

Alan Hartman
David Kreische (Eds.)

LNCS 3748

Model Driven Architecture – Foundations and Applications

First European Conference, ECMDA-FA 2005
Nuremberg, Germany, November 2005
Proceedings



Springer

TP31-53
E19
2005

Alan Hartman David Kreische (Eds.)

Model Driven Architecture – Foundations and Applications

First European Conference, ECMDA-FA 2005
Nuremberg, Germany, November 7-10, 2005
Proceedings



E200600961



Springer

Volume Editors

Alan Hartman

IBM Haifa Research Laboratory

Model Driven Engineering Technologies

Haifa University Campus

Mt. Carmel, 31905, Haifa, Israel

E-mail: hartman@il.ibm.com

David Kreische

imbus AG

Kleinseebacher Str. 9, 91096 Moehrendorf, Germany

E-mail: david.kreische@imbus.de

Library of Congress Control Number: 2005935882

CR Subject Classification (1998): C.2, D.2, D.3, F.3, C.3, H.4

ISSN 0302-9743

ISBN-10 3-540-30026-0 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-30026-7 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springeronline.com

© Springer-Verlag Berlin Heidelberg 2005

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11581741 06/3142 5 4 3 2 1 0

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

New York University, NY, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Lecture Notes in Computer Science

For information about Vols. 1–3697

please contact your bookseller or Springer

Vol. 3807: M. Dean, Y. Guo, W. Jun, R. Kaschek, S. Krishnaswamy, Z. Pan, Q.Z. Sheng (Eds.), *Web Information Systems – WISE 2005 Workshops*. XV, 275 pages. 2005.

Vol. 3806: A.H. H. Ngu, M. Kitsuregawa, E.J. Neuhold, J.-Y. Chung, Q.Z. Sheng (Eds.), *Web Information Systems Engineering – WISE 2005*. XXI, 771 pages. 2005.

Vol. 3799: A. Rodríguez, I.F. Cruz, S. Levashkin (Eds.), *GeoSpatial Semantics*. X, 259 pages. 2005.

Vol. 3793: T. Conte, N. Navarro, W.-m. W. Hwu, M. Valero, T. Ungerer (Eds.), *High Performance Embedded Architectures and Compilers*. XIII, 317 pages. 2005.

Vol. 3792: I. Richardson, P. Abrahamsson, R. Messnarz (Eds.), *Software Process Improvement*. VIII, 215 pages. 2005.

Vol. 3791: A. Adi, S. Stoutenburg, S. Tabet (Eds.), *Rules and Rule Markup Languages for the Semantic Web*. X, 225 pages. 2005.

Vol. 3790: G. Alonso (Ed.), *Middleware 2005*. XIII, 443 pages. 2005.

Vol. 3789: A. Gelbukh, Á. de Albornoz, H. Terashima-Marín (Eds.), *MICA1 2005: Advances in Artificial Intelligence*. XXVI, 1198 pages. 2005. (Subseries LNAI).

Vol. 3785: K.-K. Lau, R. Banach (Eds.), *Formal Methods and Software Engineering*. XIV, 496 pages. 2005.

Vol. 3784: J. Tao, T. Tan, R.W. Picard (Eds.), *Affective Computing and Intelligent Interaction*. XIX, 1008 pages. 2005.

Vol. 3781: S.Z. Li, Z. Sun, T. Tan, S. Pankanti, G. Chollet, D. Zhang (Eds.), *Advances in Biometric Person Authentication*. XI, 250 pages. 2005.

Vol. 3780: K. Yi (Ed.), *Programming Languages and Systems*. XI, 435 pages. 2005.

Vol. 3779: H. Jin, D. Reed, W. Jiang (Eds.), *Network and Parallel Computing*. XV, 513 pages. 2005.

Vol. 3777: O.B. Lupanov, O.M. Kasim-Zade, A.V. Chaskin, K. Steinhöfel (Eds.), *Stochastic Algorithms: Foundations and Applications*. VIII, 239 pages. 2005.

Vol. 3775: J. Schönwälder, J. Serrat (Eds.), *Ambient Networks*. XIII, 281 pages. 2005.

Vol. 3773: A.S. Cortés, M.L. Cortés (Eds.), *Progress in Pattern Recognition, Image Analysis and Applications*. XX, 1094 pages. 2005.

Vol. 3772: M. Consens, G. Navarro (Eds.), *String Processing and Information Retrieval*. XIV, 406 pages. 2005.

