Chapter XVIII

Towards a UML Profile for Building on Top of Running Software

Isabelle Mirbel
Laboratoire I3S, France

Violaine de Rivieres
Amadeus sas, France

ABSTRACT
Currently, fewer and fewer applications are developed from scratch. Therefore, with any development process, it is very important to determine during the analysis and design phases whether there are any applications that must be safeguarded and how this could be accomplished. Legacy applications, as well as Enterprise Resource Planning integration are typical examples of developments that deal with safeguarding. Indeed, safeguarding may be necessary for a specific piece of work involving the integration of new developments with different parts of running applications. To support such a difficult but fundamental task, we recommend a set of extensions through a UML profile. In this proposal, we highlight three aspects of safeguarding which have to be taken into account: the business expertise, the interfaces, and the code itself. We then present how this profile can be used along the different phases of analysis and design; applicable guidelines are provided to support software designers in their daily work.

INTRODUCTION
New applications are constantly built on top of running ones. Legacy applications and Enterprise Resource Planning (ERP) are typical examples of development on top of
running applications or components. In such a context, it is very important to deal with any existing idiosyncrasies in the earliest stages of the development process, mostly during the analysis phase, in order to properly manage the potential risks inherent to this kind of development. Therefore, it is essential to have a clear understanding, as early as possible, what will be safeguarded from the running application, as well as why and how. Integration and interfacing aspects also have to be studied carefully.

We propose a UML profile to support people involved in development; to help them through their daily analysis and design activities. Given the complexity of developing software that will be built on top of a running application, we focus on both the concrete and ad hoc guidelines of the more critical aspects of the development.

The UML profile presented in this chapter is part of the JECKO methodology where a flexible approach is proposed for analysis and design with regards to the application context (Mirbel, 2002a; Mirbel, 2002b). The need for situation-specific approaches, to better satisfy particular situational requirements, has already been emphasized (Van Slooten, 1996; Ralyte, 2001a; Ralyte, 2001b). In JECKO, the application context is described through different criteria; to develop a new application on top of running applications is one such criterion. Flexibility is handled through the different modeling rules and guidelines proposed in each phase of the process. Some rules are useful regardless of the application context while other rules are dedicated to specific criteria. By situating the application in its context and by choosing the interesting modeling rules and guidelines, the process is tailored for the application under consideration in order to allow for a more efficient development process. The profile presented in this chapter is used within the modeling rules and guidelines that are dedicated to applications developed on top of running ones.

The chapter is organized as follows. First, the background of our work is presented. Then, we introduce our dedicated profile and we highlight the different aspects of safeguarding. We also show how this profile is used with the JECKO methodology. Finally, future trends are presented and a conclusion is given.

**BACKGROUND**

Many developments now start from running software rather than from scratch. This considerably changes the way analysis and design has to be handled: different aspects of the running software must be taken into consideration for the future development. In addition to the code itself, functional domain expertise (i.e., expertise taken from the functionalities, data, and screen shots) may also be of interest. Interfaces describing relationships that the running software has with other systems (software, databases, etc.) should also be taken into consideration early on in the software development process. But such a situation is rarely handled in the current analysis and design approaches (Spit, 1995).

Problems related to building on top of running software have been studied before, but only from the implementation point of view (Fowler, 1999; Beck, 1997; Opdyke, 1992). We believe that any issues should already be taken into consideration during the analysis phase. Undeniably, in addition to the code itself, the expertise regarding the functional domain and the interfaces (describing any relationships that the running
Related Content

Labeling XML Documents
www.irma-international.org/chapter/labeling-xml-documents/41502/

Preservation of Data Warehouses: Extending the SIARD System with DWXML Language and Tools
www.irma-international.org/chapter/preservation-data-warehouses/73177/

XML Native Storage and Query Processing
www.irma-international.org/chapter/xml-native-storage-query-processing/41497/

XP2P: A Framework for Fragmenting and Managing XML Data over Structured Peer-to-Peer Networks
www.irma-international.org/chapter/xp2p-framework-fragmenting-managing-xml/41508/

Complexity-Based Evaluation of the Evolution of XML and UML Systems
Ana Isabel Cardoso, Peter Kokol, Mitja Lenic and Rui Gustavo Crespo (2005). Advances in UML and XML-Based Software Evolution (pp. 308-321).
www.irma-international.org/chapter/complexity-based-evaluation-evolution-xml/4941/