

Quality Assessment of Enterprise Software Systems

Cristina Marinescu
LOOSE Research Group
“Politehnica” University of Timișoara, Romania
cristina.marinescu@cs.upt.ro

Abstract

In the last years, as object-oriented software systems became more and more complex, the need of having tools that help us to understand and to assess the quality of their design has increased significantly. This applies also to enterprise applications, a novel category of software systems. Unfortunately, the existing techniques for design's understanding and quality assessment of object-oriented systems are not sufficient and sometimes not suitable when applied on enterprise applications. In the current Ph.D. we propose a new approach which increases the level of understanding and the accuracy assessment of the design of enterprise software systems.

Keywords: enterprise software systems, meta-models, quality assessment

1. Introduction

In the recent years a novel category of software systems namely *enterprise* emerged. These systems are about the display, manipulation, and storage of large amounts of often complex data and the support or automation of business processes with that data [2]. Usually, enterprise systems involve two programming paradigms: object-oriented and relational, the last for ensuring the persistency of the involved data within the application. Mixing the commands that ensure the persistency and application logic hampers the understanding and the testing of such applications. This led to a multi-layered architecture, consisting of three primary layers namely *data source* (ensures the communication with the database), *domain* (encapsulates the logic of the system) and *presentation* (ensures the interaction between the user and the application) [1].

In order to facilitate the maintenance of this new type of software systems, we need to understand and assess the quality of their design automatically.

2. A Meta-Model for Enterprise Systems

The source code of a software system can be hardly manipulated for understanding and assessing the quality of its design automatically and thus we need to represent it at a higher level of abstraction based on a meta-model which specify the main design entities found in the system. It is clear that a meta-model for representing object-oriented systems does not capture all the design entities found in an enterprise system (*e.g.*, database tables) and neither the relations between existing design entities from object-oriented and relational paradigms (*e.g.*, accesses to database tables from the body of methods). In this context, one of the main goals of the current Ph.D. is to define a meta-model for representing enterprise systems which contains, on one hand, entities from a regular object-oriented system and, on the other hand, entities regarding the relational part of the system and the interactions between the two paradigms.

3. Assessing the Quality of Design in Enterprise Software Systems

A fundamental problem regarding the process of quality assessment of enterprise software systems is their heterogeneity – they encapsulate different paradigms and must fulfill specific design rules and patterns [2, 3, 9, 8]. All these aspects must be considered when the quality of the design and implementation are evaluated. Unfortunately, almost none of the analyses techniques for assessing the quality of an object-oriented application take into consideration this type of heterogeneity and due to this reason the current quality assessment techniques strongly needs improvement.

In Figure 1 we depict our vision of a generic design quality assessment approach for an enterprise software systems. The approach takes into consideration the heterogeneity of such systems and the consequent need to reflect this characteristic in the techniques used for quality assessment. The first important distinction is that each of three layers of an enterprise system must be addressed by a distinct set of spe-

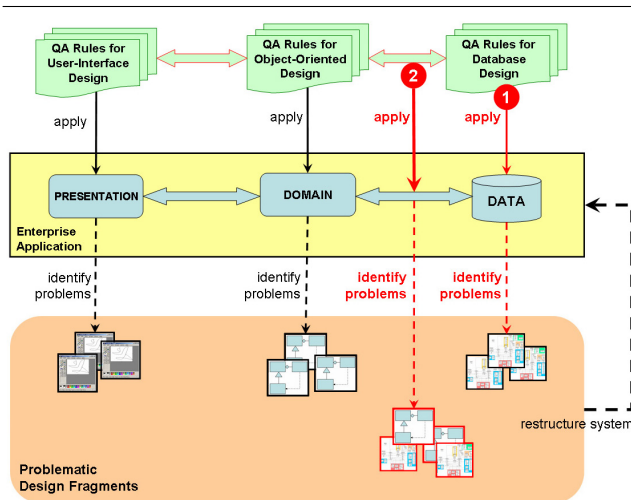


Figure 1. Quality Assessment Process for an Enterprise Software System.

cialized quality assurance rules. These design rules exist for both the design of presentation [8] and also for the design of data source [2, 9, 3] layers. Based on these rules, problematic design fragments can be identified within the aforementioned layers. Yet, none of the three layers exists in isolation; consequently there is a significant amount of the system's complexity involved in the relation between these layers. This brings us to two issues that the current Ph.D. aims to address (marked in Figure 1 with the bullets numbered 1 and 2):

1. How can the rules and patterns within the data source layer of an enterprise application be made quantifiable? What is the proper tool support needed to detect design problems within the data source layer?
2. What are the proper quality assessment design rules that specify the relation between the domain and the data-source layer, especially if the data-source layer is based on a relational database model? How can we overcome the paradigm shift? What is the proper tool support needed to detect design problems automatically?

4. Current Status of the Work

In [5] we introduced the meta-model called DATES we have defined for representing enterprise applications. According to the defined meta-model, we developed a model loader which extract design information both from the source code as well as the database schema of an enterprise software system. We evaluated the accuracy of the ex-

tracted model by conducting different experiments on a suite of enterprise software systems. The experiments reveals that the extracted model contains at the level of stored accesses from the bodies of methods to database tables several false negatives – usually they appear because the source code contains embedded statements as strings which access tables from the relational database which does not exist.

We integrated DATES within the iPLASMA [6] environment in order to be able to browse through the design entities of enterprise systems and also for understanding and assessing the quality of their design. Understanding and assessing the quality of the design of enterprise applications is performed using techniques for object-oriented systems (e.g., software metrics, detection strategies [7]) and also specific techniques like the ones we defined in [4]. In [4], based on specific characteristics of design entities belonging to the data source layer, we modified existing quality assurance techniques specific to object-oriented systems in order to make them suitable for enterprise applications.

In the future we intend to define and automatize a comprehensive suite of specific quality assessment analyses for enterprise systems (e.g., analyses related to the duplication of SQL statements within the object-oriented source code).

References

- [1] K. Brown et al. *Enterprise Java Programming with IBM WebSphere*. Addison-Wesley, 2001.
- [2] M. Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2003.
- [3] W. Keller. Mapping objects to tables: A pattern language. In *Proc. European Conference on Pattern Languages of Programs*, 1997.
- [4] C. Marinescu. Identification of design roles for the assessment of design quality in enterprise applications. In *Proc. IEEE International Conference on Program Comprehension*, 2006.
- [5] C. Marinescu and I. Jurca. A meta-model for enterprise applications. In *Proc. International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASCO6) - submitted*. IEEE Computer Society Press.
- [6] C. Marinescu, R. Marinescu, P.F. Mihancea, D. Rațiu, and R. Wetzel. iplasma: An integrated platform for quality assessment of object-oriented design. In *Proc. IEEE International Conference on Software Maintenance (Industrial and Tool Volume)*, 2005.
- [7] R. Marinescu. Detection strategies: metrics-based rules for detecting design flaws. In *Proc. IEEE International Conference on Software Maintenance*, 2004.
- [8] R. Martin, D. Riehle, and F. Buschmann. *Pattern Languages of Program Design 3*. Addison-Wesley, 1998.
- [9] C. Nock. *Data Access Patterns: Database Interactions in Object-Oriented Applications*. Addison-Wesley, 2003.