Babeş-Bolyai University
Faculty of Mathematics and Computer Science

Model-Based Code Generation

- PhD Thesis Summary -

Scientific Supervisor:
Prof. Dr. Bazil Pârv

Author:
Paul Horaţiu Stan

February 2012

Contents

List of publications

Published papers (journals)

[107] Paul Horaţiu Stan. A proposed DSL for data intensive application development. *Studia UBB, Informatica*, LVII(1):accepted, 2012.

[103] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. *Studia UBB, Informatica*, LVI(3):9–14, 2011.

[108] Paul Horaţiu Stan and Camelia Şerban. A proposed approach for platform interoperability. *Studia UBB, Informatica*, LV(2):87–98, 2010.

Published papers (proceedings)

[104] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. In *Proc KEPT 2011 (eds: M. Frentiu, HF Pop, S. Motogna)*, pages 247–258, ISSN 2067-1180, Cluj University Press, 2011 (ISI Proc).

[102] Paul Horaţiu Stan. A proposed technique for component-based software development. In *Zilele Academice Clujene*, pages 52–56, 2010.

Submitted papers

[105] Paul Horaţiu Stan. Case study: An order processing system developed using DevDSL. Submitted to: *16th East European Conference on Advances in Databases and Information Systems*, Poznan, September 18-21, 2012.

[106] Paul Horaţiu Stan. A custom validation mechanism for DevDSL. Submitted to: *Studia UBB, Informatica*, LVII(1):, 2012.

List of keywords

Model Driven Engineering, Model Driven Architecture, Domain Specific Modeling, Domain Specific Languages, Automatic Code Generation, Component-based development, Platform Independent Model, Platform Specific Model, Model Transformations, General Programming Languages.

Summary

This Ph.D. thesis is the result of my research in the field of Model Driven Engineering (MDE), particularly in Model Based Code Generation (MBCG), research which was started in 2008 under the supervision of Prof. Dr. Bazil Pârv.

Motivation

A simple and easy question is: *Why the thesis title is Model Based Code Generation?* The answers are:

- *MBCG* can be viewed as a subsection of MDE focused on model transformation engines. The source metamodel is the modeling language used to specify the platform-independent model (PIM) of a system and the target metamodel is the syntax of the programming language for which the engine will generate the source code, representing the platform-specific model (PSM) of the system.

- *MBCG* can improve the software development time due to the automated source code generation.

- *MBCG* can increase the software quality because the codding errors are eliminated.

- *MBCG* is an open research space, there are many articles, books and conferences focused on this domain.

- *MBCG* is the next step in software engineering and compilation. At the beginings the human resources were forced to understand and to write programs in the computer languages, the second step introduced the assembly language which assigned a suggestive word to each computer instruction, at the third step programming languages and compilers were introduced, based on the language syntax the compiler checks a program and generates computer statements. MBCG can be viewed as the next step on this natural way, the models specified in a platform independent way using domain specific languages (DSLs) are checked and translated into general programming language (GPL) source code files.

- Using *MBCG* it is possible to reduce the gap between technical people and customers. A domain specific language oriented on the client problem can be designed and used in order to collect client requirements in a formal and non ambiguous way.

Research objectives

In the context on current open research problems in the field of MDE, the research objectives of this thesis were focused on automatic code generation based on application models. The following enumeration presents the main objectives:

- A proposed architecture for reusing PSMs in another platforms than the platforms in which has been designed to be used. The proposed approach uses a generated proxy object [108] which acts as a mediator between the new PSM and the old one. Eah proxy object delegates the method calls to the old PSM.

- The proposal of a component-based development process which automate the source code generation. There are two aspects: (1) the generation of the skeleton source code of a component, not containg the business logic of a component, and (2) the generation of the source code for connecting components [102, 103, 104].

- The proposal of a DSL for data intensive applications which allows the automatic translation to different PSM, web oriented or not [107, 106, 105].

Each objective is described in detail in a dedicated chapter in a common way, first it is presented the problem, the second step describes the proposed solution, a case study or a list of case studies is presented at the third step and finally conclusions and future improvements or open problems are described.

Chapter 2 presents an overview about Model Driven Architecture [4, 116, 8, 87, 59, 114, 90, 22, 12, 91, 27, 33, 70]. The Model Drive Architecture targets are [116]: Technology obsolescence, Portability, Productivity and time-to-market, Quality, Integration, Maintenance, Testing and simulation and Return on investment. This chapter enumerates the major MDA concepts [4] such as: System, Model, Model driven, Architecture, Viewpoint, MDA viewpoints, Platform, Platform Independence, Platform Model, Model Transformation, Implementation, Computation Independent Model (CIM), Platform Independent Model (PIM) [26] and Platform Specific Model (PSM). OMG provided a general specifications, and as answers, MDA tools has been developed by research teams. There are many types of MDA tools, such as: [116]: Creation Tool, Analysis Tool, Transformation Tool, Composition Tool, Test Tool, Simulation Tool, Metadata Management Tool and Reverse Engineering Tool.

