



## Chapter 7

# Conclusions

This thesis has investigated software evolution – a key and costly development activity in software engineering [Moad 1990] – in the context of model-driven engineering (MDE), a state-of-the art approach to software engineering. While MDE promises increased developer productivity [Watson 2008] and increased portability of software systems [Frankel 2002], it also poses several challenges that threaten its adoption. [Mens & Demeyer 2007], for example, suggest that identifying and managing evolutionary change in the context of MDE presents many open research challenges. The thesis research has contributed to the research challenges defined in [Mens & Demeyer 2007] and has explored the following research hypothesis:

*In existing MDE projects, the evolution of MDE development artefacts is typically managed in an ad-hoc manner with little regard for re-use. Dedicated structures and processes for managing evolutionary change can be designed by analysing evolution in existing MDE projects. Furthermore, supporting those dedicated structures and processes in contemporary MDE environments is beneficial in terms of increased productivity and understandability of software development.*

To explore the thesis hypothesis, the following research objectives were defined.

1. Identify and analyse the evolution of MDE development artefacts in existing projects.
2. Investigate the extent to which existing structures and processes can be used to manage the evolution of MDE development artefacts.
3. Propose and develop new structures and processes for managing the evolution of MDE development artefacts, and integrate those structures and processes with a contemporary MDE development environment.

4. Evaluate the proposed structures and processes for managing evolutionary change, particularly with respect to the productivity and understandability of software development.

The remainder of this section summarises the contributions of the thesis in relation to the thesis hypothesis and research objectives, and gives a brief description of and motivation for several potential extensions to the thesis research.

## 7.1 Research Contributions

The primary contributions of the thesis are summarised below, and the remainder of this section discusses each contribution in turn.

- Chapter 4 presented an investigation of evolution in existing MDE projects, which indicated ways in which models, metamodels and model management operations evolve, highlighted model-metamodel co-evolution as a commonly-occurring software evolution activity in MDE projects and led to a categorisation of existing approaches to managing model-metamodel co-evolution.
- Chapter 5 described the design and implementation of structures and processes for performing model-metamodel co-evolution, which included a metamodel-independent syntax for managing non-conformant models, a textual modelling notation for manually migrating models, and a model-to-model transformation language tailored for migration. The proposed structures and processes are interoperable with the Eclipse Modeling Framework [Steinberg *et al.* 2008], arguably the most widely-used contemporary MDE modelling framework.
- Chapter 6 detailed the evaluation of the proposed structures and process using quantitative measurements, expert evaluation and application to large, independent examples of co-evolution and explored the extent to which the proposed structures and processes increase developer productivity and the understandability of model migration.

### 7.1.1 Investigation of Evolution in MDE Projects

The way in which evolution occurs and is managed in existing MDE projects was analysed in Chapter 4. The analysis investigated two types of evolutionary change, model-metamodel co-evolution and model synchronisation, which were identified from the review presented in Chapter 3. For the MDE projects considered in Chapter 4, several suitable model-metamodel co-evolution examples – and no suitable model synchronisation examples – were located,

and consequently the remainder of the thesis focused on model-metamodel co-evolution.

The co-evolution examples were used to identify the differences between existing approaches to managing model-metamodel co-evolution, and to investigate the way in which model-metamodel co-evolution is managed in existing MDE projects. The investigation led to the definition of two distinct approaches to managing model-metamodel co-evolution in MDE projects, *user-driven* and *developer-driven* and to a categorisation of existing approaches, which was published in [Rose *et al.* 2009b] and has since been used and extended in several papers, including [Herrmannsdoerfer *et al.* 2010, Jurack & Mantz 2010, Méndez *et al.* 2010].

### 7.1.2 Structures and Processes for Managing Co-evolution

The analysis of existing MDE projects presented in Chapter 4 highlighted challenges for identifying and managing co-evolution. Managing co-evolution in a user-driven manner, for instance, is particularly challenging in contemporary MDE modelling environments because conformance is enforced implicitly and models and metamodels are kept separate. Similarly, the variation in programming and transformation languages typically used to specify model migration presents a challenge for comparing existing approaches to developer-driven co-evolution approaches. Moreover, none of the languages typically used have been tailored for the specific requirements of model migration. This thesis contribute structures and processes that seek to address the challenges summarised above.

#### Metamodel-Independent Syntax

Contemporary MDE modelling frameworks cannot be used to load non-conformant models. Consequently, model migration cannot be performed using the structures typically available in contemporary MDE modelling environments, such as model editors and model management operations. The metamodel-independent syntax, introduced in Chapter 5, is a proposed extension to contemporary MDE modelling frameworks that facilitates the loading of non-conformant models and provides services for reporting conformance problems.

The metamodel-independent syntax underpins the implementation of two further structures, the textual modelling notation described below, and the automatic conformance checking service introduced in [Rose *et al.* 2010c].

#### Textual Modelling Notation

When a small number of models are to be migrated, the effort required to specify an executable migration strategy might not be justifiable. Instead, models can be migrated by editing models by hand. The textual modelling notation, presented in Chapter 5, provides a notation for editing models in

contemporary MDE development environments. The notation proposed in this thesis adopts the Human-Usable Textual Notation (HUTN) [OMG 2004], a standard notation for textual modelling proposed by the Object Management Group (OMG) [OMG 2008c]. The implementation of HUTN introduced in Section 5.2, Epsilon HUTN, is the sole reference implementation of the HUTN standard, and was published in [Rose *et al.* 2008a].

During user-driven co-evolution, model editors cannot be used for migration and editing models in their underlying storage representation, which will not have been optimised for human use, can be error-prone and time consuming. The textual modelling notation introduced in Chapter 5 provides an alternative to editing models in their underlying storage representation.

### A Process for User-Driven Co-evolution

The analysis of existing MDE projects highlighted several projects in which model migration was performed using user-driven co-evolution techniques, and yet no existing work sought to address the specific requirements of user-driven co-evolution. Contemporary MDE modelling environments typically enforce conformance in an implicit manner, and cannot be used to load non-conformant models. Consequently, user-driven co-evolution is an iterative, error-prone and time-consuming task, because model editors and model management operations cannot be used to assist migration.

A typical process for performing user-driven co-evolution involves performing model migration by repeatedly switching between a model editor (which reports conformance problems) a text editor (in which conformance is reconciled by the user). Chapter 6 proposes an alternative process in which conformance reporting and reconciliation occur in the same environment, using the metamodel-independent syntax and textual modelling notation described above. The alternative process was published in [Rose *et al.* 2009a].

### Epsilon Flock: A Model Migration Language

In addition to the structures and processes for performing user-driven co-evolution, the thesis also contributes a structure dedicated to developer-driven co-evolution, a model migration language termed *Epsilon Flock*. The analysis performed in Chapter 4 showed that model migration is often specified with a model-to-model transformation language or with a general purpose programming language, and that these languages are not tailored to the specific requirements of model migration. The investigation presented in Chapter 5, highlighted that a language tailored for model migration would provide several benefits over repurposing an existing language to specify model migration. Flock was designed and implemented to explore the extent to which a language tailored for model migration might increase developer productivity and the understandability of model migration. The investigation of languages for

model migration and the design and implementation of Flock were published in [Rose *et al.* 2010f].

Flock contributes a novel mechanism for relating source and target model elements termed *conservative copy*, which is a hybrid of the two existing mechanisms used to relate source and target model elements in contemporary model-to-model transformation languages. Conservative copy automatically copies to the target model every source model element that conforms to the target metamodel, and does not copy to the target model source model elements that do not conform to the target metamodel.

### 7.1.3 Evaluation of Structures and Processes

The structures and processes introduced in this thesis have been evaluated using a variety of techniques, including a quantitative comparison, an expert evaluation and comparison to related processes and structures. Existing work on evolutionary change in MDE has typically been evaluated with a case study (such as in [Sprinkle 2003]) or by demonstration ([Cicchetti 2008]), and few papers in the area report the strengths and weaknesses proposed approaches, or seek to contextualise their contentions. The evaluation presented in Chapter 6 is a contribution in its own right as it presents several alternative evaluation techniques, including the first expert evaluation of model migration tools, and seeks to identify situations in which the proposed structures and processes are both effective and ineffective.

## 7.2 Future Work

In the context of a doctoral thesis, it is impossible to thoroughly investigate many of the issues raised in exploring the thesis hypothesis. Below, several potential extensions to the research presented in this thesis are identified, and any initial work in those areas is described.

### 7.2.1 Further Evaluation

The extent to which the structures and processes introduced in Chapter 5 increase developer productivity and the understandability of model migration has been explored via expert evaluation and comparison to related work. Assessing the way in which the proposed structures and processes affect productivity and understandability is challenging due to the subjective nature of the characteristics. Furthermore, in practice the proposed structures and processes would not be used in isolation. Evaluating the way in which software evolution is identified and managed in practice using comprehensive case and user studies would likely allow stronger claims to be made about the efficacy of the proposed structures and processes. Given the time constraints of a doctoral thesis, comprehensive case and user studies were not feasible, and

hence are desirable extensions to the thesis research. A key first step would likely be to establish a common vocabulary for discussing software evolution activities in the context of MDE, which would facilitate the comparison of user experiences.

### 7.2.2 Extensions to the Model Migration Language

The model migration language proposed in Chapter 5 and implemented in Epsilon Flock makes idiomatic commonly occurring patterns of model migration. The evaluation presented in Chapter 6 suggested that migration strategies are often more concise when specified with Flock rather than when specified with contemporary model-to-model transformation languages. The evaluation also highlighted a limitation in the implementation of Flock and demonstrated further model migration patterns that might be captured by model migration languages.

Addressing these issues would further improve the conciseness and re-usability of model migration strategies written in Flock and, hence, is an obvious area of future work. In particular, one language construct is used to control two concerns of a model migration strategy in the current implementation of Flock, and introducing separate language constructs for each concern would likely increase the potential for re-use between model migration rules. Applying Flock to the co-evolution examples used for evaluation highlighted further model migration patterns that might be made idiomatic in the language. For example, in situations where conservative copy can have side-effects, it may be desirable to afford more control of the copying algorithm via, for example, an `ignore` keyword that identifies values that should not be automatically copied.

### 7.2.3 Unifying Co-evolution Approaches

The thesis research has focused on one type of software evolution, model-metamodel co-evolution. Many further types of evolution occur in practice, including model refactoring and model synchronisation, which were discussed in Chapter 3. Changes to a metamodel affect not only models but also model management operations, such as model transformations. When changes are propagated from a metamodel to a model during migration, further artefacts might be impacted as an indirect consequence of the metamodel evolution.

The use of distinct structures and processes for each type of evolution poses usability challenges relating to the interoperability of tools and increased training effort. Seeking, instead, a unified approach to managing evolution might address these challenges, and presents an interesting opportunity for future work. Integrating the thesis research with approaches for managing other types of co-evolution, such as transformation-metamodel co-evolution, is one way in which the formulation of a unified approach might proceed. To this end,

an outline for integrating model-metamodel and transformation-metamodel co-evolution approaches has proposed in collaboration with Anne Etien, an Associate Professor at the Université Lille, and published in [Rose *et al.* 2010a].

### 7.2.4 Higher-Order Migration

In model transformation, a higher-order transformation consumes or produces a model transformation. Higher-order transformation has been used to generate model migration strategies [Cicchetti 2008, Garcés *et al.* 2009], and to compose and analyse transformations [Tisi *et al.* 2009]. Similarly, higher-order migration might be used effectively for migrating model transformation specifications between similar model transformation languages, and for migrating model management operations in response to changes to their specification language. For example, higher-order migration might be applied migrate model migration strategies between different types of transformation language, such as from a new-target to a conservative copy language.

### 7.2.5 Genericity

Chapter 6 identified a lack of metamodel-independent re-use as one of the primary weaknesses of Flock compared to related approaches. In Flock, model migration strategies are specified in terms of metamodel concepts, and consequently, the extent to which code can be re-used across migration strategies is reduced. By mixing model management languages with ideas from generic programming, [Lara & Guerra 2010] have identified one way in which model management operations can be specified in a manner that is independent of their metamodel. Applying the ideas presented in [Lara & Guerra 2010] to Epsilon and hence to Flock would facilitate increased re-use across model migration strategies, and address one of the primary weaknesses of Flock.

## 7.3 Coda

**To be completed** Some ideas:

- For software evolution research, need more data from industry.
- Closer collaboration between industry and academia to study evolution in context.
- More comprehensive assessment: for example, user studies.



# Bibliography

- [37-Signals 2008] 37-Signals. Ruby on Rails [online]. [Accessed 30 June 2008] Available at: <http://www.rubyonrails.org/>, 2008.
- [Ackoff 1962] Russell L. Ackoff. *Scientific Method: Optimizing Applied Research Decisions*. John Wiley and Sons, 1962.
- [Aizenbud-Reshef *et al.* 2005] N. Aizenbud-Reshef, R.F. Paige, J. Rubin, Y. Shaham-Gafni, and D.S. Kolovos. Operational semantics for traceability. In *Proc. ECMDA-FA Workshop on Traceability*, pages 8–14, 2005.
- [Alexander *et al.* 1977] Christopher Alexander, Sara Ishikawa, and Murray Silverstein. *A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure Series)*. Oxford University Press, 1977.
- [Álvarez *et al.* 2001] José Álvarez, Andy Evans, and Paul Sammut. MML and the metamodel architecture. In *Proc. Workshop on Transformation in UML*, 2001.
- [Apostel 1960] Leo Apostel. Towards the formal study of models in the non-formal sciences. *Synthese*, 12:125–161, 1960.
- [Arendt *et al.* 2009] Thorsten Arendt, Florian Mantz, Lars Schneider, and Gabriele Taentzer. Model refactoring in eclipse by LTK, EWL, and EMF Refactor: A case study. In *Proc. Joint MoDSE-MCCM Workshop*, 2009.
- [ATLAS 2007] ATLAS. Atlas Transformation Language Project Website [online]. [Accessed 30 June 2008] Available at: <http://www.eclipse.org/m2m/at1/>, 2007.
- [Backus 1978] John Backus. The history of FORTRAN I, II and III. *History of Programming Languages*, 1:165–180, 1978.
- [Balazinska *et al.* 2000] Magdalena Balazinska, Ettore Merlo, Michel Dagenais, Bruno Lagüe, and Kostas Kontogiannis. Advanced clone-analysis to support object-oriented system refactoring. In *Proc. Working Conference on Reverse Engineering*, pages 98–107. IEEE Computer Society, 2000.

- [Banerjee *et al.* 1987] Jay Banerjee, Won Kim, Hyoungh-Joo Kim, and Henry F. Korth. Semantics and implementation of schema evolution in object-oriented databases. In *Proc. Special Interest Group on Management of Data*, volume 16, pages 311–322. ACM, 1987.
- [Beck & Cunningham 1989] Kent Beck and Ward Cunningham. Constructing abstractions for object-oriented applications. *Journal of Object Oriented Programming*, 2, 1989.
- [Bézivin & Gerbé 2001] Jean Bézivin and Olivier Gerbé. Towards a precise definition of the OMG/MDA framework. In *Proc. ASE*, pages 273–280. IEEE Computer Society, 2001.
- [Bézivin 2005] Jean Bézivin. On the unification power of models. *Software and System Modeling*, 4(2):171–188, 2005.
- [Biermann *et al.* 2006] Enrico Biermann, Karsten Ehrig, Christian Köhler, Günter Kuhns, Gabriele Taentzer, and Eduard Weiss. Emf model refactoring based on graph transformation concepts. *ECEASST*, 3, 2006.
- [Bloch 2005] Joshua Bloch. How to design a good API and why it matters [online]. Keynote address to the LCSD Workshop at OOPSLA, October 2005, San Diego, United States of America. [Accessed 23 July 2009] Available at: <http://lcsd05.cs.tamu.edu/slides/keynote.pdf>, 2005.
- [Bohner 2002] Shawn A. Bohner. Software change impacts - an evolving perspective. In *Proc. International Conference on Software Maintenance (ICSM)*, pages 263–272. IEEE Computer Society, 2002.
- [Bosch 1998] Jan Bosch. Design patterns as language constructs. *Journal of Object Oriented Programming*, 11(2):18–32, 1998.
- [Briand *et al.* 2003] Lionel C. Briand, Yvan Labiche, and L. O’Sullivan. Impact analysis and change management of uml models. In *Proc. International Conference on Software Maintenance (ICSM)*, pages 256–265. IEEE Computer Society, 2003.
- [Brooks 1986] Frederick P. Brooks. No silver bullet – essence and accident in software engineering (invited paper). In *Proc. International Federation for Information Processing*, pages 1069–1076, 1986.
- [Brown *et al.* 1998] William J. Brown, Raphael C. Malveau, Hays W. McCormick III, and Thomas J. Mowbray. *Anti Patterns*. Wiley, 1998.
- [Cervelle *et al.* 2006] Julien Cervelle, Rémi Forax, and Gilles Roussel. Tatoo: an innovative parser generator. In *Principles and Practice of Programming in Java*, pages 13–20. ACM, 2006.

- [Ceteva 2008] Ceteva. XMF – the extensible programming language [online]. [Accessed 30 June 2008] Available at: <http://www.ceteva.com/xmf.html>, 2008.
- [Chen & Chou 1999] J.Y.J. Chen and S.C. Chou. Consistency management in a process environment. *Systems and Software*, 47(2-3):105–110, 1999.
- [Cicchetti *et al.* 2008] Antonio Cicchetti, Davide Di Ruscio, Romina Eramo, and Alfonso Pierantonio. Automating co-evolution in model-driven engineering. In *Proc. EDOC*, pages 222–231. IEEE Computer Society, 2008.
- [Cicchetti 2008] Antonio Cicchetti. *Difference Representation and Conflict Management in Model-Driven Engineering*. PhD thesis, Università degli Studi dell’Aquila, L’Aquila, Italy, 2008.
- [Clark *et al.* 2008] Tony Clark, Paul Sammut, and James Williams. Superlanguages: Developing languages and applications with XMF [online]. [Accessed 30 June 2008] Available at: <http://www.ceteva.com/docs/Superlanguages.pdf>, 2008.
- [Cleland-Huang *et al.* 2003] Jane Cleland-Huang, Carl K. Chang, and Mark Christensen. Event-based traceability for managing evolutionary change. *IEEE Transactions on Software Engineering*, 29(9):796–810, 2003.
- [Costa & Silva 2007] M. Costa and A.R. da Silva. RT-MDD framework – a practical approach. In *Proc. ECMDA-FA Workshop on Traceability*, pages 17–26, 2007.
- [Czarnecki & Helsen 2006] Krzysztof Czarnecki and Simon Helsen. Feature-based survey of model transformation approaches. *IBM Systems Journal*, 45(3):621–646, 2006.
- [Deursen *et al.* 2000] Arie van Deursen, Paul Klint, and Joost Visser. Domain-specific languages: An annotated bibliography. *SIGPLAN Notices*, 35(6):26–36, 2000.
- [Deursen *et al.* 2007] Arie van Deursen, Eelco Visser, and Jos Warmer. Model-driven software evolution: A research agenda. In *Proc. Workshop on Model-Driven Software Evolution*, pages 41–49, 2007.
- [Dig & Johnson 2006a] Danny Dig and Ralph Johnson. Automated upgrading of component-based applications. In *OOPSLA Companion*, pages 675–676, 2006.
- [Dig & Johnson 2006b] Danny Dig and Ralph Johnson. How do APIs evolve? A story of refactoring. *Journal of Software Maintenance and Evolution*, 18(2):83–107, 2006.

- [Dig *et al.* 2006] Danny Dig, Can Comertoglu, Darko Marinov, and Ralph Johnson. Automated detection of refactorings in evolving components. In *Proc. European Conference on Object-Oriented Programming*, volume 4067 of *LNCS*, pages 404–428. Springer, 2006.
- [Dig *et al.* 2007] Danny Dig, Kashif Manzoor, Ralph Johnson, and Tien N. Nguyen. Refactoring-aware configuration management for object-oriented programs. In *Proc. International Conference on Software Engineering*, pages 427–436. IEEE Computer Society, 2007.
- [Dig 2007] Daniel Dig. *Automated Upgrading of Component-Based Applications*. PhD thesis, University of Illinois at Urbana-Champaign, USA, 2007.
- [Dmitriev 2004] Sergey Dmitriev. Language oriented programming: The next programming paradigm. *JetBrains onBoard [online]*, 1, 2004. [Accessed 30 June 2008] Available at: <http://www.onboard.jetbrains.com/is1/articles/04/10/lop/>.
- [Drivalos *et al.* 2008] Nicholas Drivalos, Richard F. Paige, Kiran J. Fernandes, and Dimitrios S. Kolovos. Towards rigorously defined model-to-model traceability. In *Proc. European Conference on the Model Driven Architecture Workshop on Traceability*, 2008.
- [Ducasse *et al.* 1999] Stéphane Ducasse, Matthias Rieger, and Serge Demeyer. A language independent approach for detecting duplicated code. In *Proc. International Conference on Software Maintenance*, pages 109–118. IEEE Computer Society, 1999.
- [Eclipse 2008a] Eclipse. Eclipse Modeling Framework project [online]. [Accessed 22 January 2009] Available at: <http://www.eclipse.org/modeling/emf/>, 2008.
- [Eclipse 2008b] Eclipse. Eclipse project [online]. [Accessed 20 January 2009] Available at: <http://www.eclipse.org/>, 2008.
- [Eclipse 2008c] Eclipse. Epsilon home page [online]. [Accessed 30 June 2008] Available at: <http://www.eclipse.org/gmt/epsilon/>, 2008.
- [Eclipse 2008d] Eclipse. Generative Modelling Technologies project [online]. [Accessed 30 June 2008] Available at: <http://www.eclipse.org/gmt/>, 2008.
- [Eclipse 2009a] Eclipse. Model Development Tools project [online]. [Accessed 6 January 2008] Available at: <http://www.eclipse.org/modeling/mdt/>, 2009.

- [Eclipse 2009b] Eclipse. UML2 Model Development Tools project [online]. [Accessed 7 September 2009] Available at: <http://www.eclipse.org/modeling/mdt/uml2>, 2009.
- [Eclipse 2010] Eclipse. Connected Data Objects Model Repository Project Website [online]. [Accessed 15 February 2010] Available at: <http://www.eclipse.org/modeling/emf/?project=cdo#cdo>, 2010.
- [Edelweiss & Freitas Moreira 2005] Nina Edelweiss and Álvaro Freitas Moreira. Temporal and versioning model for schema evolution in object-oriented databases. *Data & Knowledge Engineering*, 53(2):99–128, 2005.
- [Elmasri & Navathe 2006] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*. Addison-Wesley Longman, 2006.
- [Erlikh 2000] Len Erlikh. Leveraging legacy system dollars for e-business. *IT Professional*, 2(3):17–23, 2000.
- [Evans 2004] E. Evans. *Domain-Driven Design: Tackling Complexity In the Heart of Software*. Addison-Wesley, Boston, MA, USA, 2004.
- [Feathers 2004] Michael C. Feathers. *Working Effectively with Legacy Code*. Prentice Hall, 2004.
- [Ferrandina *et al.* 1995] Fabrizio Ferrandina, Thorsten Meyer, Roberto Zicari, Guy Ferran, and Joëlle Madec. Schema and database evolution in the O2 object database system. In *Very Large Data Bases*, pages 170–181. Morgan Kaufmann, 1995.
- [Fowler 1999] Martin Fowler. *Refactoring: improving the design of existing code*. Addison-Wesley, 1999.
- [Fowler 2002] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.
- [Fowler 2005] Martin Fowler. Language workbenches: The killer-app for domain specific languages? [online]. [Accessed 30 June 2008] Available at: <http://www.martinfowler.com/articles/languageWorkbench.html>, 2005.
- [Fowler 2010] Martin Fowler. *Domain Specific Languages*. Addison-Wesley Professional, 2010.
- [Frankel 2002] David Frankel. *Model Driven Architecture: Applying MDA to Enterprise Computing*. Wiley, 2002.

- [Frenzel 2006] Leif Frenzel. The language toolkit: An API for automated refactorings in eclipse-based IDEs [online]. [Accessed 02 August 2010] Available at: <http://www.eclipse.org/articles/Article-LTK/ltk.html>, 2006.
- [Fritzsche *et al.* 2008] M. Fritzsche, J. Johannes, S. Zschaler, A. Zharebtsov, and A. Terekhov. Application of tracing techniques in Model-Driven Performance Engineering. In *Proc. ECMDA Traceability Workshop (ECMDA-TW)*, pages 111–120, 2008.
- [Fuhrer *et al.* 2007] Robert M. Fuhrer, Adam Kiezun, and Markus Keller. Refactoring in the Eclipse JDT: Past, present, and future. In *Proc. Workshop on Refactoring Tools*, 2007.
- [Gamma *et al.* 1995] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley, 1995.
- [Garcés *et al.* 2009] Kelly Garcés, Frédéric Jouault, Pierre Cointe, and Jean Bézivin. Managing model adaptation by precise detection of metamodel changes. In *Proc. ECMDA-FA*, volume 5562 of *LNCS*, pages 34–49. Springer, 2009.
- [Gosling *et al.* 2005] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java<sup>TM</sup> Language Specification*. Addison-Wesley, Boston, MA, USA, 2005.
- [Graham 1993] Paul Graham. *On Lisp: Advanced Techniques for Common Lisp*. Prentice-Hall, 1993.
- [Greenfield *et al.* 2004] Jack Greenfield, Keith Short, Steve Cook, and Stuart Kent. *Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools*. Wiley, 2004.
- [Gronback 2009] R.C. Gronback. *Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit*. Addison-Wesley Professional, 2009.
- [Gruschko *et al.* 2007] Boris Gruschko, Dimitrios S. Kolovos, and Richard F. Paige. Towards synchronizing models with evolving metamodels. In *Proc. Workshop on Model-Driven Software Evolution*, 2007.
- [Guerrini *et al.* 2005] Giovanna Guerrini, Marco Mesiti, and Daniele Rossi. Impact of XML schema evolution on valid documents. In *Proc. Workshop on Web Information and Data Management*, pages 39–44, 2005.
- [Halstead 1977] Maurice H. Halstead. *Elements of Software Science*. Elsevier Science Inc., 1977.

- [Hearnden *et al.* 2006] David Hearnden, Michael Lawley, and Kerry Raymond. Incremental model transformation for the evolution of model-driven systems. In *Model Driven Engineering Languages and Systems*, volume 4199 of *LNCS*, pages 321–335. Springer, 2006.
- [Heidenreich *et al.* 2009] Florian Heidenreich, Jendrik Johannes, Sven Karol, Mirko Seifert, and Christian Wende. Derivation and refinement of textual syntax for models. In *Proc. ECMDA-FA*, volume 5562 of *LNCS*, pages 114–129. Springer, 2009.
- [Herrmannsdoerfer *et al.* 2008] Markus Herrmannsdoerfer, Sebastian Benz, and Elmar Juergens. Automatability of coupled evolution of metamodels and models in practice. In *Proc. International Conference on Model Driven Engineering Languages and Systems*, volume 5301 of *LNCS*, pages 645–659. Springer, 2008.
- [Herrmannsdoerfer *et al.* 2009a] M. Herrmannsdoerfer, D. Ratiu, and G. Wachsmuth. Language evolution in practice. In *Proc. SLE*, volume 5696 of *LNCS*, pages 3–22. Springer, 2009.
- [Herrmannsdoerfer *et al.* 2009b] Markus Herrmannsdoerfer, Sebastian Benz, and Elmar Juergens. COPE - automating coupled evolution of metamodels and models. In *Proc. ECOOP*, volume 5653 of *LNCS*, pages 52–76. Springer, 2009.
- [Herrmannsdoerfer *et al.* 2010] Markus Herrmannsdoerfer, Sander Vermolen, and Guido Wachsmuth. An extensive catalog of operators for the coupled evolution of metamodels and models. In *Proc. SLE*, volume TBC of *LNCS*, page TBC. Springer, 2010.
- [Hussey & Paternostro 2006] Kenn Hussey and Marcelo Paternostro. Advanced features of EMF. Tutorial at EclipseCon 2006, California, USA. [Accessed 07 September 2009] Available at: <http://www.eclipsecon.org/2006/Sub.do?id=171>, 2006.
- [IBM 2005] IBM. Emfatic Language for EMF Development [online]. [Accessed 30 June 2008] Available at: <http://www.alphaworks.ibm.com/tech/emfatic>, 2005.
- [INRIA 2007] INRIA. AMMA project page [online]. [Accessed 30 June 2008] Available at: <http://wiki.eclipse.org/AMMA>, 2007.
- [IRISA 2007] IRISA. Sintaks. <http://www.kermeta.org/sintaks/>, 2007.
- [ISO/IEC 1996] Information Technology ISO/IEC. Syntactic metalanguage – Extended BNF. ISO 14977:1996 International Standard, 1996.

- [ISO/IEC 2002] Information Technology ISO/IEC. Z Formal Specification Notation – Syntax, Type System and Semantics. ISO 13568:2002 International Standard, 2002.
- [Jackson 1995] M. Jackson. *Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices*. ACM Press, 1995.
- [JetBrains 2008] JetBrains. MPS – Meta Programming System [online]. [Accessed 30 June 2008] Available at: <http://www.jetbrains.com/mps/index.html>, 2008.
- [Jouault & Kurtev 2005] Frédéric Jouault and Ivan Kurtev. Transforming models with ATL. In *Proc. Satellite Events at the International Conference on Model Driven Engineering Languages and Systems*, volume 3844 of *LNCS*, pages 128–138. Springer, 2005.
- [Jouault 2005] Frédéric Jouault. Loosely coupled traceability for ATL. In *Proc. ECMDA-FA Workshop on Traceability*, 2005.
- [Jurack & Mantz 2010] Stefan Jurack and Florian Mantz. Towards meta-model evolution of EMF models with Henshin. In *Proc. ME Workshop*, 2010.
- [Kataoka *et al.* 2001] Yoshio Kataoka, Michael D. Ernst, William G. Griswold, and David Notkin. Automated support for program refactoring using invariants. In *Proc. International Conference on Software Maintenance*, pages 736–743. IEEE Computer Society, 2001.
- [Kelly & Tolvanen 2008] Steven Kelly and Juha-Pekka Tolvanen. *Domain-Specific Modelling*. Wiley, 2008.
- [Kerievsky 2004] Joshua Kerievsky. *Refactoring to Patterns*. Addison-Wesley, 2004.
- [Kleppe *et al.* 2003] Anneke G. Kleppe, Jos Warmer, and Wim Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley, 2003.
- [Klint *et al.* 2003] P. Klint, R. Lämmel, and C. Verhoef. Towards an engineering discipline for grammarware. *ACM Transactions on Software Engineering Methodology*, 14:331–380, 2003.
- [Kolovos *et al.* 2006a] Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Merging models with the epsilon merging language (eml). In *Proc. MoDELS*, volume 4199 of *Lecture Notes in Computer Science*, pages 215–229. Springer, 2006.



- [Kolovos *et al.* 2006b] Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Model comparison: a foundation for model composition and model transformation testing. In *Proc. Workshop on Global Integrated Model Management*, pages 13–20, 2006.
- [Kolovos *et al.* 2006c] Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. The Epsilon Object Language (EOL). In *Proc. ECMDA-FA*, volume 4066 of *LNCS*, pages 128–142. Springer, 2006.
- [Kolovos *et al.* 2007a] Dimitrios S. Kolovos, Richard F. Paige, Fiona A.C. Polack, and Louis M. Rose. Update transformations in the small with the Epsilon Wizard Language. *Journal of Object Technology*, 6(9):53–69, 2007.
- [Kolovos *et al.* 2007b] Dimitrios S. Kolovos, Richard F. Paige, Louis M. Rose, and Fiona A.C. Polack. Bridging the Epsilon Wizard Language and the Eclipse Graphical Modeling Framework. In *Proc. Eclipse Summit*, Ludwigsburg, Germany, 2007.
- [Kolovos *et al.* 2008a] Dimitrios S. Kolovos, Richard F. Paige, and Fiona Polack. The Epsilon Transformation Language. In *Proc. ICMT*, volume 5063 of *LNCS*, pages 46–60. Springer, 2008.
- [Kolovos *et al.* 2008b] Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. On the evolution of OCL for capturing structural constraints in modelling languages. In *Proc. Workshop on Rigorous Methods for Software Construction and Analysis*, 2008.
- [Kolovos *et al.* 2008c] Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Scalability : The holy grail of model driven engineering. In *Proc. Workshop on Challenges in Model Driven Engineering*, 2008.
- [Kolovos *et al.* 2009] Dimitrios S. Kolovos, Richard F. Paige, and Louis M. Rose. EuGENia: GMF for mortals. Long talk at Eclipse Summit Europe, October 2009, Ludwigsburg, Germany. Available at: [https://www.eclipsecon.org/submissions/ese2009/view\\_talk.php?id=979](https://www.eclipsecon.org/submissions/ese2009/view_talk.php?id=979) [Accessed 12 April 2010], 2009.
- [Kolovos 2009] Dimitrios S. Kolovos. *An Extensible Platform for Specification of Integrated Languages for Model Management*. PhD thesis, University of York, United Kingdom, 2009.
- [Kramer 2001] Diane Kramer. XEM: XML Evolution Management. Master’s thesis, Worcester Polytechnic Institute, MA, USA, 2001.
- [Kurtev 2004] Ivan Kurtev. *Adaptability of Model Transformations*. PhD thesis, University of Twente, Netherlands, 2004.

- [Lago *et al.* 2009] Patricia Lago, Henry Muccini, and Hans van Vliet. A scoped approach to traceability management. *Systems and Software*, 82(1):168–182, 2009.
- [Lämmel & Verhoef 2001] R. Lämmel and C. Verhoef. Semi-automatic grammar recovery. *Software - Practice and Experience*, 31(15):1395–1438, 2001.
- [Lämmel 2001] R. Lämmel. Grammar adaptation. In *Proc. Formal Methods for Increasing Software Productivity (FME), International Symposium of Formal Methods Europe*, volume 2021 of *LNCIS*, pages 550–570. Springer, 2001.
- [Lämmel 2002] R. Lämmel. Towards generic refactoring. In *Proc. ACM SIGPLAN Workshop on Rule-Based Programming*, pages 15–28. ACM, 2002.
- [Lara & Guerra 2010] Juan de Lara and Esther Guerra. Generic meta-modelling with concepts, templates and mixin layers. In *Proc. MoDELS (1)*, volume 6394 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 2010.
- [Lehman 1969] Meir M. Lehman. The programming process. Technical report, IBM Res. Rep. RC 2722, 1969.
- [Lerner 2000] Barbara Staudt Lerner. A model for compound type changes encountered in schema evolution. *ACM Transactions on Database Systems*, 25(1):83–127, 2000.
- [Mäder *et al.* 2008] P. Mäder, O. Gotel, and I. Philippow. Rule-based maintenance of post-requirements traceability relations. In *Proc. IEEE International Requirements Engineering Conference (RE)*, pages 23–32, 2008.
- [Martin & Martin 2006] R.C. Martin and M. Martin. *Agile Principles, Patterns, and Practices in C#*. Prentice Hall, Upper Saddle River, NJ, USA, 2006.
- [McCarthy 1978] John McCarthy. History of Lisp. *History of Programming Languages*, 1:217–223, 1978.
- [McNeile 2003] Ashley McNeile. MDA: The vision with the hole? [Accessed 30 June 2008] Available at: <http://www.metamaxim.com/download/documents/MDAv1.pdf>, 2003.
- [Mellor & Balcer 2002] Stephen J. Mellor and Marc Balcer. *Executable UML: A Foundation for Model-Driven Architectures*. Addison-Wesley Longman, 2002.

- [Melnik 2004] Sergey Melnik. *Generic Model Management: Concepts and Algorithms*. PhD thesis, University of Leipzig, Germany, 2004.
- [Méndez *et al.* 2010] David Méndez, Anne Etien, Alexis Muller, and Rubby Casallas. Towards transformation migration after metamodel evolution. In *Proc. ME Workshop*, 2010.
- [Mens & Demeyer 2007] Tom Mens and Serge Demeyer. *Software Evolution*. Springer-Verlag, 2007.
- [Mens & Tourwé 2004] Tom Mens and Tom Tourwé. A survey of software refactoring. *IEEE Transactions on Software Engineering*, 30(2):126–139, 2004.
- [Mens *et al.* 2007] Tom Mens, Gabriele Taentzer, and Dirk Müller. Challenges in model refactoring. In *Proc. Workshop on Object-Oriented Reengineering*, 2007.
- [Merriam-Webster 2010] Merriam-Webster. Definition of Nuclear Family. <http://www.merriam-webster.com/dictionary/nuclear%20family>, 2010.
- [Moad 1990] J Moad. Maintaining the competitive edge. *Datamation*, 36(4):61–66, 1990.
- [Moha *et al.* 2009] Naouel Moha, Vincent Mahé, Olivier Barais, and Jean-Marc Jézéquel. Generic model refactorings. In *Proc. MoDELS*, volume 5795 of *LNCS*, pages 628–643. Springer, 2009.
- [Muller & Hassenforder 2005] Pierre-Alain Muller and Michel Hassenforder. HUTN as a Bridge between ModelWare and GrammarWare. In *Proc. Workshop in Software Modelling Engineering*, 2005.
- [Nentwich *et al.* 2003] C. Nentwich, W. Emmerich, A. Finkelstein, and E. Ellmer. Flexible consistency checking. *ACM Transactions on Software Engineering and Methodology*, 12(1):28–63, 2003.
- [Nguyen *et al.* 2005] Tien Nhut Nguyen, Cheng Thao, and Ethan V. Munson. On product versioning for hypertexts. In *Proc. International Workshop on Software Configuration Management (SCM)*, pages 113–132. ACM, 2005.
- [Northrop 2006] L. Northrop. Ultra-large scale systems: The software challenge of the future. Technical report, Carnegie Mellon, June 2006.
- [Oldevik *et al.* 2005] Jon Oldevik, Tor Neple, Roy Grønmo, Jan Øyvind Aagedal, and Arne-Jørgen Berre. Toward standardised model to text transformations. In *Proc. ECMDA-FA*, volume 3748 of *LNCS*, pages 239–253. Springer, 2005.

- [Olsen & Oldevik 2007] Gøran K. Olsen and Jon Oldevik. Scenarios of traceability in model to text transformations. In *Proc. ECMDA-FA*, volume 4530 of *Lecture Notes in Computer Science*, pages 144–156. Springer, 2007.
- [OMG 2001] OMG. Unified Modelling Language 1.4 Specification [online]. [Accessed 15 September 2008] Available at: <http://www.omg.org/spec/UML/1.4/>, 2001.
- [OMG 2004] OMG. Human-Usable Textual Notation 1.0 Specification [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/technology/documents/formal/hutn.htm>, 2004.
- [OMG 2005] OMG. MOF QVT Final Adopted Specification [online]. [Accessed 22 July 2009] Available at: [www.omg.org/docs/ptc/05-11-01.pdf](http://www.omg.org/docs/ptc/05-11-01.pdf), 2005.
- [OMG 2006] OMG. Object Constraint Language 2.0 Specification [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/technology/documents/formal/ocl.htm>, 2006.
- [OMG 2007a] OMG. Unified Modelling Language 2.1.2 Specification [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/spec/UML/2.1.2/>, 2007.
- [OMG 2007b] OMG. Unified Modelling Language 2.2 Specification [online]. [Accessed 5 March 2010] Available at: <http://www.omg.org/spec/UML/2.2/>, 2007.
- [OMG 2007c] OMG. XML Metadata Interchange 2.1.1 Specification [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/technology/documents/formal/xmi.htm>, 2007.
- [OMG 2008a] OMG. Meta-Object Facility [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/mof>, 2008.
- [OMG 2008b] OMG. Model Driven Architecture [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org/mda/>, 2008.
- [OMG 2008c] OMG. Object Management Group home page [online]. [Accessed 30 June 2008] Available at: <http://www.omg.org>, 2008.
- [Opdyke 1992] William F. Opdyke. *Refactoring Object-Oriented Frameworks*. PhD thesis, University of Illinois at Urbana-Champaign, IL, USA, 1992.
- [openArchitectureWare 2007] openArchitectureWare. openArchitectureWare Project Website [online]. [Accessed 30 June 2008] Available at: <http://www.eclipse.org/gmt/oaw/>, 2007.

- [openArchitectureWare 2008] openArchitectureWare. XPand Language Reference [online]. [Accessed 18 August 2010] Available at: <http://wiki.eclipse.org/AMMA>, 2008.
- [Paige *et al.* 2007] Richard F. Paige, Phillip J. Brooke, and Jonathan S. Ostroff. Metamodel-based model conformance and multiview consistency checking. *ACM Transactions on Software Engineering and Methodology*, 16(3), 2007.
- [Paige *et al.* 2009] Richard F. Paige, Louis M. Rose, Xiaocheng Ge, Dimitrios S. Kolovos, and Phillip J. Brooke. FPTC: Automated safety analysis for domain-specific languages. In *MoDELS Workshops and Symposia*, volume 5421 of *Lecture Notes in Computer Science*, pages 229–242. Springer, 2009.
- [Parr 2007] Terence Parr. *The Definitive ANTLR Reference: Building Domain-Specific Languages*. Pragmatic Programmers, 2007.
- [Patrascoiu & Rodgers 2004] Octavian Patrascoiu and Peter Rodgers. Embedding OCL expressions in YATL. In *Proc. OCL and Model-Driven Engineering Workshop*, 2004.
- [Pilgrim *et al.* 2008] Jens von Pilgrim, Bert Vanhooft, Immo Schulz-Gerlach, and Yolande Berbers. Constructing and visualizing transformation chains. In *Proc. European Conference on the Model Driven Architecture – Foundations and Applications*, volume 5095 of *LNCS*, pages 17–32. Springer, 2008.
- [Pizka & Jürgens 2007] M. Pizka and E. Jürgens. Automating language evolution. In *Proc. Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering (TASE)*, pages 305–315. IEEE Computer Society, 2007.
- [Porres 2003] Ivan Porres. Model refactorings as rule-based update transformations. In *Proc. UML*, volume 2863 of *LNCS*, pages 159–174. Springer, 2003.
- [RAE & BCS 2004] The RAE and The BCS. The challenges of complex IT projects. Technical report, The Royal Academy of Engineering, April 2004.
- [Ramil & Lehman 2000] Juan F. Ramil and Meir M. Lehman. Cost estimation and evolvability monitoring for software evolution processes. In *Proc. Workshop on Empirical Studies of Software Maintenance*, 2000.

- [Ráth *et al.* 2008] István Ráth, Gábor Bergmann, András Ökrös, and Dániel Varró. Live model transformations driven by incremental pattern matching. In *Proc. ICMT*, volume 5063 of *LNCS*, pages 107–121. Springer, 2008.
- [Rising 2001] Linda Rising, editor. *Design patterns in communications software*. Cambridge University Press, 2001.
- [Rose *et al.* 2008a] Louis M. Rose, Richard F. Paige, Dimitrios S. Kolovos, and Fiona A.C. Polack. Constructing models with the Human-Usable Textual Notation. In *Proc. International Conference on Model Driven Engineering Languages and Systems*, volume 5301 of *LNCS*, pages 249–263. Springer, 2008.
- [Rose *et al.* 2008b] Louis M. Rose, Richard F. Paige, Dimitrios S. Kolovos, and Fiona A.C. Polack. The Epsilon Generation Language. In *Proc. European Conference on Model Driven Architecture – Foundations and Applications*, volume 5095 of *LNCS*, pages 1–16. Springer, 2008.
- [Rose *et al.* 2009a] Louis M. Rose, Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Enhanced automation for managing model and metamodel inconsistency. In *Proc. ASE*, pages 545–549. ACM Press, 2009.
- [Rose *et al.* 2009b] Louis M. Rose, Richard F. Paige, Dimitrios S. Kolovos, and Fiona A.C. Polack. An analysis of approaches to model migration. In *Proc. Joint MoDSE-MCCM Workshop*, 2009.
- [Rose *et al.* 2010a] Louis M. Rose, Anne Etien, David Méndez, Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Comparing model-metamodel and transformation-metamodel co-evolution. In *Proc. ME Workshop*, 2010.
- [Rose *et al.* 2010b] Louis M. Rose, Markus Herrmannsdoerfer, James R. Williams, Dimitrios S. Kolovos, Kelly Garcés, Richard F. Paige, and Fiona A.C. Polack. A comparison of model migration tools. In *Proc. MoDELS*, volume TBC of *Lecture Notes in Computer Science*, page TBC. Springer, 2010.
- [Rose *et al.* 2010c] Louis M. Rose, Dimitrios S. Kolovos, Nicholas Drivalos, James R. Williams, Richard F. Paige, Fiona A.C. Polack, and Kiran J. Fernandes. Concordance: An efficient framework for managing model integrity [submitted to]. In *Proc. European Conference on Modelling Foundations and Applications*, volume 6138 of *LNCS*, pages 62–73. Springer, 2010.

- [Rose *et al.* 2010d] Louis M. Rose, Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Migrating activity diagrams with Epsilon Flock. In *Proc. TTC*, 2010.
- [Rose *et al.* 2010e] Louis M. Rose, Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Model migration case. In *Proc. TTC*, 2010.
- [Rose *et al.* 2010f] Louis M. Rose, Dimitrios S. Kolovos, Richard F. Paige, and Fiona A.C. Polack. Model migration with Epsilon Flock. In *Proc. ICMT*, volume 6142 of *Lecture Notes in Computer Science*, pages 184–198. Springer, 2010.
- [Selic 2003] Bran Selic. The pragmatics of Model-Driven Development. *IEEE Software*, 20(5):19–25, 2003.
- [Selic 2005] Bran Selic. Whats new in UML 2.0? *IBM Rational software*, 2005.
- [Sendall & Kozaczynski 2003] Shane Sendall and Wojtek Kozaczynski. Model transformation: The heart and soul of model-driven software development. *IEEE Software*, 20:42–45, 2003.
- [Sjøberg 1993] Dag I.K. Sjøberg. Quantifying schema evolution. *Information & Software Technology*, 35(1):35–44, 1993.
- [Sommerville 2006] Ian Sommerville. *Software Engineering*. Addison-Wesley Longman, 2006.
- [Sprinkle & Karsai 2004] Jonathan Sprinkle and Gábor Karsai. A domain-specific visual language for domain model evolution. *Journal of Visual Languages and Computing*, 15(3-4):291–307, 2004.
- [Sprinkle 2003] Jonathan Sprinkle. *Metamodel Driven Model Migration*. PhD thesis, Vanderbilt University, TN, USA, 2003.
- [Stahl *et al.* 2006] Thomas Stahl, Markus Voelter, and Krzysztof Czarnecki. *Model-Driven Software Development: Technology, Engineering, Management*. Wiley, 2006.
- [Starfield *et al.* 1990] M. Starfield, K.A. Smith, and A.L. Bleloch. *How to model it: Problem Solving for the Computer Age*. McGraw-Hill, 1990.
- [Steinberg *et al.* 2008] Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework*. Addison-Wesley Professional, 2008.

- [Su *et al.* 2001] Hong Su, Diane Kramer, Li Chen, Kajal T. Claypool, and Elke A. Rundensteiner. XEM: Managing the evolution of XML documents. In *Proc. Workshop on Research Issues in Data Engineering*, pages 103–110, 2001.
- [Tisi *et al.* 2009] Massimo Tisi, Frédéric Jouault, Piero Fraternali, Stefano Ceri, and Jean Bézivin. On the use of higher-order model transformations. In *Proc. ECMDA-FA*, volume 5562 of *Lecture Notes in Computer Science*, pages 18–33. Springer, 2009.
- [Tratt 2008] Laurence Tratt. A change propagating model transformation language. *Journal of Object Technology*, 7(3):107–124, 2008.
- [Varró & Balogh 2007] Dániel Varró and András Balogh. The model transformation language of the VIATRA2 framework. *Science of Computer Programming*, 68(3):187–207, 2007.
- [Vries & Roddick 2004] Denise de Vries and John F. Roddick. Facilitating database attribute domain evolution using meso-data. In *Proc. Workshop on Evolution and Change in Data Management*, pages 429–440, 2004.
- [W3C 2007a] W3C. W3C XML Schema 1.1 Specification [online]. [Accessed 30 June 2008] Available at: <http://www.w3.org/XML/Schema>, 2007.
- [W3C 2007b] W3C. World Wide Web Consortium [online]. [Accessed 30 June 2008] Available at: <http://www.w3.org/>, 2007.
- [Wachsmuth 2007] Guido Wachsmuth. Metamodel adaptation and model co-adaptation. In *Proc. ECOOP*, volume 4609 of *LNCS*, pages 600–624. Springer, 2007.
- [Wallace 2005] Malcolm Wallace. Modular architectural representation and analysis of fault propagation and transformation. *Electronic Notes in Theoretical Computer Science*, 141(3):53–71, 2005.
- [Ward 1994] Martin P. Ward. Language-oriented programming. *Software — Concepts and Tools*, 15(4):147–161, 1994.
- [Watson 2008] Andrew Watson. A brief history of MDA. *Upgrade*, 9(2), 2008.
- [Welch & Barnes 2005] Peter H. Welch and Fred R. M. Barnes. Communicating mobile processes. In *Proc. Symposium on the Occasion of 25 Years of Communicating Sequential Processes (CSP)*, volume 3525 of *LNCS*, pages 175–210. Springer, 2005.
- [Winkler & Pilgrim 2009] Stefan Winkler and Jens von Pilgrim. A survey of traceability in requirements engineering and model-driven development. *Software and Systems Modeling*, December 2009.