Chapter 7

Quality-Driven, Semantic Information System Integration: The QuaD²-Framework

Steffen Mencke  
Otto-von-Guericke University, Germany

Martin Kunz  
Otto-von-Guericke University, Germany

Dmytro Rud  
Otto-von-Guericke University, Germany

Reiner Dumke  
Otto-von-Guericke University, Germany

EXECUTIVE SUMMARY

The importance of automatic integration in every field of application is beyond controversy these days. Unfortunately, existing solutions are mainly focusing on the automation aspect. But for the success in the long run, the quality must be of substantial interest – it is an inherent characteristic of any product (Garvin, 1984). Existing quality-related information can be reused to optimize this aggregation of entities to thereby always provide the best possible combination (Kunz et al., 2008b). Such aggregation of entities can be done taking into consideration different characteristics like quality attributes, functional requirements, or the ability for automated procedures.

INTRODUCTION

Existing approaches for automatic entity assembly suffer from certain problems. There exists no common model that is applicable in every situation where small parts are assembled to form a complete architecture. Existing approaches focus on special domains like e.g. SOA or e-Learning. Another point of critique is the only sporadically emerging, throughout focus on quality. Existing knowledge
is often not reused in contrast to information and data. Rarely expert knowledge is used to describe the assembly of entities (Meder, 2006; Pawlowski, 2005; Helic, 2005; Mencke & Dumke, 2007).

Sometimes individual quality requirements are taken into consideration (e.g. for services: Zeng et al., 2003; Lin et al., 2008). Their focus is on Quality of Service only. No product-related quality attributes are continuously used. A framework for the quality-driven assembly of entities, taking into account the derived need for better solutions, is proposed here. Besides this quality-oriented characteristic, the usage of semantic knowledge and structured process descriptions enable an automatic procedure. Especially the combination of both is a promising approach.

It will be shown, that the introduced framework is even more valuable due to the fact that it is not restricted to content aggregation, but also usefully applicable for all domains where a whole should be qualitatively assembled out of small parts. This work follows this point of view to not to restrict the scope of the framework. The concrete occurrence of the intended system uses elements that need to work together to provide the intended system functionality. These elements are abstractly defined as entities at this point.

High-flexible infrastructures have manifold advantages compared to monolithic products. Because of this, a lot of initiatives propose approaches for the integration of single components (e.g. services, content). Semantic metadata provide the basis for the automation of this process (Kunz et al., 2008a). But so far, either only functional requirements or single quality attributes are taken into consideration. A throughout consideration of existing and updated empirical data and the use of semantic descriptions of included functionality or reached quality of an entity promises better solutions (Kunz et al., 2008b).

THE FOCUS ON QUALITY

The better is the enemy of the good. Why should somebody be satisfied with something, if he has the need and resources to achieve a better result? The answer is: he should not. And this is entirely about quality. A product’s perceivable quality is a key factor for the long term success of a company (Buzzell & Gale, 1987). Therefore, quality is defined according to the definition of the ISO 9000 standard (ISO/IEC, 2004b):

**Definition:** Quality is the “degree to which a set of inherent characteristics fulfils requirements” (ISO/IEC, 2004b).

A quality attribute is such a characteristic. To achieve quality in the field of software engineering, measurement is the fundamental basis: “you cannot improve what you cannot measure.” With software measurement it becomes possible to understand and communicate, to specify and achieve objectives, to identify and resolve problems as well as to decide and improve (Ebert & Dumke, 2007).

**Definition:** Software measurement is the approach to control and manage the software and to track and improve its performance (Ebert & Dumke, 2007).

Figure 1 comprises general software measurement phases and methods.

Measuring certain attributes is only the first step. The interpretation of the results is important, too. It is necessary because the human mankind is rarely capable to directly comprehend the meaningful information of the real world (see Figure 2).

Certain international activities were and are performed in order to standardize the expertise in this field. That ensures the usage and improvement of the current state of art on a global scale. The most important standards for software measurement are:

How to do things (described in life cycle processes):