LNAI 3392

Dietmar Seipel
Michael Hanus
Ulrich Geske
Oskar Bartenstein (Eds.)

Applications of Declarative Programming and Knowledge Management

15th International Conference on Applications of Declarative Programming and Knowledge Management, INAP 2004 and 18th Workshop on Logic Programming, WLP 2004 Potsdam, Germany, March 2004, Revised Selected Papers

Springer

Dietmar Seipel Michael Hanus
Ulrich Geske Oskar Bartenstein (Eds.)

Applications of Declarative Programming and Knowledge Management

15th International Conference
on Applications of Declarative Programming
and Knowledge Management, INAP 2004
and 18th Workshop on Logic Programming, WLP 2004
Potsdam, Germany, March 4-6, 2004
Revised Selected Papers

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Dietmar Seipel
Universität Würzburg, Institut für Informatik
Am Hubland, 97074 Würzburg, Germany
E-mail: seipel@informatik.uni-wuerzburg.de

Michael Hanus
Christian-Albrechts-Universität Kiel
Institut für Informatik und Praktische Mathematik
Olshausenstr. 40, 24098 Kiel, Germany
E-mail: mh@informatik.uni-kiel.de

Ulrich Geske
Fraunhofer FIRST
Kekulestr. 7, 12489 Berlin, Germany
E-mail: geske@first.fhg.de

Oskar Bartenstein
IF Computer Japan Limited
5-28-2 Sendagi, Bunkyo-ku, Tokyo, 113-0022, Japan
E-mail: oskar@ifcomputer.co.jp

Library of Congress Control Number: 2005923659

CR Subject Classification (1998): I.2.4, I.2, D.1.6

ISSN 0302-9743
ISBN-10 3-540-25560-5 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-25560-4 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: 11415763 06/3142 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 3392

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Preface

This volume contains a selection of papers presented at the 15th International Conference on Applications of Declarative Programming and Knowledge Management, INAP 2004, and the 18th Workshop on Logic Programming, WLP 2004, which were held jointly in Potsdam, Germany, from March 4th to 6th, 2004.

Declarative programming is an advanced paradigm for the modeling and solving of complex problems. This specification method has become more and more attractive in recent years, for example, in the domains of databases, for the processing of natural language, for the modeling and processing of combinatorial problems, and for establishing knowledge-based systems for the Web.

The INAP conferences provide a forum for intensive discussions of applications of important technologies around logic programming, constraint problem solving, and closely related advanced software. They comprehensively cover the impact of programmable logic solvers in the Internet society, its underlying technologies, and leading-edge applications in industry, commerce, government, and social services.

The Workshops on Logic Programming are the annual meeting of the Society for Logic Programming (GLP e.V.). They bring together researchers interested in logic programming, constraint programming, and related areas like databases and artificial intelligence. Previous workshops have been held in Germany, Austria, and Switzerland.

The topics of the selected papers of this year's joint conference concentrate on three currently important fields: knowledge management and decision support, constraint programming and constraint solving, and declarative programming and Web-based systems.

During the last couple of years a lot of research has been conducted on the use of declarative programming for the management of knowledge-based systems and for decision support. Reasoning about knowledge wrapped in rules, databases, or the Web allows us to explore interesting hidden knowledge. Declarative techniques for the transformation, deduction, induction, visualization, or querying of knowledge, or data mining techniques for exploring knowledge have the advantage of high transparency and better maintainability compared to procedural approaches.

The problem when using knowledge to find solutions for large industrial tasks is that these problems have an exponential complexity, which normally prohibits the fast generation of exact solutions. One method that has made substantial progress over the last few years is the constraint programming paradigm. The declarative nature of this paradigm offers significant advantages for software engineering both in the implementation and in the maintenance phases. Different interesting aspects are in discussion: how can this paradigm be improved or combined with known, classical methods; how can practical problems be modelled as constraint problems; and what are the experiences of applications in really large industrial planning and simulation tasks?

An emerging topic in knowledge management is the use of the World Wide Web to distribute, store, and use knowledge. This spans vision, technology, and the application of non-monolithic cooperating Web-based systems. With respect to declarative programming, representation languages, transformation, and search procedures are of interest, and they are easily adaptable to the fast-changing content and structure of the Web, for example, in W3C Web services and queries. Other aspects are commercial Web-based consulting or the use of the Web as a platform for concurrent engineering or program development for effective distributed collaborative design.

The two conferences INAP 2004 and WLP 2004 were jointly organized at the University of Potsdam, Germany by the following institutions: the Society for Logic Programming (GLP e.V.), the Hasso Plattner Institute for Software Systems Engineering (HPI), and the Fraunhofer Institute for Computer Architecture and Software Technology (FhG FIRST). We would like to thank all authors who submitted papers and all conference participants for the fruitful discussions. We are grateful to the members of the Program Committee and the external referees for their timely expertise in carefully reviewing the papers, and we would like to express our thanks to the Hasso Plattner Institute for hosting the conference at the modern campus in the traditional atmosphere of Potsdam.

December 2004

Dietmar Seipel, Michael Hanus,
Ulrich Geske, Oskar Bartenstein

Organization

Program Chair

Dietmar Seipel	University of Würzburg, Germany

Organization

Ulrich Geske	Fraunhofer FIRST, Berlin, Germany
Thomas Linke	University of Potsdam, Germany
Wolfgang Severin	University of Potsdam, Germany
Armin Wolf	Fraunhofer FIRST, Berlin, Germany

Program Committee of INAP

Sergio A. Alvarez	Boston College, USA
Roman Barták	Charles University, Prague, Czech Republic
Oskar Bartenstein	IF Computer Japan, Japan
Joachim Baumeister	University of Würzburg, Germany
James P. Delgrande	Simon Fraser University, Canada
Robin Drogemuller	CSIRO, Australia
Shuichi Fukuda	Tokyo Metropolitan Institute of Technology, Japan
Rita Gavriloaie	Learning Lab Lower Saxony, Hannover, Germany
Nicola Henze	University of Hannover, Germany
Ulrich Geske	Fraunhofer FIRST, Germany
Geun-Sik Jo	Inha University, Korea
Thomas Linke	University of Potsdam, Germany
Steve Prestwich	4C/UCC, Ireland
Georg Ringwelski	4C/UCC, Ireland
Carolina Ruiz	Worcester Polytechnic Institute, USA
Torsten Schaub	University of Potsdam, Germany
Dietmar Seipel	University of Würzburg, Germany (Chair)
Paul Tarau	Binnet Corporation, USA
Armin Wolf	Fraunhofer FIRST, Germany
Akihiro Yamamoto	Kyoto University, Japan
Osamu Yoshie	Waseda University, Japan

Program Committee of WLP

Slim Abdennadher	German University Cairo, Egypt
Christoph Beierle	FernUniversität in Hagen, Germany

François Bry	University of München, Germany
Jürgen Dix	Technical University of Clausthal, Germany
Uwe Egly	Technical University of Vienna, Austria
Thomas Eiter	Technical University of Vienna, Austria
Burkhard Freitag	University of Passau, Germany
Thom Frühwirth	University of Ulm, Germany
Norbert E. Fuchs	University of Zürich, Switzerland
Ulrich Geske	Fraunhofer FIRST, Berlin, Germany
Michael Hanus	Christian Albrechts University Kiel, Germany (Chair)
Petra Hofstedt	Technical University of Berlin, Germany
Steffen Hölldobler	Technical University of Dresden, Germany
Ulrich Neumerkel	Technical University of Vienna, Austria
Dietmar Seipel	University of Würzburg, Germany
Armin Wolf	Fraunhofer FIRST, Berlin, Germany

External Referees for INAP and WLP

Christian Anger
Matthias Beck
Ole Boysen
Mona Gharib
Marbod Hopfner
Horst Reichel
Armagan Tarim
Stefan Woltran
Martin Atzmueller
Bernd Braßel
Tom Carchrae
Bernd Heumesser
Kathrin Konczak
Maged F. El Sayed
Manfred Widera

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 3452: F. Baader, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning. XI, 562 pages. 2005.

Vol. 3419: B. Faltings, A. Petcu, F. Fages, F. Rossi (Eds.), Constraint Satisfaction and Constraint Logic Programming. X, 217 pages. 2005.

Vol. 3416: M. Böhlen, J. Gamper, W. Polasek, M.A. Wimmer (Eds.), E-Government: Towards Electronic Democracy. XIII, 311 pages. 2005.

Vol. 3415: P. Davidsson, B. Logan, K. Takadama (Eds.), Multi-Agent and Multi-Agent-Based Simulation. X, 265 pages. 2005.

Vol. 3403: B. Ganter, R. Godin (Eds.), Formal Concept Analysis. XI, 419 pages. 2005.

Vol. 3398: D.-K. Baik (Ed.), Systems Modeling and Simulation: Theory and Applications. XIV, 733 pages. 2005.

Vol. 3397: T.G. Kim (Ed.), Artificial Intelligence and Simulation. XV, 711 pages. 2005.

Vol. 3396: R.M. van Eijk, M.-P. Huget, F. Dignum (Eds.), Agent Communication. X, 261 pages. 2005.

Vol. 3394: D. Kudenko, D. Kazakov, E. Alonso (Eds.), Adaptive Agents and Multi-Agent Systems III. VIII, 313 pages. 2005.

Vol. 3392: D. Seipel, M. Hanus, U. Geske, O. Bartenstein (Eds.), Applications of Declarative Programming and Knowledge Management. X, 309 pages. 2005.

Vol. 3374: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems. X, 279 pages. 2005.

Vol. 3371: M.W. Barley, N. Kasabov (Eds.), Intelligent Agents and Multi-Agent Systems. X, 329 pages. 2005.

Vol. 3369: V.R. Benjamins, P. Casanovas, J. Breuker, A. Gangemi (Eds.), Law and the Semantic Web. XII, 249 pages. 2005.

Vol. 3366: I. Rahwan, P. Moraitis, C. Reed (Eds.), Argumentation in Multi-Agent Systems. XII, 263 pages. 2005.

Vol. 3359: G. Grieser, Y. Tanaka (Eds.), Intuitive Human Interfaces for Organizing and Accessing Intellectual Assets. XIV, 257 pages. 2005.

Vol. 3346: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2005.

Vol. 3345: Y. Cai (Ed.), Ambient Intelligence for Scientific Discovery. XII, 311 pages. 2005.

Vol. 3343: C. Freksa, M. Knauff, B. Krieg-Brückner, B. Nebel, T. Barkowsky (Eds.), Spatial Cognition IV. Reasoning, Action, and Interaction. XIII, 519 pages. 2005.

Vol. 3339: G.I. Webb, X. Yu (Eds.), AI 2004: Advances in Artificial Intelligence. XXII, 1272 pages. 2004.

Vol. 3336: D. Karagiannis, U. Reimer (Eds.), Practical Aspects of Knowledge Management. X, 523 pages. 2004.

Vol. 3327: Y. Shi, W. Xu, Z. Chen (Eds.), Data Mining and Knowledge Management. XIII, 263 pages. 2005.

Vol. 3315: C. Lemaître, C.A. Reyes, J.A. González (Eds.), Advances in Artificial Intelligence – IBERAMIA 2004. XX, 987 pages. 2004.

Vol. 3303: J.A. López, E. Benfenati, W. Dubitzky (Eds.), Knowledge Exploration in Life Science Informatics. X, 249 pages. 2004.

Vol. 3276: D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.), RoboCup 2004: Robot Soccer World Cup VIII. XVIII, 678 pages. 2005.

Vol. 3275: P. Perner (Ed.), Advances in Data Mining. VIII, 173 pages. 2004.

Vol. 3265: R.E. Frederking, K.B. Taylor (Eds.), Machine Translation: From Real Users to Research. XI, 392 pages. 2004.

Vol. 3264: G. Paliouras, