Andreas Prinz
Rick Reed
Jeanne Reed (Eds.)

LNCS 3530

SDL 2005: Model Driven

12th International SDL Forum
Grimstad, Norway, June 2005
Proceedings

Springer

Andreas Prinz Rick Reed
Jeanne Reed (Eds.)

SDL 2005: Model Driven

12th International SDL Forum
Grimstad, Norway, June 20-23, 2005
Proceedings

Springer

Volume Editors

Andreas Prinz
Agder University College, Faculty of Engineering and Science
Grooseveien 36, 4876 Grimstad, Norway
E-mail: andreas.prinz@hia.no

Rick Reed
Jeanne Reed
Telecommunications Software Engineering Limited
The Laurels, Victoria Road, Windermere, Cumbria LA23 2DL, United Kingdom
E-mail: {rickreed, jeanne}@tseng.co.uk

Library of Congress Control Number: 2005927490

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

ISSN 0302-9743
ISBN-10 3-540-26612-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-26612-9 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11506843 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3530

Preface

This volume contains the papers presented at the 12th SDL Forum, Grimstad, Norway.

The SDL Forum was first held in 1982, and then every two years from 1985. Initially the Forum was concerned only with the Specification and Description Language that was first standardized in the 1976 Orange Book of the International Telecommunication Union (ITU). Since then, many developments took place and the language has undergone several changes.

However, the main underlying paradigm has survived, and it is the reason for the success of the Specification and Description Language in many projects. This paradigm is based on the following important principles of distributed applications:

Communication: large systems tend to be described using smaller parts that communicate with each other;

State: the systems are described on the basis of an explicit notion of state;

State change: the behavior of the system is described in terms of (local) changes of the state.

The original language is not the only representative for this kind of paradigm, so the scope of the SDL Forum was extended quite soon after the first few events to also include other ITU standardized languages of the same family, such as MSC, ASN.1 and TTCN. This led to the current scope of System Design Languages covering all stages of the development process including in particular SDL, MSC, UML, ASN.1, eODL, TTCN, and URN. The focus is clearly on the advantages to users, and how to get from these languages the same advantage given by the ITU Specification and Description Language: code generation from high-level specifications.

Not only have the languages and the scope of the SDL Forum evolved, but for the first time the Programme Committee for SDL2005 decided to have short papers as well as the normal full papers with a strong scientific background for full presentation and publication. The rationale is that the SDL Forum is targeted at a mixture of participants from research to industrial engineering backgrounds, and the inclusion of the short papers allows more issues to be presented and discussed while still maintaining normal conference standards for full papers. The short papers are of essentially three kinds:

Application reports. The SDL Forum has a tradition of publishing application papers, and users need such reports to benefit from the experience of other users, in particular where the reports contain quantitative results on cost effectiveness. Typically such reports do not contain advances in techniques or technology that justify presentation as full papers.

Tool reports. Similarly, reports on new tools or new releases of tools are of interest, without requiring the paper to describe a major advance – though it may be an advance in use or technology for the particular tool.

Position papers. These are papers that raise legitimate issues that need discussion, and the paper is a contribution to the discussion but from its content it could not be considered on the same basis as a fully investigated research paper.

No doubt the concept of short papers will be further developed for future events.

One important facet of the SDL Forum is the concentration on real-world examples mentioned above, which is also present in the SDL design contest: following a '*tradition*' started in 2002, the 12th SDL Forum hosted an SDL design competition sponsored by SAFIRE SDL with cash prizes for the winning designs.[1]

As editors of this volume, we have read through all the papers and are pleased with the interesting and varied selection taken by the Programme Committee. You will find all aspects of System Design Languages covered in this volume, ranging from state-of-the-art research results to modern application examples.

April 2005 Andreas Prinz and Rick Reed

SDL Forum Society

The SDL Forum Society is a not-for-profit organization that in addition to running the SDL Forum:

- runs the SAM (SDL and MSC) workshop every 2 years between SDL Forum years;
- is a body recognized by ITU-T as co-developing the Z.100 to Z.109 and Z.120 to Z.129 standards;
- promotes the ITU-T System Design Languages.

For more information on the SDL Forum Society, see `www.sdl-forum.org`.

[1] The descriptions of the winning entries are to be found at the SOLINET web pages.

