Model Driven Engineering and Ontology Development

Dragan Gašević · Dragan Djurić · Vladan Devedžić

Model Driven Engineering and Ontology Development

Second Edition

Foreword to First Edition by Bran Selic
Foreword to Second Edition by Jean Bézivin

Springer

Dr. Dragan Gašević
School of Computing and Information Systems
Athabasca University
1 University Drive
Athabasca, AB T9S 3A3
Canada
dgasevic@acm.org

Dr. Dragan Djurić
University of Belgrade
FON – School of Business Administration
Department of Information Systems and
Jove Ilica 154
11000 Belgrade
Serbia
dragandj@gmail.com

Prof. Vladan Devedžić
University of Belgrade
School of Business Administration
Dept. Information Systems & Technologies
Jove Ilica 154
11000 Belgrade
Serbia
devedzic@fon.rs

ISBN 978-3-642-00281-6 e-ISBN 978-3-642-00282-3
DOI 10.1007/978-3-642-00282-3
Springer Dordrecht Heidelberg London New York

Library of Congress Control Number: 2009921153

Cover design: KuenkelLopka GmbH

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To our families

Foreword to the 2nd Edition

In the young history of informatics, this book tells yet another story of connecting the world and the machine. Our discipline continuously attempts to create precise mathematical models of the world around us and of the basic mechanisms of our networked computer systems. These models have different properties that make them more or less appropriate to different goals. Their objective is not only to understand the word, but to help complement it and act on it. The main challenge is the alignment between the business system and the technical platform system. Many organizations struggle to meet their evolving business needs and goals in explicit and precise relation to their underlying technical information system. Shared abstract notations allow descriptions of both situations with common formalisms.

In the sixties and seventies, computer science pioneers proposed to bridge the problem space and the solution space through low-level constructs like procedures. Methods like top-down programming, or step-wise refinement, helped achieving this coupling. In the eighties, the object paradigm was found to be practical and efficient in describing the business side and the technical platform side at the same time. This has certainly triggered more work to study even richer abstractions based on different additional and combined paradigms like rules, relations, events, states, functions, services, and many more. Ways to accommodate multiparadigm systems have been investigated. Many tree-based notations have been proposed with XML technologies. The scope of this book, however, is that of graph-based notations like model-driven engineering or ontology engineering. One of the lessons it teaches is that there may not be a unique ideal abstraction to bridge the world problem space and the machine solution space. On the contrary, we may well have to live with different abstraction frameworks, different representation systems, and different technical or modeling spaces.

While there are some other books on similar subjects, this one is unique for several reasons. Instead of opposing different technologies or schools, it tries to understand them, to characterize them, to compare them, and finally to bridge them so that one may be able to use them in simultaneous or alternative ways when solving real problems. Technologies have to be agilely combined to contribute to solutions in a collaborative way. This

book offers not only high-level conceptual presentations, but also implementation-level coverage of the presented technologies, and even hands-on guidance for their joint or separate applications to practical cases.

The authors take the reader through the entire presentation of the multiple facets of model-driven engineering and ontology engineering. They provide a wonderful pedagogical work that clearly and progressively introduces the main concepts, and their contribution may be recommended as an excellent introductory textbook on both technologies. They give an understanding of why and how these solutions may be concretely used in problem-solving and this itself is of tremendous interest to the researcher, the engineer, or the student that will read the book. Beyond these separate presentations, however, they relate them both conceptually and practically; and this is a complete originality of their contribution. The message is not about a silver bullet revealed, but instead about how different conceptual tools may be wisely selected and applied to achieve optimal solutions. New technologies are arriving on the market at a very rapid pace. It is hard to choose between them. Furthermore, as an organization accumulates assets in its information system, new technologies constantly emerge that seem to make previous ones obsolete. Technology interoperability must now be seriously considered. In this book, the authors have successfully performed a clever balancing act by producing a coherent and comprehensive guide on two technical spaces and a bridging framework that is well-grounded, both conceptually and practically. But the main message is that their method may also be generalized and applied to other technical spaces as well, and I am sure this will provide much inspiration for further work.

Finally, the reader will discover that the authors are presenting important variants of language engineering. Modeling languages and programming languages, general purpose languages, and domain-specific languages are becoming central to software engineering, to data engineering, and to system engineering. We know that the number of computer-based applications that will have to be built for various needs in upcoming decades is exponentially increasing. However, the number of professional computer scientists that will be available to produce these applications will follow a very slow linear progression. The only way out of this difficult situation is to mobilize computer scientists to not directly build the applications, but to provide the numerous domain languages that may guide end users to write precise and verifiable domain code themselves. At the end of this book, the reader will realize that she/he is now much more prepared to face these important new challenges of language engineering.