The basic MDA flow consists in defining a platform independent model (PIM) [26] and transforming it automatically to one or more platform-specific models (PSMs) [28]. Model transformation domain can be divided into [28]: (1) Model to text transformations and (2) Text to model transformations. At the end of this chapter a list of standards used by the MDA is presented, such as: XML, XMI, UML, MOF, MDDA, SCA, BPMN [8, 38, 23, 97, 125].

Chapter 3 presents a set of MDA tools and frameworks used in the research presented in this thesis. The presentation is structured into two main sections: (1) Academic Tools and (2) Industrial Tools. Academic MDA Tools section contains instruments developed by research teams from different universities and Industrial MDA Tools sections shows productions IDE's and frameworks that are used by software development companies.

Personal contributions are presented into Chapters 4, 5 and 6.

- Chapter 4 *"An approach for platform interoperability based on proxy objects"*: presents a technique for platform interoperability based on model transformation approaches. Instead of serializing objects for implementing the communication between platforms, the proposed approach involves generating proxy objects for the remote objects. The proxy objects types are generated automatically. The source code written for implementing the remote object functionality acts as a model that conforms to a metamodel (the programming language syntax), the proxy object's type acts as a model that conforms to another metamodel (the syntax of the programming language used for implementing the proxy object). As a conclusion, the technique presented in this chapter can be integrated in the model transformation area contributions. The original solution is described in greater detail in [108].

- Chapter 5 *"A solution fo component-based developing using automatic code generation for component connections"* proposes a technique for component-based software development. The novelty of the proposed solution resides in each component has public properties that can have one of the following directions: IN, OUT and INOUT, and the communication between components is implemented using these properties, called pins. As a result of the proposed technique, the components are not dependent on each another. The components are only dependent on their pins data types. The original contributions are presented in more detail in the papers [104, 102, 103].

- Chapter 6: *"A domain-specific language for the development of data-intensive applications"* presents a solution which involves (1) a language for defining the platform-independent model of an application and (2) a transformation engine which support translation from the platform-independent model to .NET web application model. Both packages are included into an Eclipse plug-in with code completion and syntax highlight. The original contributions from this chapter are described in more detail in the papers [107, 105, 106].

Chapter 4 presents a new approach regarding interoperability. The novelty of the proposed solution resides in using a proxy object for each used remote object in order to avoid serialization.

An actual software engineering problem is the improvement of the software development process and the final product quality [43]. In order to achive this, the source code should be reusable [73]. Obviously, it is good news for a developer if an old library developed in an old programming language can be used in a new programming language without being necessary to rewrite the code. In this case, it saves time and improves the quality of the final software system, because the old library has been tested in the past and it works fine.

In the context of model transformation, one target of this PhD thesis was to determine an algorithm for generating the source code for a specific programming language based on a source code written for another programming language in an automatically mode. This is not a translation of the source code from a programming language to another, the autogenerated code acts as a proxy to the real code, so it delegates the execution to the real routines. This research can be included into "Model Transformation" area because it transforms an executable model resulted from compiling a source code written for a programming language to a source code written in another programming language which acts as a proxy to the initial executable model.

There are many frameworks written for both Java and .NET, for instance: Hibernate [25] respectively NHibernate, JUnit [80] respectively NUnit etc. These frameworks could be written for a platform and reused from another platform, for instance instead, of rewriting Hibernate for .NET it can be reused directly. Chapter 5 uses the theoretical concepts presented in Chapter 4 in a demo application presentation. The application manages entities from a library such as: books, authors and members. The conceptual model of the application is written in .NET, but the database component of this application is written in Java, then it have to use proxy objects written in Java for the real library model which is written in .NET. The beauty of the solution resides in offering the posibility to save and load .NET objects using the Java version of Hibernate and Java proxy objects for the real .NET objects. It is not necessary to use the NHibernate framework.

Chapter 4 is structured as follows: the first section presents the problem statement together with: (1) the motivation of solving it, (2) current approaches and (3) their drawbacks, the second presents the proposed approach and a proof of concept has been presented in section three. Section four applies two evaluation metrics for the proposed code generation solution. Finally, contributions of this work are presented together with future improvements and open problems.

Chapter 5 presents a technique inspired from hardware development that can be applied in software engineering in order to develop flexible and modular systems based on independent components. On the other hand, it presents the advantages and constraints of this technique and how to implement it in an actual development platform, for instance Java and .NET.

In the hardware development process, an engineer can use many integrated circuits in order to develop a complex system, for instance he/she can use multiplexers, counters, logic AND, logic OR, memory, register [76, 54, 85, 61, 52, 71, 50, 129, 93] etc, these components are interconnected on a main board and become a complex system. The counter, the multiplexer or other components are developed by a third party company, and they are not dependent by components that produce/consume their inputs/outputs. This principle can be applied both on the hardware and software development process, for example a software system can have many independent components that should be interconnected on a main board in order to become a complex system. These components can be developed by a third party company. Nowadays there are many third party components that are used in software systems, but the connections between them are made by a developer, and when a person should write a program that will use the third party components, this can introduce many errors, more than this, changing the code written by a developer is an expensive process that needs time and money.