Organization

Each SDL Forum is organized by the SDL Forum Society with the help of local organizers. The Organizing Committee consists of the Board of the SDL Forum Society plus the local organizers and others as needed depending on the actual event. For SDL 2005 the local organizers from Agder University College need to be thanked for their effort to ensure that everything was in place for the presentation of the papers in this volume.

Organizing Committee

Chairman, SDL Forum Society:	Rick Reed (TSE Ltd.)
Treasurer, SDL Forum Society:	Uwe Glässer (Simon Fraser University)
Secretary, SDL Forum Society:	Andreas Prinz (Agder University College)
ITU-T:	Georges Sebek (Counsellor to Study Group 17)
Local Organizers	Prof. Andreas Prinz
Agder University College:	Maren Assev, Lisanne Prinz, Ingunn Espedal

Programme Committee

Daniel Amyot, University of Ottawa, Canada
Laurent Doldi, TransMeth Sud-Ouest, France
Fabrice Dubois, France Télécom, France
Anders Ek, Telelogic, Sweden
Joachim Fischer, Humboldt-Universität zu Berlin, Germany
Emmanuel Gaudin, PragmaDev, France
Uwe Glässer, Simon Fraser University, Canada
Reinhard Gotzhein, University of Kaiserslautern, Germany
Jens Grabowski, Georg-August University Göttingen, Germany
Susanne Graf, Verimag, France
Øystein Haugen, University of Oslo, Norway
Dieter Hogrefe, Georg-August University Göttingen, Germany
Eckhardt Holz, Hasso-Plattner-Institut Potsdam, Germany
Sune Jakobsson, Telenor, Norway
Ferhat Khendek, Concordia University, Canada
Martin von Löwis, Hasso-Plattner-Institute Potsdam, Germany
Birger Møller-Pedersen, University of Oslo, Norway
Ian Oliver, Nokia, Finland
Anders Olsen, Cinderella, Denmark
Andreas Prinz, Agder University College, Norway

Steve Randall, PQM Consultants, UK
Rick Reed, TSE Ltd., UK
Richard Sanders, SINTEF, Norway
Amardeo Sarma, NEC, Germany
Ina Schieferdecker, Fraunhofer FOKUS, Germany
Edel Sherratt, University of Wales Aberystwyth, UK
William Skelton, SOLINET, Germany
Bent Vale, Ericsson, Norway
Thomas Weigert, Motorola, USA
Milan Zoric, ETSI, France

Thanks

A volume such as this could not, of course, exist without the contributions of the authors who are thanked for their work.

Moreover the organization was greatly assisted through the various sponsors that provided valuable support. SDL 2005 was sponsored by:

- Grimstad Kommune
- Cinderella ApS
- Xactium Ltd.
- PragmaDev
- Telelogic

Lecture Notes in Computer Science

For information about Vols. 1–3458

please contact your bookseller or Springer

Vol. 3569: F. Bacchus, T. Walsh (Eds.), Theory and Applications of Satisfiability Testing, SAT 2005. XII, 492 pages. 2005.

Vol. 3562: J. Mira, J.R. Álvarez (Eds.), Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Part II. XXIV, 636 pages. 2005.

Vol. 3561: J. Mira, J.R. Álvarez (Eds.), Mechanisms, Symbols, and Models Underlying Cognition, Part I. XXIV, 532 pages. 2005.

Vol. 3560: V.K. Prasanna, S. Iyengar, P. Spirakis, M. Welsh (Eds.), Distributed Computing in Sensor Systems. XV, 423 pages. 2005.

Vol. 3559: P. Auer, R. Meir (Eds.), Learning Theory. XI, 692 pages. 2005. (Subseries LNAI).

Vol. 3556: H. Baumeister, M. Marchesi, M. Holcombe (Eds.), Extreme Programming and Agile Processes in Software Engineering. XIV, 332 pages. 2005.

Vol. 3555: T. Vardanega, A. Wellings (Eds.), Reliable Software Technology – Ada-Europe 2005. XV, 273 pages. 2005.

Vol. 3552: H. de Meer, N. Bhatti (Eds.), Quality of Service – IWQoS 2005. XV, 400 pages. 2005.

Vol. 3547: F. Bomarius, S. Komi-Sirviö (Eds.), Product Focused Software Process Improvement. XIII, 588 pages. 2005.

Vol. 3543: L. Kutvonen, N. Alonistioti (Eds.), Distributed Applications and Interoperable Systems. XI, 235 pages. 2005.