Nantes, France *Jean Bézivin*
February 2009

Foreword to the 1st Edition

The first time I paid attention to the term "ontology" was in the late 1980s when I was part of an engineering team that was responsible for defining what we would now call a domain-specific modeling language. In our case, the domain was telecommunications software and the purpose of our language was to give system architects the ability to describe the high-level structure of their software in the most direct and most expressive manner possible.

The team members were all experienced designers with deep knowledge of the domain so that we had no trouble putting together the initial list of key language concepts. We knew that we needed to include standard architectural modeling constructs such as components, ports, connectors, and the like. We also wanted our language to be object-oriented, so notions such as class, objects, and inheritance were added to the list. However, soon after this very promising start, all progress ground to a halt. Somehow, the definition of the seemingly trivial fine-grain details of these constructs kept eluding us despite long, passionate, and occasionally acrimonious discussions that can only be compared to medieval theological debates.

It was our good fortune that at that point we met Professor Doug Skuce of the University of Ottawa. He had a method and a tool that helped us develop an explicit ontology for our domain. From that exercise we learned that our difficulties stemmed from the fact that, although we shared a general intuition for the chosen constructs of our language, there were numerous subtle and *unstated* differences in our individual conceptualizations that were a barrier to mutual understanding. Furthermore, we discovered that certain commonly used terms had multiple meanings—all equally valid—but which we had not differentiated adequately, leading to much confusion. Only after we had defined our ontology, which included semi-formal definitions of all key terms and their relationships, were we able to finish our task successfully.

Ever since, I've felt that defining a formal domain ontology is a useful and often necessary step in almost any software project. This is because software deals principally with ideas rather than physical artifacts. Whereas the nature of physical artifacts is generally self-evident, this is not

the case with conceptual entities, which are products of the mind. As we all know, different minds see the same thing differently.

The definition and application of ontologies for developing software systems is a central theme of this book. However, the book is about much more than that. It explains, in a clear and didactic manner, how a variety of recent buzzword developments in software theory and practice (intelligent agents, Model Driven Architecture, metamodeling, etc.) can be combined, and brings us to the threshold of the next step in the evolution of the World Wide Web: the *Semantic Web*. Like the Internet before it, the Semantic Web promises to introduce a significant and qualitatively new phenomenon into our lives. This is because it endows the network of disparate information that is currently accessible on the Internet with *meaning*. Because this meaning can be gleaned and processed *automatically* by software, the Semantic Web opens up the exciting and awe-inducing possibility of a unified global intelligence accessible to all.

In the first half of the book, the authors navigate deftly through the prolific and highly confusing *gemüscht* of technologies, tools, and standards such as XML, RDF, OWL, MDA, and UML, and explain how they relate to each other in the context of the idea of the Semantic Web. They introduce the notion of *modeling spaces*, which provides a conceptually simple yet comprehensive framework for understanding and addressing issues within the domain considered. Using that framework, the second half of the book describes a practical strategy for realizing key elements of the Semantic Web based on existing industry standards.

The book is equally suited to those who merely want to be informed of the relevant technological landscape, to practitioners dealing with concrete problems, and to researchers seeking pointers to potentially fruitful areas of research. The writing is technical yet clear and accessible, and is illustrated throughout with useful and easily digestible examples.

I would also highly recommend this book to sociologists studying the interplay between society and technology. It clearly demonstrates that the core technologies required for constructing the Semantic Web are available and moving forward inexorably. Society must be prepared to deal with something so ripe with potential. We must understand not only how the Semantic Web can be useful but also what dangers lurk within it.

Ottawa, Canada *Bran Selic*
December 2005

Preface

The idea of ontologies emerged in applied artificial intelligence some time ago as a means for sharing knowledge (Gruber 1993). Following the development of ontologies and related Web technologies (e.g., HTML and XML), Tim Berners-Lee, Jim Hendler, and Ora Lassila envisioned the next generation of the Web, called the Semantic Web (Berners-Lee et al. 2001). Being based on ontologies, the Semantic Web has the potential for semantically richer representations of things (e.g., Web pages, applications, and persons) and their relations on the Web, and thus should provide us with more intelligent services. That idea might have initially sounded very futuristic and too enthusiastic, but it has recruited a lot of important players from both academia and industry into very extensive and well-funded research efforts. Today, we have quite impressive results, manifested by standards that have been adopted (RDF and OWL), development frameworks (Jena), best-practice and deployment recommendations, and many applications (e.g., PiggyBank).