This approach presents a technique that allows the automatic code generation for connecting different objects. This technique implies constraints for object classes in order to support the connecting process. This chapter combines the presented theoretical concepts with the notions introduced in Chapter 4 in order to exemplify the benefits of the proposed solutions. The main benefit of the solution proposed in Chapter 5 is: a software component design technique which allows the posibility to write independent components that will be automatically connected using a source code generator. The development process has three main steps: (1) design the system in a graphical environment, (2) generate the skeleton source code and (3) implement the component functionalities.

Chapter 5 is structured as follows: the first section presents the open problem in three subsections: (1) Problem motivation, (2) Current approaches and (3) drawbacks of current approaches; section 2 shows the proposed solution in four subsections: (1) Conceptual View, (2) Glossary, (2) Architecture and (3) How it works; section 3 presents the *Building Blocks Dev Studio* into four subsections: (1) the main architecture, (2) component communication mechanism, (3) a component-based development approach based on the proposed solution and (4) a case study which contains a demo application developed using the *Building Blocks Dev Studio* and *Indep* tools [103, 104]; the fourth section presents two evaluation metrics for the proposed solution and finally section five summarizes the contributions of this work and future improvements.

Chapter 6 presents my research during the three months of mobility to the University of Debrecen from October to December 2010 [107, 105, 106]. Together with professor Adamko Attila, we developed a domain-specific language (DSL) for data intensive application. The proof of concept resides in an Eclipse plug-in which can be used in order to specify applications and to transform them to .NET web applications. The benefits of the proposed DSL resides in the possibility to write transformations to many platform specific models not only to web. Finally a comparison with WebML [109, 86, 123, 98, 79, 17, 21, 133] and WebDSL [29, 56, 48, 49, 47] is presented.

The research presents a DSL for specifying data intensive applications. Using the proposed DSL a PIM of an application can be defined, then using transformations to given PSM the source code can be automatically generated. The DSL has been implemented using Eclipse Xtext, XPand and MWE2, based on documentations from [40, 81, 64, 74].

The terms PIM and PSM [26] are most frequently used in the context of the MDA approach. The key concept is that it should be possible to use a MTL [35, 69, 82, 131] to transform a PIM into a PSM [112, 31, 115, 82, 131].

This chapter proposes two transformation engines: (1) a PIM to .NET web applications engine and (2) a PIM to PHP Code Igniter applications engine. These engines are proofs of concept presented into the Chapter 6, they demonstrate the possibility to transform the platform independent model written using the proposed DSL to many platform specific models.

Chapter 6 is structured as follows: section 1 presents the open problem in two subsections: (1) problem motivation and (2) drawbacks of current DSLs for data intensive applications; section 2 shows the technical details of the proposed solution into four subsections: (1) conceptual view of the solution, (2) technical details, (3) how the solution works and (4) a comparison with WebML and WebDSL. The *DevDSL* Eclipse plugin, which is a proof of theoretical concepts, is presented in section 3 with the following subsections: (1) the main architecture, (2) how it works; section four presents a list of case studies which contains two demo applications [105] developed using the proposed DSL, a custom transformation engine to PHP Code Igniter framework and a custom validation mechanism for user inputs [106].

Finally, Chapter 7 presents the conclusions and future work. The intent of MDE are: (1) improve software quality; (2) reduce the development time and budget by using artifacts generators engines such as code generators, test cases generators etc; (3) reduce complexity, and (4) improve reuse by enabling developers to work at higher levels of abstraction and to ignore irrelevant details [7]. In this context, current research has been focused on three open problems and proposed conceptual and technical solutions.

Each of the proposed solutions are accompanied by developed tools. The proposed tools are proof of concepts presented as personal contributions. The following list presents a short description of them:

- Indep, presented in Chapter 4 is a framework developed in order to use Java objects in .NET and vice–versa. This framework is a proof of platform interoperability concept presented in [108]. Indep framework provides: (1) the source code generator for proxy objects and (2) the communication infrastructure between Java and .NET.

- Building Blocks Dev Studio, presented in Chapter 5 is a stand alone application developed in .NET. It is a proof of component-based development concept presented in [102], and can be used together with *Indep* for developing component-based applications with components written in Java and .NET. The software solution has:

(1) a graphical user interface environment for designing the components and links between them and (2) an automatic code generator engine for components skeletons.

- DevDSL is an Eclipse plugin which implements the theoretical concepts presented in [107]. This plugin offers: (1) a specification module which can be used in order to write the application model and (2) a transformation engine which is used in order to generate the .NET source code from the platform independent model written with the specification module.

I would like to express my gratitude to my supervisor, professor Bazil Pârv, for his competent guidance, suggestions and encouragement. I would like to thank to professor Adamko Attila, professor Simona Motogna, professor Militon Frentiu, professor Ioan Laz(ă)r and to all my collaborators for their precious help and guidance. I am also grateful to all the members of Computer Science Department for their helpful suggestions.

I wish to thank for the financial support provided from programs co-financed by the SECTORAL OPERATIONAL PROGRAMME HUMAN RESOURCES DEVELOPMENT, Contract **POSDRU 6/1.5/S/3** – "Doctoral studies: through science towards society".

Bibliography

[1] *Eclipse development using the graphical editing framework and the eclipse modeling framework*. IBM Corp., Riverton, NJ, USA, 2004.

[2] Nour Ali, Rukmani Nellipaiappan, Rajalaxmi Chandran, and Muhammad Ali Babar. Model driven support for the service oriented architecture modeling language. In *Proceedings of the 2nd International Workshop on Principles of Engineering Service-Oriented Systems*, PESOS '10, pages 8–14, New York, NY, USA, 2010. ACM.

[3] Yusuf Altunel and Mehmet R. Tolun. Component-based software development with component variants. In *Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering*, pages 235–241, Anaheim, CA, USA, 2007. ACTA Press.

[4] Wim Bast Anneke Kleppe, Jos Warmer. *MDA Explained*. Addison-Wesley Professional, 2003.

[5] Apache. Apache velocity home. http://velocity.apache.org. Last accessed on May 03,2011.

[6] ATL. Atlas transformation language home. http://www.eclipse.org/m2m/atl/. Last accessed on May 03,2011.

[7] P. Tarr B. Hailpern. Model-driven development: The good, the bad, and the ugly. *IBM Systems Journal*, 45(3):451–461, 2006.

[8] L Balmelli B Nolan, B Brown and T Bohn. *Model Driven Systems Development with Rational Products*. IBM, 2008.

[9] A. Basukoski, P. Buhler, V. Getov, S. Isaiadis, and T. Weigold. Methodology for component-based development of grid applications. In *Proceedings of the 2008 compFrame/HPC-GECO workshop on Component based high performance*, CBHPC '08, pages 1:1–1:6, New York, NY, USA, 2008. ACM.

[10] Don Batory, Clay Johnson, Bob MacDonald, and Dale von Heeder. Achieving extensibility through product-lines and domain-specific languages: a case study. *ACM Trans. Softw. Eng. Methodol.*, 11:191–214, April 2002.

[11] Shahid Nazir Bhatti and Asif Muhammad Malik. An xml-based framework for bidirectional transformation in model-driven architecture (MDA). *SIGSOFT Softw. Eng. Notes*, 34:1–5, May 2009.

[12] Stefan Biffl, Richard Mordinyi, and Alexander Schatten. A model-driven architecture approach using explicit stakeholder quality requirement models for building dependable information systems. In *Proceedings of the 5th International Workshop on Software Quality*, WoSQ '07, page 6, Washington, DC, USA, 2007. IEEE Computer Society.

[13] Anna Gerber Bill Moore, David Dean and Gunnar Wagenknecht. *Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework.* IBM, 2004.

[14] G. Blaschek. *Object-Oriented Programming with Prototypes.* Verlag-Springer, 1994. New York.

[15] Grady Booch. The promise the limitsthe beauty of software. http://intranet.cs.man.ac.uk/Events_subweb/special/turing07/PowerPoint _Complete/PPP.pdf, 2007.

[16] Paul Boocock. JaMDA: The java model driven architecture. http://jaMDA.sourceforge.net, 2003. Last accessed on July 13, 2011.

[17] Marco Brambilla. Generation of WebML web application models from business process specifications. In *Proceedings of the 6th international conference on Web engineering*, ICWE '06, pages 85–86, New York, NY, USA, 2006. ACM.

[18] Frank Budinsky, Stephen A. Brodsky, and Ed Merks. *Eclipse Modeling Framework.* Pearson Education, 2003.

[19] Emmanuel Cecchet, Julie Marguerite, and Willy Zwaenepoel. Performance and scalability of ejb applications. In *Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, OOPSLA '02, pages 246–261, New York, NY, USA, 2002. ACM.

[20] Ethan Cerami. *Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI and WSDL.* O'Reilly Media, Inc., 2002.

[21] Stefano Ceri, Marco Brambilla, and Piero Fraternali. Conceptual modeling: Foundations and applications. chapter The History of WebML Lessons Learned from 10

Years of Model-Driven Development of Web Applications, pages 273–292. Springer-Verlag, Berlin, Heidelberg, 2009.

[22] Kenneth Chan and Iman Poernomo. Qos-aware model driven architecture through the uml and cim. *Information Systems Frontiers*, 9:209–224, July 2007.

[23] David Chappell. Introducing sca. http://www.davidchappell.com/articles/ introducing_sca.pdf, 2007. Last accessed on July 14, 2011.

[24] Hyun Cho and Jeff Gray. A domain-specific modeling language for scientific data composition and interoperability. In *Proceedings of the 48th Annual Southeast Regional Conference*, ACM SE '10, pages 107:1–107:4, New York, NY, USA, 2010. ACM.

[25] Gavin King Christian Bauer. *Hibernate in Action: Practical Object/Relational Mapping.* Manning Publications, 2004.

[26] Nuno Amálio Christian Glodt, Pierre Kelsen and Qin Ma. From platform-independent to platform-specific models using democles. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 795–796, 2009.

[27] Stephen Cranefield and Jin Pan. Bridging the gap between the model-driven architecture and ontology engineering. *Int. J. Hum.-Comput. Stud.*, 65:595–609, July 2007.

[28] Krzysztof Czarnecki and Simon Helsen. Classification of model transformation approaches. In *Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.

[29] Eelco Visser Danny M. Groenewegen. Weaving web applications with WebDSL: (demonstration). In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 797–798, 2009.

[30] Andrew J. Kornecki David P. Gluch. Automated code generation for safety related applications: A case study. In *Proceedings of the International Multiconference on Computer Science and Information Technology*, pages 383–391, 2006.

[31] Zekai Demirezen. Semantic framework for dsls. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 833–834, 2009.

[32] Tom Dinkelaker and Mira Mezini. Dynamically linked domain-specific extensions for advice languages. In *Proceedings of the 2008 AOSD workshop on Domain-specific aspect languages*, DSAL '08, pages 3:1–3:7, New York, NY, USA, 2008. ACM.

[33] Mouhamed Diouf, Sofian Maabout, and Kaninda Musumbu. Merging model driven architecture and semantic web for business rules generation. In *Proceedings of the 1st international conference on Web reasoning and rule systems*, RR'07, pages 118–132, Berlin, Heidelberg, 2007. Springer-Verlag.

[34] Bruce Eckel. *Thinking in Java*. MindView, Inc, 2004.

[35] Eclipse. Model to text. http://www.eclipse.org/modeling/m2t, 2010. Last accessed on July 13, 2011.

[36] Elisa & Domain Orientation Case study: An order processing system http://jklunder.home.xs4all.nl/elisa/part04/doc070.html Last accessed on January 24, 2012.

[37] Ralph Johnson Erich Gamma, Richard Helm and John Vlissides. *Design Patterns: 50 specific ways to improve your use of the standard template library*. Pearson Education, Inc, 1995.

[38] Eric Evans. *Domain-Driven Design Quickly*. C4Media Inc, 2006.

[39] Heiko Behrens Michael Clay Sven Efftinge Moritz Eysholdt and contributors. *Xtext User Guide*. XText, 2010.

[40] Moritz Eysholdt and Heiko Behrens. Xtext: implement your language faster than the quick and dirty way. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, SPLASH '10, pages 307–309, New York, NY, USA, 2010. ACM.

[41] Lidia Fuentes, Mónica Pinto, and Pablo Sánchez. Generating CAM aspect-oriented architectures using model-driven development. *Inf. Softw. Technol.*, 50:1248–1265, November 2008.

[42] Dragan Gaaevic, Dragan Djuric, Vladan Devedzic, and Bran Selic. *Model Driven Architecture and Ontology Development*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[43] Mark Gabel, Junfeng Yang, Yuan Yu, Moises Goldszmidt, and Zhendong Su. Scalable and systematic detection of buggy inconsistencies in source code. In *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*, OOPSLA '10, pages 175–190, New York, NY, USA, 2010. ACM.

[44] Nasib S. Gill. Reusability issues in component-based development. *SIGSOFT Softw. Eng. Notes*, 28:4–4, July 2003.

[45] Steffen Goebel and Michael Nestler. Composite component support for ejb. In *Proceedings of the winter international synposium on Information and communication technologies*, WISICT '04, pages 1–6. Trinity College Dublin, 2004.

[46] Adam Griffith. *CodeIgniter 1.7 professional development*. Packt Publishing, 2010.

[47] Danny Groenewegen, Zef Hemel, and Eelco Visser. Separation of concerns and linguistic integration in WebDSL. *IEEE Softw.*, 27:31–37, September 2010.

[48] Danny M. Groenewegen, Zef Hemel, Lennart C.L. Kats, and Eelco Visser. WebDSL: a domain-specific language for dynamic web applications. In *Companion to the 23rd ACM SIGPLAN conference on Object-oriented programming systems languages and applications*, OOPSLA Companion '08, pages 779–780, New York, NY, USA, 2008. ACM.

[49] Danny M. Groenewegen and Eelco Visser. Weaving web applications with WebDSL: (demonstration). In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, OOPSLA '09, pages 797–798, New York, NY, USA, 2009. ACM.

[50] Thomas R. Gross, John L. Hennessy, Steven A. Przybylski, and Christopher Rowen. Measurement and evaluation of the mips architecture and processor. *ACM Trans. Comput. Syst.*, 6:229–257, August 1988.

[51] William Grosso. *Java RMI (Java Series)*. O'Reilly Media, Inc, 2001.

[52] M. Gschwind and D. Maurer. An extendable mips-i processor kernel in vhdl for hardware/software co-design. In *Proceedings of the conference on European design automation*, EURO-DAC '96/EURO-VHDL '96, pages 548–553, Los Alamitos, CA, USA, 1996. IEEE Computer Society Press.

[53] Stuart Hansen and Timothy V. Fossum. Refactoring model-view-controller. *J. Comput. Small Coll.*, 21:120–129, October 2005.

[54] Joseph Heinrich. *MIPS R4000 user's manual*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[55] Zef Hemel. The point of WebDSL. http://zef.me/tag/WebDSL. Last accessed on December 22, 2010.