Vol. 3541: N.C. Oza, R. Polikar, J. Kittler, F. Roli (Eds.), Multiple Classifier Systems. XII, 430 pages. 2005.

Vol. 3540: H. Kalviainen, J. Parkkinen, A. Kaarna (Eds.), Image Analysis. XXII, 1270 pages. 2005.

Vol. 3537: A. Apostolico, M. Crochemore, K. Park (Eds.), Combinatorial Pattern Matching. XI, 444 pages. 2005.

Vol. 3536: G. Ciardo, P. Darondeau (Eds.), Applications and Theory of Petri Nets 2005. XI, 470 pages. 2005.

Vol. 3535: M. Steffen, G. Zavattaro (Eds.), Formal Methods for Open Object-Based Distributed Systems. X, 323 pages. 2005.

Vol. 3533: M. Ali, F. Esposito (Eds.), Innovations in Applied Artificial Intelligence. XX, 858 pages. 2005. (Subseries LNAI).

Vol. 3532: A. Gómez-Pérez, J. Euzenat (Eds.), The Semantic Web: Research and Applications. XV, 728 pages. 2005.

Vol. 3531: J. Ioannidis, A. Keromytis, M. Yung (Eds.), Applied Cryptography and Network Security. XI, 530 pages. 2005.

Vol. 3530: A. Prinz, R. Reed, J. Reed (Eds.), SDL 2005: Model Driven. XI, 361 pages. 2005.

Vol. 3528: P.S. Szczepaniak, J. Kacprzyk, A. Niewiadomski (Eds.), Advances in Web Intelligence. XVII, 513 pages. 2005. (Subseries LNAI).

Vol. 3527: R. Morrison, F. Oquendo (Eds.), Software Architecture. XII, 263 pages. 2005.

Vol. 3526: S.B. Cooper, B. Löwe, L. Torenvliet (Eds.), New Computational Paradigms. XVII, 574 pages. 2005.

Vol. 3525: A.E. Abdallah, C.B. Jones, J.W. Sanders (Eds.), Communicating Sequential Processes. XIV, 321 pages. 2005.

Vol. 3524: R. Barták, M. Milano (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 320 pages. 2005.

Vol. 3523: J.S. Marques, N.P. de la Blanca, P. Pina (Eds.), Pattern Recognition and Image Analysis, Part II. XXVI, 733 pages. 2005.

Vol. 3522: J.S. Marques, N.P. de la Blanca, P. Pina (Eds.), Pattern Recognition and Image Analysis, Part I. XXVI, 703 pages. 2005.

Vol. 3521: N. Megiddo, Y. Xu, B. Zhu (Eds.), Algorithmic Applications in Management. XIII, 484 pages. 2005.

Vol. 3520: O. Pastor, J. Falcão e Cunha (Eds.), Advanced Information Systems Engineering. XVI, 584 pages. 2005.

Vol. 3519: H. Li, P. J. Olver, G. Sommer (Eds.), Computer Algebra and Geometric Algebra with Applications. IX, 449 pages. 2005.

Vol. 3518: T.B. Ho, D. Cheung, H. Li (Eds.), Advances in Knowledge Discovery and Data Mining. XXI, 864 pages. 2005. (Subseries LNAI).

Vol. 3517: H.S. Baird, D.P. Lopresti (Eds.), Human Interactive Proofs. IX, 143 pages. 2005.

Vol. 3516: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part III. LXIII, 1143 pages. 2005.

Vol. 3515: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part II. LXIII, 1101 pages. 2005.

Vol. 3514: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part I. LXIII, 1089 pages. 2005.

Vol. 3513: A. Montoyo, R. Muñoz, E. Métais (Eds.), Natural Language Processing and Information Systems. XII, 408 pages. 2005.

Vol. 3512: J. Cabestany, A. Prieto, F. Sandoval (Eds.), Computational Intelligence and Bioinspired Systems. XXV, 1260 pages. 2005.

Vol. 3510: T. Braun, G. Carle, Y. Koucheryavy, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XIV, 366 pages. 2005.

Vol. 3509: M. Jünger, V. Kaibel (Eds.), Integer Programming and Combinatorial Optimization. XI, 484 pages. 2005.

Vol. 3508: P. Bresciani, P. Giorgini, B. Henderson-Sellers, G. Low, M. Winikoff (Eds.), Agent-Oriented Information Systems II. X, 227 pages. 2005. (Subseries LNAI).

Vol. 3507: F. Crestani, I. Ruthven (Eds.), Information Context: Nature, Impact, and Role. XIII, 253 pages. 2005.

Vol. 3506: C. Park, S. Chee (Eds.), Information Security and Cryptology – ICISC 2004. XIV, 490 pages. 2005.

Vol. 3505: V. Gorodetsky, J. Liu, V. A. Skormin (Eds.), Autonomous Intelligent Systems: Agents and Data Mining. XIII, 303 pages. 2005. (Subseries LNAI).

Vol. 3504: A.F. Frangi, P.I. Radeva, A. Santos, M. Hernandez (Eds.), Functional Imaging and Modeling of the Heart. XV, 489 pages. 2005.

Vol. 3503: S.E. Nikoletseas (Ed.), Experimental and Efficient Algorithms. XV, 624 pages. 2005.

Vol. 3502: F. Khendek, R. Dssouli (Eds.), Testing of Communicating Systems. X, 381 pages. 2005.

Vol. 3501: B. Kégl, G. Lapalme (Eds.), Advances in Artificial Intelligence. XV, 458 pages. 2005. (Subseries LNAI).

Vol. 3500: S. Miyano, J. Mesirov, S. Kasif, S. Istrail, P. Pevzner, M. Waterman (Eds.), Research in Computational Molecular Biology. XVII, 632 pages. 2005. (Subseries LNBI).

Vol. 3499: A. Pelc, M. Raynal (Eds.), Structural Information and Communication Complexity. X, 323 pages. 2005.

Vol. 3498: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part III. L, 1077 pages. 2005.

Vol. 3497: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part II. L, 947 pages. 2005.

Vol. 3496: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part II. L, 1055 pages. 2005.

Vol. 3495: P. Kantor, G. Muresan, F. Roberts, D.D. Zeng, F.-Y. Wang, H. Chen, R.C. Merkle (Eds.), Intelligence and Security Informatics. XVIII, 674 pages. 2005.

Vol. 3494: R. Cramer (Ed.), Advances in Cryptology – EUROCRYPT 2005. XIV, 576 pages. 2005.

Vol. 3493: N. Fuhr, M. Lalmas, S. Malik, Z. Szlávik (Eds.), Advances in XML Information Retrieval. XI, 438 pages. 2005.

Vol. 3492: P. Blache, E. Stabler, J. Busquets, R. Moot (Eds.), Logical Aspects of Computational Linguistics. X, 363 pages. 2005. (Subseries LNAI).

Vol. 3489: G.T. Heineman, I. Crnkovic, H.W. Schmidt, J.A. Stafford, C. Szyperski, K. Wallnau (Eds.), Component-Based Software Engineering. XI, 358 pages. 2005.

Vol. 3488: M.-S. Hacid, N.V. Murray, Z.W. Raś, S. Tsumoto (Eds.), Foundations of Intelligent Systems. XIII, 700 pages. 2005. (Subseries LNAI).

Vol. 3486: T. Helleseth, D. Sarwate, H.-Y. Song, K. Yang (Eds.), Sequences and Their Applications - SETA 2004. XII, 451 pages. 2005.

Vol. 3483: O. Gervasi, M.L. Gavrilova, V. Kumar, A. Laganà, H.P. Lee, Y. Mun, D. Taniar, C.J.K. Tan (Eds.), Computational Science and Its Applications – ICCSA 2005, Part IV. XXVII, 1362 pages. 2005.

Vol. 3482: O. Gervasi, M.L. Gavrilova, V. Kumar, A. Laganà, H.P. Lee, Y. Mun, D. Taniar, C.J.K. Tan (Eds.), Computational Science and Its Applications – ICCSA 2005, Part III. LXVI, 1340 pages. 2005.

Vol. 3481: O. Gervasi, M.L. Gavrilova, V. Kumar, A. Laganà, H.P. Lee, Y. Mun, D. Taniar, C.J.K. Tan (Eds.), Computational Science and Its Applications – ICCSA 2005, Part II. LXIV, 1316 pages. 2005.

Vol. 3480: O. Gervasi, M.L. Gavrilova, V. Kumar, A. Laganà, H.P. Lee, Y. Mun, D. Taniar, C.J.K. Tan (Eds.), Computational Science and Its Applications – ICCSA 2005, Part I. LXV, 1234 pages. 2005.

