interchange.htm

[6] Reißing, R. Towards a Model for Object-Oriented Design Measurement. In Brito e Abreu, F. et al. (eds.): Proceedings of the 5th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (Budapest, June 2001), 71-84.

[7] Rombach, H. Design Measurement: Some Lessons Learned. IEEE Software 7, 2, 1990, 17-25.