[56] Zef Hemel, Danny M. Groenewegen, Lennart C. L. Kats, and Eelco Visser. Static consistency checking of web applications with WebDSL. *J. Symb. Comput.*, 46:150–182, February 2011.

[57] Felienne Hermans, Martin Pinzger, and Arie Deursen. Domain-specific languages in practice: A user study on the success factors. In *Proceedings of the 12th International Conference on Model Driven Engineering Languages and Systems*, MODELS '09, pages 423–437, Berlin, Heidelberg, 2009. Springer-Verlag.

[58] Wolfram Hohmann. Supporting modelbased development with unambiguous specifications. In *SAE World Congress, Detroit, MI*, 2004.

[59] Cuiyun Hu, Xinjun Mao, and Hong Ning. Integrating model transformation in agent-oriented software engineering. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*, WI-IAT '09, pages 62–65, Washington, DC, USA, 2009. IEEE Computer Society.

[60] Shan Shan Huang, David Zook, and Yannis Smaragdakis. Domain-specific languages and program generation with meta-aspectj. *ACM Trans. Softw. Eng. Methodol.*, 18:6:1–6:32, November 2008.

[61] Muhammad Jahangir Ikram. A configurable mips simulator for teaching computer architecture. In *Proceedings of the 10th IASTED International Conference on Computers and Advanced Technology in Education*, pages 91–95, Anaheim, CA, USA, 2007. ACTA Press.

[62] Kyungsoo Im, Tacksoo Im, and John D. McGregor. Automating test case definition using a domain specific language. In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ACM-SE 46, pages 180–185, New York, NY, USA, 2008. ACM.

[63] Mario Rammer Ingo Szpuszta. *Advanced .NET Remoting 2nd Edition*. APRESS, 2005.

[64] Jeronimo Irazabal and Claudia Pons. Supporting modularization in textual dsl development. In *Proceedings of the 2010 XXIX International Conference of the Chilean Computer Science Society*, SCCC '10, pages 124–130, Washington, DC, USA, 2010. IEEE Computer Society.

[65] Hemant Jain, Padmal Vitharana, and Fatemah "Mariam" Zahedi. An assessment model for requirements identification in component-based software development. *SIGMIS Database*, 34:48–63, November 2003.

[66] Pierre-Alain Muller Jean B ezivin, S ebastien G erard and Laurent Rioux. MDA components: Challenges and opportunities. In *Metamodelling for MDA, First International Workshop York, UK, Proceedings*, pages 23–41, 2003.

[67] Jean-Marc Jézéquel, Olivier Barais, and Franck Fleurey. Model driven language engineering with kermeta. In *Proceedings of the 3rd international summer school conference on Generative and transformational techniques in software engineering III*, GTTSE'09, pages 201–221, Berlin, Heidelberg, 2011. Springer-Verlag.

[68] Rod Johnson, Juergen Hoeller, Alef Arendsen, Thomas Risberg, and Dmitriy Kopylenko. *Professional Java Development with the Spring Framework*. Wrox Press Ltd., Birmingham, UK, UK, 2005.

[69] Steven Kelly Juha-Pekka Tolvanen. Metaedit+: defining and using integrated domain-specific modeling languages. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 819–820, 2009.

[70] Salah Kabanda and Mathew Adigun. Extending model driven architecture benefits to requirements engineering. In *Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, SAICSIT '06, pages 22–30, , Republic of South Africa, 2006. South African Institute for Computer Scientists and Information Technologists.

[71] Gerry Kane and Joe Heinrich. *MIPS RISC architectures*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1992.

[72] Lennart C. L. Kats, Karl T. Kalleberg, and Eelco Visser. Domain-specific languages for composable editor plugins. *Electron. Notes Theor. Comput. Sci.*, 253:149–163, September 2010.

[73] Beck Kent. *Test Driven Development: By Example*. Addison-Wesley Professional, 2003.

[74] Christian Köllner, Georg Dummer, Andreas Rentschler, and K. D. Müller-Glaser. Designing a graphical domain-specific modelling language targeting a filter-based data analysis framework. In *Proceedings of the 2010 13th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, ISORCW '10, pages 152–157, Washington, DC, USA, 2010. IEEE Computer Society.

[75] Toma Kosar, Pablo E. Martinez Lopez, Pablo A. Barrientos, and Marjan Mernik. A preliminary study on various implementation approaches of domain-specific language. *Inf. Softw. Technol.*, 50:390–405, April 2008.

[76] Stephen Kou and Jens Palsberg. From oo to fpga: fitting round objects into square hardware? In *Proceedings of the ACM international conference on Object oriented*

programming systems languages and applications, OOPSLA '10, pages 109–124, New York, NY, USA, 2010. ACM.

[77] Avraham Leff and James T. Rayfield. Web-application development using the model/view/controller design pattern. In *Proceedings of the 5th IEEE International Conference on Enterprise Distributed Object Computing*, EDOC '01, pages 118, Washington, DC, USA, 2001. IEEE Computer Society.

[78] H. Lieberman. Using prototypical objects to implement shared behavior in objectoriented systems. *Meyrowitz Proceedings*, pages 214–223, 1986.

[79] David Lowe and Rachatrin Tongrungrojana. WebML+ for communication of information flows: an empirical study. In *Proceedings of the 2003 international conference on Web engineering*, ICWE'03, pages 218–221, Berlin, Heidelberg, 2003. Springer-Verlag.

[80] Vincent Massol. *JUnit in Action*. Manning Publications, 2003.

[81] Bernhard Merkle. Textual modeling tools: overview and comparison of language workbenches. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, SPLASH '10, pages 139–148, New York, NY, USA, 2010. ACM.

[82] Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. *ACM Comput. Surv.*, 37:316–344, December 2005.

[83] Microsoft. Introduction to COM interop (visual basic). http://msdn.microsoft.com/en-us/library/kew41ycz.aspx. Last accessed on July 14, 2011.

[84] Mips architecture pictures. http://withfriendship.com/user/servex/mips-architecture.php, 2011. Last accessed on January 25, 2012.

[85] Sunil Mirapuri, Michael Woodacre, and Nader Vasseghi. The mips r4000 processor. *IEEE Micro*, 12:10–22, March 1992.

[86] Nathalie Moreno, Piero Fraternalli, and Antonio Vallecillo. A uml 2.0 profile for WebML modeling. In *Workshop proceedings of the sixth international conference on Web engineering*, ICWE '06, New York, NY, USA, 2006. ACM.

[87] OMG. MOF model to text transformation language, 1.0. http://www.omg.org/spec/MOFM2T/1.0/, 2010. Last accessed on July 13, 2011.

[88] Oracle. Java rmi over iiop. http://download.oracle.com/javase/1.4.2/docs/guide/rmi-iiop. Last accessed on July 14, 2011.

[89] Turhan Ozgur. *Domain-Specific Modeling: A Practical Approach A comparison of Microsoft DSL Tools and Eclipse Modeling Frameworks in the context of Model-Driven Development.* LAP Lambert Academic Publishing, Germany, 2009.

[90] Claus Pahl. Semantic model driven development of web service architectures. *Int. J. Web Eng. Technol.*, 4:386–404, July 2008.

[91] Daniel Perovich, Maria Cecilia Bastarrica, and Cristian Rojas. Model-driven approach to software architecture design. In *Proceedings of the 2009 ICSE Workshop on Sharing and Reusing Architectural Knowledge*, SHARK '09, pages 1–8, Washington, DC, USA, 2009. IEEE Computer Society.

[92] Tracy Gardner Peter Swithinbank, Mandy Chessell and Catherine Griffin. *Patterns: Model-Driven Development Using IBM Rational Software Architect.* IBM, 2005.

[93] Nathaniel Pinckney, Thomas Barr, Michael Dayringer, Matthew McKnett, Nan Jiang, Carl Nygaard, David Money Harris, Joel Stanley, and Braden Phillips. A mips r2000 implementation. In *Proceedings of the 45th annual Design Automation Conference*, DAC '08, pages 102–107, New York, NY, USA, 2008. ACM.

[94] Roman Popp. Defining communication in soa based on discourse models. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 829–830, 2009.

[95] Bartosz Porebski, Karol Przystalski, and Leszek Nowak. *Building PHP Applications with Symfony, CakePHP, and Zend Framework.* Wrox Press Ltd., Birmingham, UK, UK, 1st edition, 2011.

[96] Steven Sanderson. *ASP.NET MVC Framework Preview.* Apress, Berkely, CA, USA, 1 edition, 2008.

[97] Stephen R. Schach. *Object-Oriented Software Engineering.* McGraw-Hill Science/Engineering, 2007.

[98] Andrea Schauerhuber, Manuel Wimmer, and Elisabeth Kapsammer. Bridging existing web modeling languages to model-driven engineering: a metamodel for WebML. In *Workshop proceedings of the sixth international conference on Web engineering*, ICWE '06, New York, NY, USA, 2006. ACM.

[99] Jason H. Sharp and Sherry D. Ryan. A theoretical framework of component-based software development phases. *SIGMIS Database*, 41:56–75, February 2010.

[100] Sylvain Sicard, Noel De Palma, and Daniel Hagimont. J2ee server scalability through ejb replication. In *Proceedings of the 2006 ACM symposium on Applied computing*, SAC '06, pages 778–785, New York, NY, USA, 2006. ACM.

[101] João L. Sobral and Miguel P. Monteiro. A domain-specific language for parallel and grid computing. In *Proceedings of the 2008 AOSD workshop on Domain-specific aspect languages*, DSAL '08, pages 2:1–2:4, New York, NY, USA, 2008. ACM.

[102] Paul Horaţiu Stan. A proposed technique for component based software development. In *Zilele Academice Clujene*, pages 52–56, 2010.

[103] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. *Studia UBB, Informatica*, LVI(3):9–14, 2011.

[104] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. In *Proc KEPT 2011 (eds: M. Frentiu, HF Pop, S. Motogna)*, pages 247–258, ISSN 2067-1180, Cluj University Press, 2011 (ISI Proc).

[105] Paul Horaţiu Stan. Case study: An order processing system developed using DevDSL. Submitted to: *16th East European Conference on Advances in Databases and Information Systems*, Poznan, September 18-21, 2012.