Vol. 3770: J. Akoka, S.W. Liddle, I.-Y. Song, M. Bertolotto, I. Comyn-Wattiau, W.-J. van den Heuvel, M. Kolp, J.C. Trujillo, C. Kop, H.C. Mayr (Eds.), *Perspectives in Conceptual Modeling*. XXII, 476 pages. 2005.

Vol. 3768: Y.-S. Ho, H.J. Kim (Eds.), *Advances in Multimedia Information Processing – PCM 2005, Part II*. XXVIII, 1088 pages. 2005.

Vol. 3767: Y.-S. Ho, H.J. Kim (Eds.), *Advances in Multimedia Information Processing – PCM 2005, Part I*. XXVIII, 1022 pages. 2005.

Vol. 3766: N. Sebe, M.S. Lew, T.S. Huang (Eds.), *Computer Vision in Human-Computer Interaction*. X, 231 pages. 2005.

Vol. 3765: Y. Liu, T. Jiang, C. Zhang (Eds.), *Computer Vision for Biomedical Image Applications*. X, 563 pages. 2005.

Vol. 3764: S. Tixeul, T. Herman (Eds.), *Self-Stabilizing Systems*. VIII, 229 pages. 2005.

Vol. 3762: R. Meersman, Z. Tari, P. Herrero (Eds.), *On the Move to Meaningful Internet Systems 2005: OTM Workshops*. XXXI, 1228 pages. 2005.

Vol. 3761: R. Meersman, Z. Tari (Eds.), *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Part II*. XXVII, 653 pages. 2005.

Vol. 3760: R. Meersman, Z. Tari (Eds.), *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Part I*. XXVII, 921 pages. 2005.

Vol. 3759: G. Chen, Y. Pan, M. Guo, J. Lu (Eds.), *Parallel and Distributed Processing and Applications – ISPA 2005 Workshops*. XIII, 669 pages. 2005.

Vol. 3758: Y. Pan, D. Chen, M. Guo, J. Cao, J. Dongarra (Eds.), *Parallel and Distributed Processing and Applications*. XXIII, 1162 pages. 2005.

Vol. 3757: A. Rangarajan, B. Vemuri, A.L. Yuille (Eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition*. XII, 666 pages. 2005.

Vol. 3756: J. Cao, W. Nejdl, M. Xu (Eds.), *Advanced Parallel Processing Technologies*. XIV, 526 pages. 2005.

Vol. 3754: J. Dalmáu Royo, G. Hasegawa (Eds.), *Management of Multimedia Networks and Services*. XII, 384 pages. 2005.

Vol. 3753: O.F. Olsen, L.M.J. Florack, A. Kuijper (Eds.), *Deep Structure, Singularities, and Computer Vision*. X, 259 pages. 2005.

Vol. 3752: N. Paragios, O. Faugeras, T. Chan, C. Schnörr (Eds.), *Variational, Geometric, and Level Set Methods in Computer Vision*. XI, 369 pages. 2005.

Vol. 3751: T. Magedanz, E.R. M. Madeira, P. Dini (Eds.), *Operations and Management in IP-Based Networks*. X, 213 pages. 2005.

Vol. 3750: J.S. Duncan, G. Gerig (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005, Part II*. XL, 1018 pages. 2005.