Vol. 3479: T. Strang, C. Linnhoff-Popien (Eds.), Location- and Context-Awareness. XII, 378 pages. 2005.

Vol. 3478: C. Jermann, A. Neumaier, D. Sam (Eds.), Global Optimization and Constraint Satisfaction. XIII, 193 pages. 2005.

Vol. 3477: P. Herrmann, V. Issarny, S. Shiu (Eds.), Trust Management. XII, 426 pages. 2005.

Vol. 3476: J. Leite, A. Omicini, P. Torroni, P. Yolum (Eds.), Declarative Agent Languages and Technologies. XII, 289 pages. 2005.

Vol. 3475: N. Guelfi (Ed.), Rapid Integration of Software Engineering Techniques. X, 145 pages. 2005.

Vol. 3474: C. Grelck, F. Huch, G.J. Michaelson, P. Trinder (Eds.), Implementation and Application of Functional Languages. X, 227 pages. 2005.

Vol. 3468: H.W. Gellersen, R. Want, A. Schmidt (Eds.), Pervasive Computing. XIII, 347 pages. 2005.

Vol. 3467: J. Giesl (Ed.), Term Rewriting and Applications. XIII, 517 pages. 2005.

Vol. 3466: S. Leue, T.J. Systä (Eds.), Scenarios: Models, Transformations and Tools. XII, 279 pages. 2005.

Vol. 3465: M. Bernardo, A. Bogliolo (Eds.), Formal Methods for Mobile Computing. VII, 271 pages. 2005.

Vol. 3464: S.A. Brueckner, G.D.M. Serugendo, A. Karageorgos, R. Nagpal (Eds.), Engineering Self-Organising Systems. XIII, 299 pages. 2005. (Subseries LNAI).

Vol. 3463: M. Dal Cin, M. Kaâniche, A. Pataricza (Eds.), Dependable Computing - EDCC 2005. XVI, 472 pages. 2005.

Vol. 3462: R. Boutaba, K.C. Almeroth, R. Puigjaner, S. Shen, J.P. Black (Eds.), NETWORKING 2005. XXX, 1483 pages. 2005.

Vol. 3461: P. Urzyczyn (Ed.), Typed Lambda Calculi and Applications. XI, 433 pages. 2005.

Vol. 3460: Ö. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer, S. Leonardi, A. van Moorsel, M. van Steen (Eds.), Self-star Properties in Complex Information Systems. IX, 447 pages. 2005.

Vol. 3459: R. Kimmel, N.A. Sochen, J. Weickert (Eds.), Scale Space and PDE Methods in Computer Vision. XI, 634 pages. 2005.

Table of Contents

Applications and Tools (Short Papers)

Model Driven Architecture (Short Papers)

Test and Validation

Code Generation

ULF-Ware – An Open Framework for Integrated Tools for ITU-T Languages

Joachim Fischer, Andreas Kunert, Michael Piefel, and Markus Scheidgen

Humboldt-Universität zu Berlin, Institut für Informatik,
Unter den Linden 6, 10099 Berlin, Germany
{fischer, kunert, piefel, scheidge}@informatik.hu-berlin.de

Abstract. Model driven engineering is a popular attempt to deal with the complexity of modern software systems. For the telecommunication sector a model driven approach means that you have to handle several ITU-T modelling languages in a single process to cover all aspects of telecommunication system development. Unfortunately, this is a difficult task, because the ITU-T languages are hard to use together. That is why the ITU-T started the Unified Language Family (ULF) initiative with the goal to unify the ITU-T language definitions and allow an easier alignment and integrated use of these languages.

We present a tooling framework for those ULF languages: ULF-ware. Our framework uses metamodelling and a shared use of common language concepts for a tight language integration. Around these language models it incorporates a set of tools to cover the various responsibilities of development environments such as program parsing, model checking, model transformation and code generation.

This paper shows work in progress. We demonstrate our ideas on a tool chain for a subset of SDL. But the overall goal is an open framework that is extendable with other languages, even beyond ULF, and with tools for other software engineering tasks such as model simulation or software deployment.

1 Introduction

Over the past decades the ITU-T developed a series of modelling languages; each to cover a special aspect of telecommunication system specification. These languages are called: *eODL* – used for high-level component description; *SDL* [1] and *MSC* – to define different approaches to behaviour description and modelling; *ASN.1* – to define data; and *TTCN* – to write test cases. So there is virtually a modelling technique for every need, but in reality this is meaningless if these languages cannot be used together.

Different methodologies used in language development and definition make it hard to align and relate these languages with each other, so that integration is not trivial – it is barely possible. Of course, this is not news, and various calls for integration have been made. The ITU-T proposed the idea of a *Unified Language*

A. Prinz, R. Reed, and J. Reed (Eds.): SDL 2005, LNCS 3530, pp. 1–15, 2005.

Family (ULF): a consistent, uniform foundation for all ITU-T languages, but what is the method of choice to produce this foundation?

Two rivals have emerged: the field-tested and well-founded context free grammars, versus the new (incarnation of an old idea) metamodelling that has proven itself by building the base for ULF's "antagonist" UML [2]. Omitting all political arguments, metamodelling seems the more promising, and therefore scientifically more interesting approach. This paper proposes an approach to language tool development that uses metamodelling that is named after its overall goal: ULF-Ware.

ULF-Ware concerns utilizing metamodelling's potential for: faster tool and language development cycles, reuse of language concepts, and language integration. The metamodelling method gives us two advantages: first, you can define the abstract syntax of many languages as a combined model; second, it allows the various tools that are needed to use a language properly to be developed separately.

The first point is founded on the independence from concrete notation and metamodelling's ability to form reusable object-oriented structures. It is the independence from concrete syntax that allows modelling of language concepts abstractly, independent from syntax details. It is object-orientation that allows reuse and specialization of the common, abstract concepts in concrete languages. In these ways the separate ULF languages can become the ULFamily.

The second point is based on metamodels that are data models specifying (and can even standardizing) all the interfaces needed between different language tools. The use of abstract, coherent concepts in the metamodel further loosens the coupling between concept implementations and enables reuse.

With ULF-Ware we propose a metamodel-based, extendable framework for the implementation of ULF in the spirit of the OMG's MDA [3]. Section 2 explains the overall idea and philosophy of ULF-Ware, and we introduce a first piece of ULF-Ware that we are implementing right now – an SDL/UML compiler tool chain. In section 3 we present our current work in progress; this section gives interesting insights into the various aspects of metamodel-based compiler construction. The concluding section discusses the future of ULF.

2 ULF-Ware

The label ULF-Ware denotes all our tools around the Unified Language Family. We constituted all ULF-Ware components around a conceptual model based architecture: the ULF-Ware philosophy. We have begun to implement combined SDL and UML compiler tools. These first ULF-Ware pieces have to prove the applicability of the ULF-Ware philosophy.

2.1 Philosophy Behind ULF-Ware

ULF-Ware uses a centralized architecture; it has orbits placed around a core. Figure 1 gives an overview on ULF-Ware.

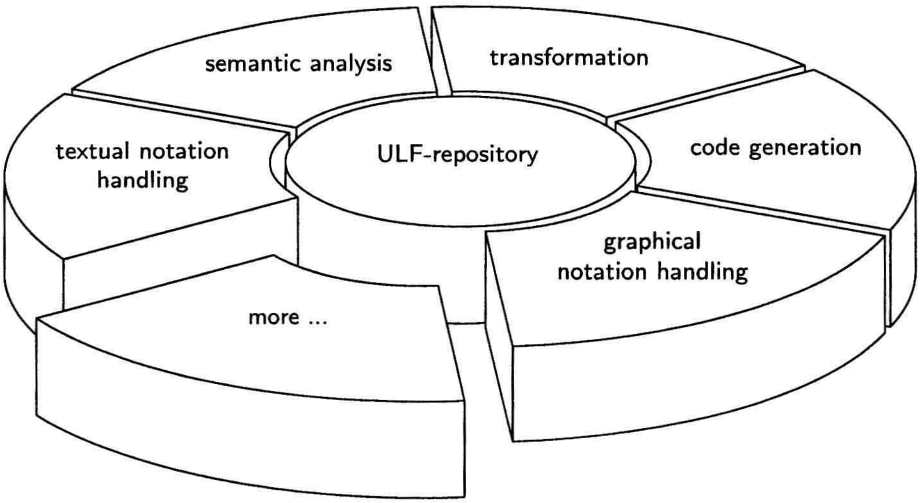

Fig. 1. An Overview of the ULF-Ware

The core's responsibility is to handle all models: these are M1-models such as specifications and programs as well as M2-metamodels, the metamodels. It is a model-centered architecture. It is responsible for model storage and representation; the core also facilitates functionality for model exchange, and therefore connects the various language tools. It offers all the functionality needed to integrate the orbiting tools. It can be understood as a provider for all common functionality and shared data that the language tools need. We realize the core by using a MOF-compliant repository, where MOF (*Model Object Facility*) is the standardized metamodelling architecture of the OMG [4].

The orbits around the core use, import and modify the models in the core; they use the core's operational interface. The distinct orbits act independently of each other, except that their behaviour is based on the shared data provided by the core repository. Because all orbits are independent of each other, the architecture is not fixed to the initial given orbits and is easily extensible.

If we step back from this structural viewpoint and look at core and orbits in terms of languages, we see the core handles abstract syntax and the orbits handle semantics, where the concrete notations are considered a part of semantics.

The core handles the metamodels and provides a repository for actual language instances (specifications or programs). The language instances are realized by the *extent* concept. An extent is a conceptual space, where the lifecycle of model elements takes place. An extent is automatically generated from the metamodel for which it provides an instance.

The orbits add meaning to the abstract syntax stored in the core. Examples for those semantics are: static semantics – the check of models for static

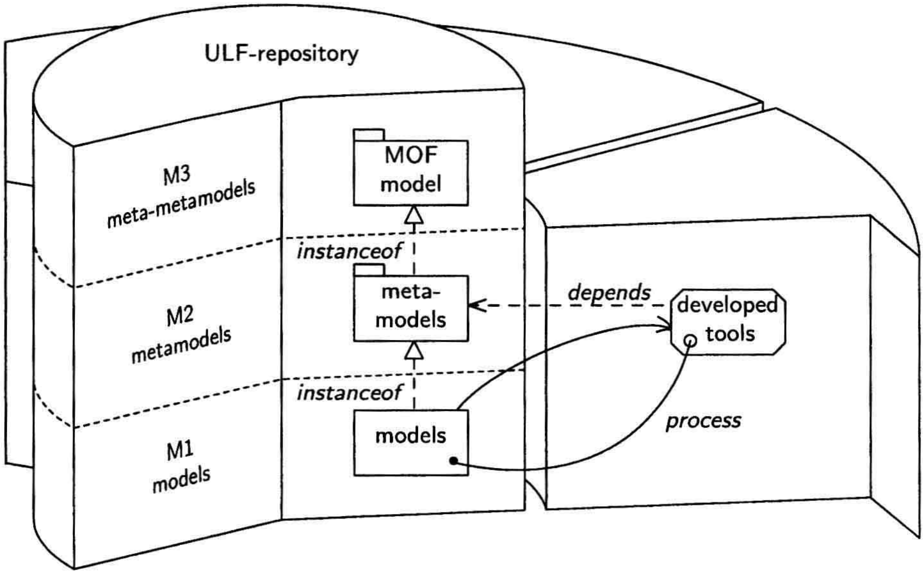

Fig. 2. The philosophy of the distinct ULF-Ware orbits

correctness; model transformation – as a possible representation of dynamic semantics; code generation – the question of how a model can be represented by implementation code; textual and graphical representations – which relate graphical or textual tokens to abstract model entities. There are many other possible semantics, such as simulation or deployment.

Figure 2 provides a closer view of the ULF-Ware philosophy. The core is a realization of a 4-layered metamodelling architecture: The models of every layer are described by a more abstract (*more 'meta'*) model in the layer above. An example: the M1-layer represents SDL specifications; the M2-layer contains the language description, the SDL metamodel; and the third layer defines the language used to write metamodels: for a MOF-repository this is the MOF-model.[1]

The semantics are realized by tools. In the first development state these might be hand-written tools that depend on the languages that they are written for. This dependency shows itself in the fact that the tools rely on the metamodel, they rely on the syntax. Tools can express semantics by modifying, creating and using models in the repository – they process models.

An SDL model checker, for example, is a tool that implements rules such as: *every agent of process kind must only contain other processes*. Such a rule depends on the SDL metamodel, because uses the metamodel elements *agent,*

[1] Please refer to [5] for an introduction into metamodelling architectures.