[106] Paul Horaţiu Stan. A custom validation mechanism for DevDSL. Submitted to: *Studia UBB, Informatica*, LVII(1):, 2012.

[107] Paul Horaţiu Stan. A proposed dsl for data intensive application development. *Studia UBB, Informatica*, LVII(1):accepted, 2012.

[108] Paul Horaţiu Stan and Camelia Şerban. A proposed approach for platform interoperability. *Studia UBB, Informatica*, LV(2):87–98, 2010.

[109] Aldo Bongio Stefano Ceri, Piero Fraternali. *Web Modeling Language (WebML): a modeling language for designing Web sites.* Politecnico di Milano, 2000.

[110] David Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework 2.0.* Addison-Wesley Professional, 2nd edition, 2009.

[111] Susan Stepney. *High Integrity Compilation : a case study.* Prentice-Hall, 1993.

[112] Yu Sun. Model transformation by demonstration. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 831–832, 2009.

[113] Clemens Szyperski. *Beyond Object-Oriented Programming.* ACM Press, 2002. New York.

[114] Mohamed Taleb, Ahmed Seffah, and Alain Abran. Model-driven architecture for web applications. In *Proceedings of the 12th international conference on Human-computer interaction: interaction design and usability*, HCI'07, pages 1198–1205, Berlin, Heidelberg, 2007. Springer-Verlag.

[115] Laurence Tratt Tony Clark. Language factories. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 949–955, 2009.

[116] Frank Truyen. *The Basics of Model Driven Architecture (MDA)*. Cephas Consulting Corp, 2006.

[117] Pravin V. Tulachan. *Developing EJB 2.0 Components*. Prentice Hall Professional Technical Reference, 2002.

[118] David Upton. *CodeIgniter for Rapid PHP Application Development: Improve your PHP coding productivity with the free compact open-source MVC CodeIgniter framework!* Packt Publishing, 2007.

[119] Matthias Veit and Stephan Herrmann. Model-view-controller and object teams: a perfect match of paradigms. In *Proceedings of the 2nd international conference on Aspect-oriented software development*, AOSD '03, pages 140–149, New York, NY, USA, 2003. ACM.

[120] Thomas Van de Velde, Christian Dupuis, Naveen Balani, and Sing Li. *Beginning Spring Framework 2*. John Wiley & Sons, Inc., New York, NY, USA, 2007.

[121] Eelco Visser. *WebDSL: A Case Study in Domain-Specific Language Engineering*. Delft University of Technology Software Engineering Research Group, 2008.

[122] HaiTao Wang and BaoXian Jia. Research based on web development of Spring integration framework. In *Proceedings of the 2010 International Forum on Information Technology and Applications - Volume 02*, IFITA '10, pages 325–328, Washington, DC, USA, 2010. IEEE Computer Society.

[123] Willian Massami Watanabe, David Fernandes Neto, Thiago Jabur Bittar, and Renata P. M. Fortes. Wcag conformance approach based on model-driven development and WebML. In *Proceedings of the 28th ACM International Conference on Design of Communication*, SIGDOC '10, pages 167–174, New York, NY, USA, 2010. ACM.

[124] WebDSL. A domain-specific language for developing dynamic web applications with a rich data model. http://WebDSL.org. Last accessed on December 12, 2010.

[125] Stephen A. White. Introduction to bpmn. http://www.zurich.ibm.com/~olz/teaching/ETH2011/White-BPMN-Intro.pdf. Last accessed on July 14, 2011.

[126] Wikipedia. Common object request broker architecture. http://en.wikipedia.org/wiki/CORBA. Last accessed on July 14, 2011.

[127] Wikipedia. Component-based software engineering. http://en.wikipedia.org/wiki/Component-based_software_engineering. Last accessed on May 03,2011.

[128] Wikipedia. Spring framework. http://en.wikipedia.org/wiki/Spring_Framework. Last accessed on July 14, 2011.

[129] Daniel T. Wojtaszek and John W. Chinneck. Faster mip solutions via new node selection rules. *Comput. Oper. Res.*, 37:1544–1556, September 2010.

[130] Hui Wu and Jeff Gray. Testing domain-specific languages in eclipse. In *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, OOPSLA '05, pages 173–174, New York, NY, USA, 2005. ACM.

[131] Hui Wu, Jeff Gray, and Marjan Mernik. Unit testing for domain-specific languages. In *Proceedings of the IFIP TC 2 Working Conference on Domain-Specific Languages*, DSL '09, pages 125–147, Berlin, Heidelberg, 2009. Springer-Verlag.

[132] E. Visser Z. Hemel, L.C.L Kats. *Code Generation by Model Transformation. A Case Study in Transformation Modularity.* Delft University of Technology Software Engineering Research Group, 2008.

[133] Guotao Zhuang and Junwei Du. MDA-based modeling and implementation of e-commerce web applications in WebML. In *Proceedings of the 2009 Second International Workshop on Computer Science and Engineering - Volume 02*, IWCSE '09, pages 507–510, Washington, DC, USA, 2009. IEEE Computer Society.