Of course, researchers are still facing many challenges in their efforts to realize the full vision of the Semantic Web. Probably the first and most important goal is to persuade many industrial developers and software engineers to use and develop ontologies in their everyday practice. However, ontologies rely on well-defined and semantically powerful concepts in artificial intelligence, such as description logics, reasoning, and rule-based systems. Since software engineers are largely unfamiliar with these concepts, ontologies have a price that must be paid for the benefits that they provide.

Trying to address the above problems, researchers have started exploring the potential of some widely adopted software engineering tools and methodologies for ontology development. Stephen Cranefield did the pioneering research by proposing that UML, a well-known software modeling language, should be used for ontology development (Cranefield 2001a). After him, several researchers have explored further the similarities, differences, and equivalences between UML and ontology languages, as well as the potential of the most recent software engineering initiative called the Model Driven Architecture (MDA), and its accompanying standards, the Meta-Object Facility (MOF) and XML Metadata Interchange (XMI), for ontology development (Baclawski et al. 2002a; Djurić

et al. 2005a; Falkovych et al. 2003). This resulted in the initiation of a process for adopting an MDA-based ontology standard by the Object Management Group (OMG), a software engineering standardization consortium (OMG ODM RFP 2003).

In this book we try to fill the gap in the literature covering the subject of applications of the MDA for ontology development on the Semantic Web. Other books cover either the MDA initiative (Kleppe et al. 2003; Mellor et al. 2003b) or the Semantic Web (i.e., ontology development) only (Fensel 2004; Stuckenschmidt and van Harmelen 2005; Zhong et al. 2003). This book gives a comprehensive overview of both themes, with the main emphasis on how we can employ MDA-related standards to develop Semantic Web ontologies. The book is closely related to the recent OMG initiative for the ODM; it is the first description of that new language.

The book is based on our experience obtained from a series of tutorials entitled "MDA Standards for Ontology Development" that we have given at several international conferences on the Semantic Web (the International Semantic Web Conference, the European Semantic Web Conference, and the International World Wide Web Conference) and on software engineering (the International UML Conference and the International Conference of Web Engineering).

What Happened Since the First Edition?

Since the publication of the first edition of the book, there has been a lot of research done in the areas of the Semantic Web and the MDA. Here, we highlight a few of the most notable initiatives and results:

- The OMG adopted the final specification of the ODM document (OMG ODM 2007). While the ODM is not an official standard yet, the final adopted specification is an important sign signaling the end of the journey in achieving the initial goals set up in the OMG's ODM request for proposals (OMG ODM RFP 2003). Moreover, there have been first tools that implement the emerging standard.
- Several research groups started looking into the research challenges that are going beyond the scope of the ODM. These include exploration of the use of the ODM with software modeling languages; metamodeling of Semantic Web rule languages (e.g., Semantic Web Rule Language and Rule Interchange Format); transforming between rule and constraints used in software engineering and the Semantic Web; and developing ontology reasoners.

- Social technologies and their wide adoption made a significant impact on the research related to the Semantic Web. One of the most notable ones is related to knowledge capture through the use of mechanisms such as collaborative tagging, folksonomies, and social networking. This certainly opens new ways in which ontologies are developed, used, and evolved.
- The software modeling community came closer to the organization and definition of a new software engineering discipline. Model Driven Engineering (MDE) is now a widely adopted term that refers to the area of software modeling, model-driven language definitions (i.e., metamodeling), and model transformations. Looking from this MDE broader perspective, the MDA is just one metamodeling architecture along with the set of OMG standards. Similarly, there is the Eclipse Modeling Framework (EMF), which also provides its own set of languages similar to those of the OMG. As well, the MDE research community very much advanced the area of model transformations and their tooling support. Doubtless, this is also an important step toward the reconciliation of the MDE and Semantic Web paradigms.

Second Edition

The major goal of the second edition is to reflect the above changes and provide an up-to-date perspective on the state of the art in this area. Probably the most notable change is that of the book's title. As explained in the last bulleted point above, we felt that the earlier title *Model Driven Architecture and Ontology Development* does not reflect the scope of the book properly. Therefore, we renamed the book *Model Driven Engineering and Ontology Development*.

Besides these changes, let us also mention that the Semantic Web community has started working on the development of OWL 2, the next version of the Web Ontology Language (OWL). However, the state of the OWL development is still in the early stages at the time of writing this edition (December 2008). This is the reason why OWL 2 considerations and the OWL 2 metamodel definitions are not included in this edition. Knowing that the OWL 2 working documents are also using metamodeling, we are certain that future editions of this book will have a lot of exciting results to report on.