- Vol. 3749: J.S. Duncan, G. Gerig (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005, Part I*. XXXIX, 942 pages. 2005.
- Vol. 3748: A. Hartman, D. Kreische (Eds.), *Model Driven Architecture – Foundations and Applications*. IX, 349 pages. 2005.
- Vol. 3747: C.A. Maziero, J.G. Silva, A.M.S. Andrade, F.M.d. Assis Silva (Eds.), *Dependable Computing*. XV, 267 pages. 2005.
- Vol. 3746: P. Bozanis, E.N. Houstis (Eds.), *Advances in Informatics*. XIX, 879 pages. 2005.
- Vol. 3745: J.L. Oliveira, V. Maojo, F. Martín-Sánchez, A.S. Pereira (Eds.), *Biological and Medical Data Analysis*. XII, 422 pages. 2005. (Subseries LNBI).
- Vol. 3744: T. Magedanz, A. Karmouch, S. Pierre, I. Venieris (Eds.), *Mobility Aware Technologies and Applications*. XIV, 418 pages. 2005.
- Vol. 3740: T. Srikanthan, J. Xue, C.-H. Chang (Eds.), *Advances in Computer Systems Architecture*. XVII, 833 pages. 2005.
- Vol. 3739: W. Fan, Z.-h. Wu, J. Yang (Eds.), *Advances in Web-Age Information Management*. XXIV, 930 pages. 2005.
- Vol. 3738: V.R. Syrotiuk, E. Chávez (Eds.), *Ad-Hoc, Mobile, and Wireless Networks*. XI, 360 pages. 2005.
- Vol. 3735: A. Hoffmann, H. Motoda, T. Scheffer (Eds.), *Discovery Science*. XVI, 400 pages. 2005. (Subseries LNAI).
- Vol. 3734: S. Jain, H.U. Simon, E. Tomita (Eds.), *Algorithmic Learning Theory*. XII, 490 pages. 2005. (Subseries LNAI).
- Vol. 3733: P. Yolum, T. Güngör, F. Gürgen, C. Özturan (Eds.), *Computer and Information Sciences - ISCIS 2005*. XXI, 973 pages. 2005.
- Vol. 3731: F. Wang (Ed.), *Formal Techniques for Networked and Distributed Systems - FORTE 2005*. XII, 558 pages. 2005.
- Vol. 3729: Y. Gil, E. Motta, V. R. Benjamins, M.A. Musen (Eds.), *The Semantic Web – ISWC 2005*. XXIII, 1073 pages. 2005.
- Vol. 3728: V. Paliouras, J. Vounckx, D. Verkest (Eds.), *Integrated Circuit and System Design*. XV, 753 pages. 2005.
- Vol. 3726: L.T. Yang, O.F. Rana, B. Di Martino, J. Dongarra (Eds.), *High Performance Computing and Communications*. XXVI, 1116 pages. 2005.
- Vol. 3725: D. Borriore, W. Paul (Eds.), *Correct Hardware Design and Verification Methods*. XII, 412 pages. 2005.
- Vol. 3724: P. Fraigniaud (Ed.), *Distributed Computing*. XIV, 520 pages. 2005.
- Vol. 3723: W. Zhao, S. Gong, X. Tang (Eds.), *Analysis and Modelling of Faces and Gestures*. XI, 4234 pages. 2005.
- Vol. 3722: D. Van Hung, M. Wirsing (Eds.), *Theoretical Aspects of Computing – ICTAC 2005*. XIV, 614 pages. 2005.
- Vol. 3721: A. Jorge, L. Torgo, P.B. Brazdil, R. Camacho, J. Gama (Eds.), *Knowledge Discovery in Databases: PKDD 2005*. XXIII, 719 pages. 2005. (Subseries LNAI).
- Vol. 3720: J. Gama, R. Camacho, P.B. Brazdil, A. Jorge, L. Torgo (Eds.), *Machine Learning: ECLM 2005*. XXIII, 769 pages. 2005. (Subseries LNAI).
- Vol. 3719: M. Hobbs, A.M. Goscinski, W. Zhou (Eds.), *Distributed and Parallel Computing*. XI, 448 pages. 2005.
- Vol. 3718: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), *Computer Algebra in Scientific Computing*. XII, 502 pages. 2005.
- Vol. 3717: B. Gramlich (Ed.), *Frontiers of Combining Systems*. X, 321 pages. 2005. (Subseries LNAI).
- Vol. 3716: L. Delcambre, C. Kop, H.C. Mayr, J. Mylopoulos, Ó. Pastor (Eds.), *Conceptual Modeling – ER 2005*. XVI, 498 pages. 2005.
- Vol. 3715: E. Dawson, S. Vaudenay (Eds.), *Progress in Cryptology – Mycrypt 2005*. XI, 329 pages. 2005.
- Vol. 3714: H. Obbink, K. Pohl (Eds.), *Software Product Lines*. XIII, 235 pages. 2005.
- Vol. 3713: L.C. Briand, C. Williams (Eds.), *Model Driven Engineering Languages and Systems*. XV, 722 pages. 2005.
- Vol. 3712: R. Reussner, J. Mayer, J.A. Stafford, S. Overhage, S. Becker, P.J. Schroeder (Eds.), *Quality of Software Architectures and Software Quality*. XIII, 289 pages. 2005.
- Vol. 3711: F. Kishino, Y. Kitamura, H. Kato, N. Nagata (Eds.), *Entertainment Computing – ICEC 2005*. XXIV, 540 pages. 2005.
- Vol. 3710: M. Barni, I. Cox, T. Kalker, H.J. Kim (Eds.), *Digital Watermarking*. XII, 485 pages. 2005.
- Vol. 3709: P. van Beek (Ed.), *Principles and Practice of Constraint Programming – CP 2005*. XX, 887 pages. 2005.
- Vol. 3708: J. Blanc-Talon, W. Philips, D.C. Popescu, P. Scheunders (Eds.), *Advanced Concepts for Intelligent Vision Systems*. XXII, 725 pages. 2005.
- Vol. 3707: D.A. Peled, Y.-K. Tsay (Eds.), *Automated Technology for Verification and Analysis*. XII, 506 pages. 2005.
- Vol. 3706: H. Fukú, S. Lukosch, A.C. Salgado (Eds.), *Groupware: Design, Implementation, and Use*. XII, 378 pages. 2005.
- Vol. 3704: M. De Gregorio, V. Di Maio, M. Frucci, C. Musio (Eds.), *Brain, Vision, and Artificial Intelligence*. XV, 556 pages. 2005.
- Vol. 3703: F. Fages, S. Soliman (Eds.), *Principles and Practice of Semantic Web Reasoning*. VIII, 163 pages. 2005.
- Vol. 3702: B. Beckert (Ed.), *Automated Reasoning with Analytic Tableaux and Related Methods*. XIII, 343 pages. 2005. (Subseries LNAI).
- Vol. 3701: M. Coppo, E. Lodi, G. M. Pinna (Eds.), *Theoretical Computer Science*. XI, 411 pages. 2005.
- Vol. 3700: J.F. Peters, A. Skowron (Eds.), *Transactions on Rough Sets IV*. X, 375 pages. 2005.
- Vol. 3699: C.S. Calude, M.J. Dinneen, G. Păun, M. J. Pérez-Jiménez, G. Rozenberg (Eds.), *Unconventional Computation*. XI, 267 pages. 2005.
- Vol. 3698: U. Furbach (Ed.), *KI 2005: Advances in Artificial Intelligence*. XIII, 409 pages. 2005. (Subseries LNAI).

¥490.88元

Preface

The European Conference on Model Driven Architecture - Foundations and Applications (ECMDA-FA) is a new conference dedicated to the study and industrial adoption of the model-driven approach to software engineering. It has grown out of a number of workshops and smaller conferences in the area of model driven development and model driven architecture (MDATM). The conference is dedicated to providing a forum for cross fertilization between the European software industry and the academic community. We aim to present the industrial experience and highlight the pain points of industry in order to promote focused academic research that will bring real value to society. At the same time, we hope to challenge industry leaders to conduct a realistic appraisal of the emerging technologies presented by academics, consultants, and tool vendors, and eventually to adopt the model driven approach.

The conference provides both a forum for the papers judged as being of the highest quality and a venue for workshops, tutorials and tool exhibitions on model driven software engineering. This year, we are host to five workshops and four tutorials in subject matter ranging from the highly theoretical to the practical industrial aspects of MDA and a tool exhibition featuring nine commercial and seven open source or academic tools. This volume contains nine papers from the applications track and fifteen from the foundations track, chosen from 82 submitted papers. These works provide the latest and most relevant information on model driven software engineering in the industrial and academic spheres.

I would like to express my thanks to all the members of the steering committee, the program committee, and the referees, who gave freely of their time and wisdom to make this conference a success. The ECMDA-FA is supported by the European Commission's Information Society Technologies (IST) initiative, and by the Object Management Group (OMG).

November 2005

Alan Hartman
Program Chair
ECMDA-FA'2005

Organization

Steering Committee

| | |
|----------------------------|---------------------------|
| Program Chair: | Alan Hartman (IBM) |
| Local Arrangements Chair: | David Kreische (imbus AG) |
| Workshop Chair: | Arend Rensink (Twente U) |
| Tools and Tutorials Chair: | Jos Warmer (Ordina) |
| | Uwe Assman (Dresden TU) |
| | Asier Azaceta (ESI) |
| | David Akehurst (Kent U) |

Programm Committee

| | | |
|---------------------|---------------------|---------------------|
| Jan Aagedal | Miguel A. de Miguel | Bran Selic |
| Mehmet Aksit | Philippe Desfray | Marten van Sinderen |
| Sergio Bandinelli | Reiko Heckel | Gerd Wagner |
| Mariano Belaunde | James J. Hunt | James Willans |
| Jean Bezivin | Jean-Marc Jezequel | Jim Woodcock |
| Xavier Blanc | Anneke Kleppe | |
| Manfred Broy | Richard Paige | |
| Krzysztof Czarnecki | Bernhard Rumpe | |

Additional Reviewers

| | | |
|-------------------------|-----------------------|-----------------------|
| Andreas Bauer | Jan Jürjens | Sergey Olvovsky |
| Machiel van der Bijl | Dave Kelsey | Jonathan Ostroff |
| Peter Braun | Mila Keren | Fiona Polack |
| Phil Brooke | Andrei Kirshin | Gerhard Popp |
| Maria Victoria Cengarle | Holger Krahn | Martin Rapp |
| Anthony Elder | Sergey Lukichev | Julia Rubin |
| Eitan Farchi | Keith Mantell | Thomas Roßner |
| Boris Gajanovic | Frank Marschall | Martin Schindler |
| Adrian Giurca | Tor Neple | Yael Shaham-Gafni |
| Roy Grønmo | Dimitrios Kolovos | Zoe Stephenson |
| Hans Grönniger | Shiri Kremer-Davidson | Alexander Wißpeintner |
| Wilke Havinga | Jon Oldevik | |

Table of Contents

MDA Development Processes

| | |
|---|----|
| Applying MDA to Voice Applications: An Experience in Building an MDA Tool Chain <i>Maria José Presso, Mariano Belaunde</i> | 1 |
| MDA, Meta-Modelling and Model Transformation: Introducing New Technology into the Defence Industry <i>Tony Bloomfield</i> | 9 |
| Using Domain Driven Development for Monitoring Distributed Systems <i>Rainer Burgstaller, Egon Wuchner, Ludger Fiege, Michael Becker, Thomas Fritz</i> | 19 |

MDA for Embedded and Real-Time Systems

| | |
|--|----|
| Model-Driven Architecture for Hard Real-Time Systems: From Platform Independent Models to Code <i>Sven Burmester, Holger Giese, Wilhelm Schäfer</i> | 25 |
| Model-Driven Performance Analysis of UML Design Models Based on Stochastic Process Algebra <i>Naoshi Tabuchi, Naoto Sato, Hiroaki Nakamura</i> | 41 |
| MDA Components: A Flexible Way for Implementing the MDA Approach <i>Reda Bendraou, Philippe Desfray, Marie-Pierre Gervais</i> | 59 |

MDA and Component-Based Software Engineering

| | |
|---|-----|
| An MDA Approach for Adaptable Components <i>Steffen Göbel</i> | 74 |
| Layered Ontological Modelling for Web Service-Oriented Model-Driven Architecture <i>Claus Pahl</i> | 88 |
| Model-Driven Development – Hot Spots in Business Information Systems <i>Bernhard Humm, Ulf Schreier, Johannes Siedersleben</i> | 103 |

Metamodelling

Semantic Anchoring with Model Transformations
Kai Chen, Janos Sztipanovits, Sherif Abdelwalhed, Ethan Jackson . . . 115

On Some Properties of Parameterized Model Application
Alexis Muller, Olivier Caron, Bernard Carré, Gilles Vanwormhoudt .. 130

A Comparative Study of Metamodel Integration and Interoperability
in UML and Web Services
Athanasios Staikopoulos, Behzad Bordbar 145

Control Flow Analysis of UML 2.0 Sequence Diagrams
Vahid Garousi, Lionel C. Briand, Yvan Labiche 160

Designing a Domain-Specific Contract Language: A Metamodelling
Approach
Zhan En Chan, Richard F. Paige 175

Making Metamodels Aware of Concrete Syntax
Frédéric Fondement, Thomas Baar 190

Model Transformation

XRound: Bidirectional Transformations and Unifications Via a
Reversible Template Language
Howard Chivers, Richard F. Paige 205

Towards General Purpose, High Level, Software Languages
Anneke Kleppe 220

Toward Standardised Model to Text Transformations
Jon Oldevik, Tor Neple, Roy Grønmo, Jan Aagedal, Arne-J. Berre ... 239

On Relationships Between Query Models
Dominik Stein, Stefan Hanenberg, Rainer Unland 254

Transformations Between UML and OWL-S
Roy Grønmo, Michael C. Jaeger, Hjørdis Hoff 269

A Graphical Specification of Model Transformations with Triple Graph
Grammars
Lars Grunske, Leif Geiger, Michael Lawley 284

Model Synchronization and Consistency

| | |
|--|-----|
| Horizontal Transformation of PSMs <i>Jamal Abd-Ali, Karim El Guemhioui</i> | 299 |
| Automatic Support for Traceability in a Generic Model Management Framework <i>Artur Boronat, José Á. Carsí, Isidro Ramos</i> | 316 |
| Synchronizing Cardinality-Based Feature Models and Their Specializations <i>Chang Hwan, Peter Kim, Krzysztof Czarnecki</i> | 331 |
| Author Index | 349 |

Applying MDA to Voice Applications: An Experience in Building an MDA Tool Chain

Maria José Presso and Mariano Belaunde

France Telecom, Div. R&D,
2, Av Pierre Marzin, 22307 Lannion, France
{mariajose.presso, mariano.belaunde}@francetelecom.com

Abstract. Before a development project based on MDA can start, an important effort has to be done in order to select and adapt existing MDA technology to the considered application domain. This article presents our experience in applying MDA technology to the voice application domain. It describes the iterative approach followed and discusses issues and needs raised by the experience in the area of building MDA tool chains.¹

1 Introduction

Interactive voice-based applications are specific telephony applications that are designed to allow end-users to interact with a machine using speech and telephone keys in order to request a service. The interaction – called a dialog – typically consists of a state machine that executes the logic of the conversation and that is capable of invoking business code which stands independently of the user interface mechanism – could be web, batch or speech-based. Because state-machines can be specified and modelled formally, it is possible to design a tool chain that automates large amounts of the dialog implementation. The application of model-driven techniques to this domain is without any doubt very promising. However, the question that arises is about the methods and the cost of building a complete environment capable of taking full advantage of models: not only ensuring automated code production but also offering user-friendly interfaces to designers, model simulation and test generation.

A general methodology for MDA-based development has been defined in [1]. The authors define the main phases, and make a distinction between preparation and execution activities: execution activities refer to actual project execution, during which software artefacts and final products are produced, while preparation activities typically start before project execution, and setup the context that allows the reuse of knowledge during the project. The preparation activities can be seen as selecting and adapting existing generic MDA technology in order to define an MDA approach for the considered application domain and provide an appropriate tool chain.

¹ The work presented here has been partially carried out within the MODELWARE project. MODELWARE is a project co-funded by the European Commission under the "Information Society Technologies" Sixth Framework Programme (2002-2006). Information included in this document reflects only the author's views. The European Community is not liable for any use that may be made of the information contained herein.

Whereas [1] gives a general description of preparation activities and their chaining, there is currently little or no guidance available for them. In the current state of MDA technologies, these preparation activities demand an important effort which is not paid of by the first project and should be shared among a set of projects within the same domain. In particular, preparation has an impact on the first application project that follows it, as this first project necessarily requires iterations in the preparation activities.

This work presents our experience in building an MDA approach for the voice application domain and finishes by a discussion on the encountered issues and expectations.

2 MDD Preparation for the Voice Application Domain

According to [1], the preparation activities are divided into the *preliminary preparation phase*, the *detailed preparation phase*, and the *infrastructure set-up phase*. The *preliminary preparation* comprises the identification of the platform, the modelling language identification, the transformation identification and the traceability strategy definition. *Detailed preparation* comprises specification of modelling languages and specification of transformations. *Infrastructure setup* includes tool selection and metadata management.

In our project, these activities were performed in an iterative and incremental way, in order to better suit the needs of the users of the MDD environment involved in the execution phases, like voice dialog design, business application coding and functional testing. These users apply the tool facilities constructed by the preparation activities to produce the voice applications (the tool facilities are described later).

The preparation took place in three main stages. In each phase, some preparation activities were executed, together with some validation activities involving future users of the tool-chain, mainly service designers. The rest of this section presents the stages we followed.

2.1 Stage 1: Definition

In the first part of this stage, the current process of voice application creation was analysed and the integration of MDD techniques to this process was studied in order to identify the requirements for the voice development environment (VDE).

From this study, the following roles and their corresponding scenarios of use of the VDE were identified:

- the **service designer** uses the VDE model editor and simulator to model and simulate iteratively the dialogs of the application,
- the **usability practitioner** uses the VDE simulator to perform usability expertise on the dialogs of the service and he uses the VDE model editor to correct the dialog model,
- the **internal customer** (project owner) uses the VDE simulator a prototype of the service, to validate dialog design,
- the **service implementer** implements the service in the target platform, ideally by completing the skeletons generated by the modelling tool,
- the **service validator** produces the conformance test cases using the VDE test generation tool.

In order to serve as a conceptual basis for the VDE, a meta-model for platform independent modelling of voice applications was defined and UML 2 was chosen as a concrete syntax. Thus, a UML 2 profile for voice application models was defined².

Although the choice of UML for the concrete syntax may seem obvious, UML 2 being "the" modelling language standard, we will see later that this choice induces a significant cost in the infrastructure setup phase. For this reason, it is important to recall the rationale behind his choice:

- Voice application logic can easily be assimilated to a reactive state machine: the application reacts to user input such as voice and telephone keys, and produces output for the user: the vocal messages. The concepts of states and transitions are used in the voice application meta-model and supported by UML.
- Voice applications usually interact with the enterprise's information system. As UML is used as a modelling language in the information system domain, using the same language for voice applications allows to seamlessly integrate information system models with voice application models.
- Communication services are becoming integrated and multimodal. The use of a standard, largely used notation is expected to favour future integration with other services and modalities.
- Existing modelling tools and skills can be reused.

Also at this stage, the architecture of the VDE was defined (see Figure 1) and some of the tools involved were selected. A UML tool was chosen to play the role of model editor and model repository and the criteria for its selection were defined. Among the criteria defined, the ones that differentiated the tools were : i) the support for UML 2 transition oriented syntax for state machines (this notation had been found easier to read by users than the state oriented syntax), ii) the support for rigorous syntax checking (in particular for actions) and iii) model simulation/execution capabilities. TAU G2 from Telelogic was chosen as modelling tool.

Following this first round of preparation activities, we conducted some experiments in order to determine the ability of the profile to capture the intended service logic, verify that service designers could feel comfortable with the tool and to identify the necessary adaptations to the modelling tool (i.e. specific functionalities necessary to better support the voice application profile). These experiments consisted in modelling some existing services with the proposed profile and tool. Modelling was initiated by a modelling expert and a service designer working in pairs and finished by the service designer.

As a result of this phase, we could see that the proposed profile captured most of the necessary elements to describe a voice application, but it should be enhanced to support executable modelling of messages and the definition of grammars for voice recognition. Designers (who are not modelling experts) could get familiar with the modelling tool with a fair amount of effort. The main need for tool adaptation that came up was that a high level of support for the specification of messages allowing to reuse message parts, as well as facilities to read them during the dialog specification task. Also, high level commands for the creation of the other domain elements should

² This profile was later used as a basis for a submission to the OMG's RFP for a Metamodel and UML Profile for voice applications [3].

be available (such as creating a dialog). The restitution of the specification in the form of a document should be optimized in order to limit its volume and improve readability, hyperlink navigation should be available in the documents.

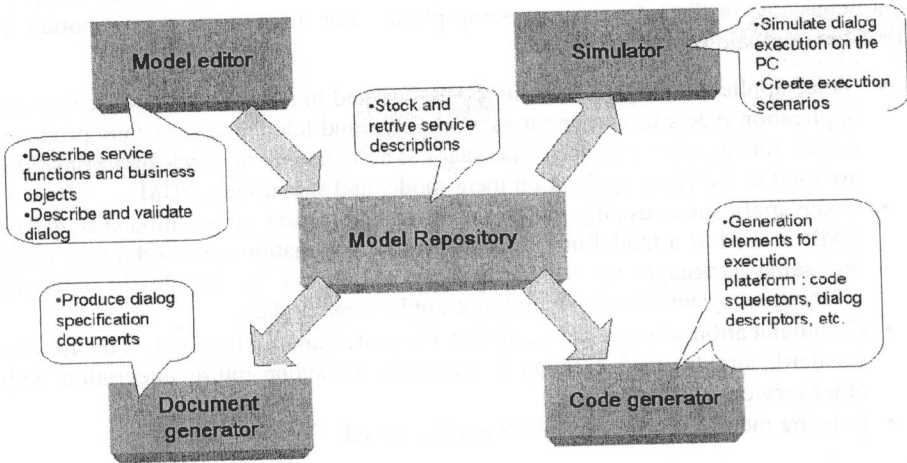


Fig. 1. Architecture of the MDD Voice Development Environment (VDE)

2.2 Stage 2: VDE Development – Iteration 1

The second stage consisted mainly in the development of a first version of the tool chain, offering assistance for the creation of dialog modelling elements (dialogs, messages, recognition interpretation concepts, etc.), documentation generation and a limited form of dialog simulation. This version was developed using scripting and code generation capabilities provided by the modelling tool. Concretely, it appears as a plug-in to the modelling tool and a separate telephone-like GUI connected to a text to speech engine, that allows the designer to execute the dialog logic of the modelled service and evaluate its appropriateness, its ergonomics, etc.

After the development of this first version, a second row of experiments took place, which consisted in using the tool chain to enhance the service models produced in the first stage, and use the document generation and simulation functionalities for this models.

Although this first version of the tool chain was very promising and showed that useful functionality for service creation could be offered, it presented some limitations: the GUI for modelling assistance that could be developed through scripting was limited and not satisfying from the point of view of ergonomics; the service simulation was not able to propose the possible inputs in a given situation, neither to go arbitrarily back and forth into the execution tree. The scripting technique used for development posed maintainability issues and was not appropriate to support a growing software.

At the same time, studies were carried out about simulation and test generation for voice services. These studies showed that existing simulation technologies based

on the IF language[2] provided the necessary level of support to build a simulator for voice services that overcomes the limitation of the method employed in the first version.

At the end of this stage the decision was taken to build a more industrial version of the tool chain based on a programming language (rather than scripting), to offer richer GUI capability in particular for message creation, and to provide the service simulation functionality through model simulation techniques, rather than code generation.

2.3 Stage 3: VDE Development – Iteration 2

This stage started by the definition of the architecture for the modelling tool plug-in, and the choice of the implementation technology. The main characteristic of this architecture was the definition of a layer that provides a view of the underlying UML model in the terms of the voice application metamodel. This layer implements an on-the-fly bi-directional transformation between UML and the voice application metamodel. This layer provides an adapted API and is used as a basis to develop the GUI and a set of generators that implemented various model transformations. Also, the architecture proposed a way for simple integration of the different generators in the plug-in. The generators provided at this stage were:

- document generators, which produce documents according to different templates and in html and MS Word formats. These generators use an intermediate XML generation phase, followed by XSLT transformations,
- an XMI generator, which exports the model in the terms of the voice application metamodel. This generator uses the adaptation layer API, and was automatically generated (and re-generated as needed) from the voice application metamodel,
- a generator that produces an IF model for service simulation and test generation,
- a code generator having as target an n-tier architecture using VoiceXML. The generated code executes in the application server tier and produces on-the-fly the presentation pages in VoiceXML. The generated code integrates in a framework (which was also developed in this phase), that provides the basis for the execution of a dialog state machine and VoiceXML generation.

The first three generators above were directly integrated into the plug-in, in order to facilitate the installation of the toolkit in the user's workstation and its use by the service designers, while the last one is external and uses the results of the XMI export. Something important to note about this stage is that the metamodel was called to change often, as the implementation of transformations asked for corrections or improvements. As different transformations were developed in parallel, the changes asked by one of them had an impact on the others.

2.4 Stage 4: Pilots

The last stage is that of pilot projects. These are the first projects using the MDD chain (in the terms of [1], the first runs of the "execution" phase). The stage is still in progress at the time of writing. During this phase, iteration with preparation activities

goes on, mainly to adapt the code generator to the project's target platform and to add extra functions asked by the pilot projects. An important effort in this stage is spent on user training and support.

3 Discussion

The result of the preparation activities described in the previous section is a process and a tool chain providing a high degree of automation. Starting from the PIM model, the tool chain produces automatically a simulation of the dialog, functional test cases, and the executable code for the dialog logic, and a is good representative of the MDA vision. However, these activities consumed an important effort, that should be shared by several projects. In this section we briefly discuss some of the issues and expectations that come from our experience in building this MDA tool chain.

In our experiment, we encountered a strong user's demand to have a rich GUI for modelling in terms of the domain vocabulary. This appeared as a critical issue for the adoption of the tool chain. As UML had been chosen as a concrete syntax, this request lead to important extensions to the modelling tool in the form of a plug-in. The ability to extend the modelling tool using a full-fledged programming language was necessary to develop the required GUI and to apply good engineering practices (such as the MVC pattern) to this development.

The above issue comes to the famous problem of whether a general-purpose modelling tool should be specialized or whether the tool should be built from scratch. Beyond tool usage is the question whether the specific language for the considered domain – in our case voice dialog definition – has to be built on top of an existing language – like UML, or a new language should be defined – typically using MOF or an XML schema. It is easy to adhere to the principle of maintaining the distinction between the abstract syntax (the domain metamodel) and the concrete syntax (given by a UML profile or a textual notation) since it provides potentially much more freedom – ability to use various concrete syntaxes - and is more comfortable for domain designers – since the vocabulary used is directly the one of the domain. However, as our experiment has demonstrated, maintaining this distinction potentially induces a high cost to the development of the tool chain. In our experiment, we used an "API adaptation" technique which allows to program a specific GUI and model transformations within the UML tool by using an API dedicated to the domain metamodel – instead of using the general-purpose UML-based API. This technique presents interesting advantages from the engineering point of view : i) the knowledge about the mapping between UML and the domain metamodel is localized, ii) the coding of the GUI is simplified, since the complexity of UML is hidden iii) the model transformations can be implemented in domain terms and are thus facilitated, and iv) the XMI generator exporting the model in the voice metamodel terms can be produced and updated automatically from the metamodel itself. However, one of the important problems encountered at this level was the instability of the metamodel, which in general changed more often than the graphical and the textual notation. Ideally, to solve the instability problem, the mapping between the metamodel and the