Chapter VII

Introducing Non-functional Requirements in UML

Guadalupe Salazar-Zárate
Technical University of Catalonia, Spain

Pere Botella
Technical University of Catalonia, Spain

Ajantha Dahanayake
Delft University of Technology, The Netherlands

ABSTRACT

This chapter introduces an approach concerned with the non-functional features of software systems. The specific objectives of the research focus on the possibility of developing mechanisms to capture non-functional information in the development of software systems in a similar manner to its counterpart, the functional information. Particularly, the research described in this chapter focuses on the possible extension of the Unified Modeling Language (UML) (Booch et al., 1998). In order to get an initial specification of some non-functional requirements, the Software Quality Standard ISO/IEC 9126 (International Standard, 1991) is used. The language NoFun (Botella et al., 2001) is the basis used to achieve some organization about the non-functional concepts used in this approach.

INTRODUCTION

Requirements Engineering (RE) has been progressively recognized during the last decade as an important discipline within its own research community (Filkenstein, 1994; Lamsweerde, 2000; Nuseibeh & Easterbrook, 2000). In Zave (1997, p.315), RE has been defined as “a branch of software engineering concerned with the real-world goals for functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families.”
Industry is also clamoring for importance to be placed on this discipline. A European industry survey points out “Requirements specification” and “Managing customer requirements” as the two main problems in software and system development (50% have marked both as a “major problem,” 35% as a “minor problem,” and less than 12% as “no problem”). This survey was performed by the European Software Institute (European Software Institute – ESI, 1996) in 17 different countries. The survey analyzed the data obtained from about 4,000 questionnaires distributed among product and services companies in the IT sector.

The interest, relevance, and vitality of the RE emerging discipline is clearly reflected in specific conferences (such as the IEEE Joint Conference on Requirements Engineering), specific Journals (e.g., the *Springer Requirements Engineering Journal*), as well as in more general conferences, workshops, and journals that consider this topic.

In the RE processes (e.g., elicitation, modeling, validation, etc.), it is common to distinguish between functional and non-functional requirements. The functional requirements describe the functions and services of the system in terms of the users’ goals. Non-functional requirements are a broad term covering both qualities and constraints. Qualities (e.g., usability, scalability) can be negotiated, but constraints (as time constraints or the operating system to be used) are not subject to negotiation and must be satisfied. Therefore, both functional and non-functional requirements are relevant to software systems development.

The relevance of functional requirements has been traditionally well-covered by the modeling techniques, where a lot of research has been done. The model captures the functions and services to be provided by the system that can then be converted into design, and finally, into code. This research has resulted in techniques and tools, especially for modeling or specification languages, either formal or semi-formal. However, non-functional requirements, quality, and constraint properties are not usually covered by these modeling techniques. Quality requirements, when considered, are usually expressed in natural language as a note (an “anchor” note in the case of UML). Constraints are often expressed by means of logic expressions (using OCL in the case of UML class diagrams), inside brackets, or also in natural language rather than as part of the model. This happens despite the fact that many researchers have pointed out the convenience of non-functional features appearing in those languages (Cohen et al., 1994; Mylopoulos et al., 1992; Sitaraman, 1994; Wing, 1990; Franch, 1998). As another example, Cysneiros and Leite (2001c) states that, “During 2001 edition of ICSE, Mantis Chen, from ACD System, presented the three most important aspects for a software in the stakeholders’ point of view and the three most important ones in the developers’ point of view. All the six were non-functional requirements.”

This chapter focuses on the possibility of developing mechanisms to capture non-functional information in the development of software systems. It is concerned with the possible extension of the UML (Booch et al., 1998) in order to include non-functionality. The language NoFun (Botella et al., 2001) is taken as the basis to achieve some organization for the non-functional concepts used in this approach. Special attention is given to a standard that defines the quality characteristics of software. The Software Quality Standard ISO/IEC 9126 (International Standard, 1991) is used to obtain the framework for quality measurement and the identification of some important key concepts (characteristics, subcharacteristics, attributes, metrics, etc.).
Related Content

Evaluating UML Using a Generic Quality Framework
[www.irma-international.org/chapter/evaluating-uml-using-generic-quality/30534/](www.irma-international.org/chapter/evaluating-uml-using-generic-quality/30534/)

Using the Semantic Web Rule Language in the Development of Ontology-Driven Applications
[www.irma-international.org/chapter/using-semantic-web-rule-language/35873/](www.irma-international.org/chapter/using-semantic-web-rule-language/35873/)

A Generic Framework for Defining Domain-Specific Models
[www.irma-international.org/chapter/generic-framework-defining-domain-specific/30535/](www.irma-international.org/chapter/generic-framework-defining-domain-specific/30535/)

The Debugging of Rule Bases
[www.irma-international.org/chapter/debugging-rule-bases/35864/](www.irma-international.org/chapter/debugging-rule-bases/35864/)

Using Rule-Based Concepts as Foundation for Higher-Level Agent Architectures
[www.irma-international.org/chapter/using-rule-based-concepts-foundation/35872/](www.irma-international.org/chapter/using-rule-based-concepts-foundation/35872/)