Organization, Structure, and Changes

The book is divided into three parts. Part I covers the basics of both the main topics—ontologies and the MDA. First, Chap. 1 gives a brief overview of the field of knowledge representation in artificial intelligence. Chapters 2 and 3 introduce the main concepts of ontologies, the Semantic Web, standards, applications, tools, and some open research questions. The major changes in Chap. 2 are related to the examples of ontologies for social networking, and explanation of the Social Web principles of knowledge capture. Chapter 4 has probably undergone the biggest changes, in order to broaden its initial focus only on the Model Driven Architecture, and its main standards (the MOF and XMI) and mechanisms (UML profiles). Now this chapter also introduces the basic principles of model driven engineering, discusses other metamodeling platforms, and defines model transformation and constraint languages. Part I is concluded by Chap. 5 with modeling spaces, a conceptual framework defined to provide an easier understanding of approaches to modeling, such as ontologies and MOF-defined modeling languages (UML and the ODM).

Part II is the central part of the book. It starts with Chap. 6, which presents a comprehensive review of several approaches and tools that aim to bridge the gap between ontology development and software engineering methodologies. This chapter also lists the relations between UML and ontology languages, and shows the research results on metamodeling of Semantic Web rule languages. Chapter 7 explains the motivation for the forthcoming OMG ontology development standard for the ODM, and the requirements the standard has to fulfill. Next, Chaps. 8 and 9 describe the current specifications of the OMD and the Ontology UML Profile, respectively. Finally, Chap. 10 analyzes the mappings between MDA-based languages (the ODM and the OUP) and Semantic Web ontology languages.

Part III is dedicated to applications that will support the practical use of languages that conform to the OMG ontology development standard, and to some practical aspects of how to develop ontologies using those MDA-based languages. First, Chap. 11 is a short tutorial showing how to develop ontologies using the OUP in two state-of-the-art UML tools (MagicDraw and Poseidon for UML). The new version of this chapter also introduces the Atlas Transformation Language (ATL) and provides basic information about its tooling support. Chapter 12 describes an implementation of an ontology-building platform called AIR, developed entirely following MDE principles. Chapter 13 discusses two examples of ontologies developed using the OUP and MDA standards. Finally, we added Chap. 14, which provides an insight into the research that goes beyond the scope of the ODM specification and metamodel-based definition of ontology language. It provides insights into model driven engineering of ontology development

tools and reasoners as well as integration of Semantic Web rule languages with software modeling.

Throughout the book, we use many ontologies, UML and other MDA-based models, and transformations between them. In order to allow you to try them out and use them in practice, we have created a Web page containing supplementary resources. You can reach this Web page at http://www.modelingspaces.org. Besides the resources referred to in the book, this Web page contains the slide handouts of the tutorials that we have given at many international conferences.

Acknowledgments

Some of the content of this book has been previously published by organizations that are not part of the Springer group. We thank these organizations for their kind permission to use the material for this book; in particular:

- IEEE Press,
- Rinton Press,
- Journal of Educational Technology & Society, and
- Journal of Object Technology.

We are very grateful to Jelena Jovanović, Nenad Krdžavac, and Colin Knight for a huge amount of help in preparing the first edition of the book and for giving us many useful comments and suggestions. We very much appreciate the great help of Milan Milanović in preparting this second revision. We are also very thankful to Adrian Giurca, Sergey Lukichev, Marko Ribarić, and Gerd Wagner for great research collaboration that very much inspired changes made in this edition. We would also like to thank all the members of the GOOD OLD AI research group (http://goodoldai.org.yu) at the University of Belgrade, Serbia and Montenegro, for providing us with a great environment and many useful ideas that enormously improved the quality of the book. Dragan Gašević is very grateful to Tom Calvert, Marek Hatala, Kinshuk, Rory McGreal, and Griff Richards for their great support and understanding. In addition, Dragan Gašević is very grateful to the Natural Sciences and Engineering Research Council of Canada, the Alberta Ingenuity Fund, and Athabasca University, for their continuing research funding support.

We thank Bran Selic for agreeing to write the book's foreword. A special mention should be given to Ralf Gerstner of Springer, who invited us to write this book, motivated us to work on its revision, and had a lot of

patience in working with us. We are very thankful to Evan Wallace, Elisa Kendall, Bob Colomb, Anna Gerber, Dan Chang, Lewis Hart, and other colleagues from the OMG's Ontology PSIG for giving us up-to-date information about the OMG's ODM standardization efforts in the last six years.

Vancouver, Canada *Dragan Gašević*
Belgrade, Serbia *Dragan Djurić*
Belgrade, Serbia *Vladan Devedžić*
December 2008

Contents