Eduard Babkin
Boris Ulitin

Ontology-Based Evolution of Domain-Oriented Languages

Models, Methods and Tools for User Interface Design in General-Purpose Software Systems

Springer

Ontology-Based Evolution of Domain-Oriented Languages

Eduard Babkin • Boris Ulitin

Ontology-Based Evolution of Domain-Oriented Languages

Models, Methods and Tools for User Interface Design in General-Purpose Software Systems

Springer

Eduard Babkin
HSE University
Nizhny Novgorod, Russia

Boris Ulitin
HSE University
Nizhny Novgorod, Russia

ISBN 978-3-031-42201-0 ISBN 978-3-031-42202-7 (eBook)
https://doi.org/10.1007/978-3-031-42202-7

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This work focuses on the notion of domain-specific languages (DSLs) in the context of modelling the structure of interfaces of general-purpose software systems. In this case, the structure refers to the set of objects involved in an interface and the relationships between them. Algorithmic DSLs used to describe the solution of some problem within a domain are not the focus of this book. It is important to note that the DSL approaches used in this work are applicable to the modelling of both software interfaces between the various components of a complex software system and human-machine interfaces (i.e., objects and links displayed on the screen and used by the user). Within the scope of this study, we demonstrate the application and development of these approaches only to improve the efficiency of human-machine interface design as part of general-purpose software systems.

Domain-oriented programming has one main principle. And the principle says that you should always focus on a single mission, for which a specific, specialized programming language has been developed, which is used just to solve this set mission better than all known methods. DSLs are used in a narrow domain, taking into account all its specific points. Their main task is to solve all the problems of the application for which they were created. Therefore, domain-oriented programming is a rather specific area of development. It is tied around some specific activity and with some unique language. In fact, there are a lot of such "microspheres" where a separate DSL is used. In general, domain-specific programming touches areas where general-purpose languages cannot "reach" or where their use is simply inappropriate. Very often, a DSL is used as an addition to the main programming languages, expanding their capabilities and sphere of influence.

It is difficult to define the advantages and disadvantages of a DSL, if only because it functions in very specific domains where it simply would not work out "in a different way." This is the same as evaluating the merits of surgery or dentistry as a field of medicine. On the other hand, like any language, DSLs have similar components in their structure (its semantics and syntax). In addition, based on the domain, the DSL must conform to the domain model. Consequently, we can "dissect" the DSL at the level of objects and relationships by establishing a correspondence between the DSM and the DSL.

This correspondence opens up great opportunities for using various model-based approaches to the development of DSLs. It is these ideas that form the basis of our study and the proposed projection approach to the development of DSLs. The main goal of this study is to demonstrate the possibilities of the projection approach and its flexibility in the context of the DSL evolution. In our case, we use the ontological representation as the initial domain model. This is reasonable both from the point of view of the model-oriented approach (the ontology is a formal artifact and can be used in cross-model transformations) and from the point of view of universality (there are a large number of ontologies with varying degrees of domain detail). In addition, ontologies contain not only a set of objects and relationships between them but also domain constraints, which are essential and must be taken into account in the case of developing domain-specific languages.

Our results are presented in the following form. The introduction (Chap. 1) presents the relevance of the work, the aim and objectives of the research, the scientific novelty, and the theoretical and practical significance of the research, as well as a summary of the contents of the work.

Chapter 2 analyzes the existing classical DSL design and implementation methodology for modelling general-purpose human-machine interfaces in the context of the life cycle of general-purpose software systems. For this purpose, the chapter provides the definition of a DSL and its place in general-purpose software systems. A description of the main stages of the classical life cycle of DSLs and software systems is presented. The shortcomings of existing DSL lifecycle management methods for modelling human-machine interfaces of general-purpose software systems are highlighted.

Based on this analysis, it is concluded that the life cycles of DSLs and general-purpose software systems are not fully consistent. As a consequence, there is a need to develop methods that not only address the shortcomings of the classical approach to software development for modelling general-purpose human-machine interface software systems but also support its evolution in line with the evolution of the subject domain.

Chapter 3 is devoted to an analysis of existing methods and formalisms used in describing the structure of a DSL for modelling general-purpose human-machine interfaces of software systems. In particular, formalizations of artifacts, such as domain semantic model (DSM) and the components of DSL structure, semantics, abstract, and concrete syntax, are considered.

Based on the analysis of classical approaches to the formalization of a DSL for the modeling of general-purpose human-machine interfaces of software systems, a generalized unified model-oriented representation of a DSL for the modeling of general-purpose human-machine interfaces of software systems is formulated. This representation makes it possible to represent each component of the DSL structure as a model and the DSL evolution process as a set of cross-model transformations.

Formation of such a unified representation makes it possible to directly use the results of subject domain analysis (its representation in the form of a DSM) in the process of DSL development for general-purpose software systems' human-machine interface modeling, thereby automating the processes of DSL structure determination, thus eliminating the need for DSL re-creation in case of DSM modification in the process of subject domain evolution.

Chapter 4 provides a detailed description of the proposed new projection-based approach to the development of DSLs for modelling human-machine interfaces of general-purpose software systems, based on the model-based representation of the DSL structure formulated in the previous chapter. In the proposed projection approach, the transition between the different components of the DSL structure takes place by applying a set of cross-model transformation rules that ensure not only that changes made in the DSL are consistent with changes in the subject domain (DSM) but also that the DSL evolution process can be automated.

The resulting set of cross-model transformation rules for organizing DSL evolution based on graph transformations can be applied to DSL design for modelling general-purpose DSLs of software systems in various subject areas. For this purpose, only parts of the rules need to be adapted according to the used subject domain models, while the structure of the rules and their set remain unchanged.

Based on this set of cross-model transformation rules, the chapter presents details of the algorithmization and software implementation of the human-machine interface evolution procedure for general-purpose software systems. The software algorithms developed have been used to implement software systems in the two subject areas described in the next chapter.

Chapter 5 describes and analyzes the software implementation of the human-machine interface evolution of general-purpose software systems as an example of an external DSL in two subject domains: "Software System for University Admissions Office" and "Software System for Resource Allocation of Railway Station." The definition of cross-model transformation rules in ATL language is presented [7] to implement the horizontal and vertical evolution of a DSL for modelling human-machine interfaces of general-purpose software systems. Tools (modules) are developed to support DSL evolution for simulation of general-purpose human-machine interfaces of software systems. The results of the evaluation of the characteristics of existing and proposed software environments are given according to the following criteria: time to modify the DSL and number of lines of code manually inserted during the DSL modification.

Chapter 6 is devoted to the analysis of the subsequent application of the proposed projection approach for more complex systems, namely, decision support systems based on heterogeneous information of decision-makers. In this case, we demonstrate the possibilities of the approach for implementing evolution both at the level of the DSL and the original DSM (a set of criteria and their limitations, adapted to each expert participating in the assessment).

The conclusion (Chap. 7) of the work contains a list of the main results of the research, an assessment of the level of achievement of the objective, as well as suggestions for the further development and practical application of the findings in various subject domains.

Nizhny Novgorod, Russia
June 2023

Eduard Babkin
Boris Ulitin

Acknowledgments

I would like to express my sincere gratitude to Tatiana, whose continuous support and benevolent patience became a solid foundation of my scientific endeavors.

Nizhny Novgorod, Russia Eduard Babkin

Foremost, I would like to express my sincere gratitude to my advisor Prof. Eduard Babkin for the continuous support of my PhD study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me all the time in the research and writing of this thesis. I could not have imagined having a better advisor and mentor for my PhD study.

Besides my advisor, I would like to thank my thesis committee, Prof. Sergey Zykov, Prof. Abdulrab S. Habib, Prof. Robert Pergl, Dr. Sergey Shershakov, and Dr. Rostislav Yavorskiy, for their encouragement, insightful comments, and hard questions.

Last but not the least, I would like to thank my family for giving birth to me in the first place and supporting me throughout my life.

Nizhny Novgorod, Russia Boris Ulitin

The research is supported by grant of the RSF (project № 23-21-00112 “Models and methods to support sustainable development of socio-technical systems in digital transformation under crisis conditions”).

Contents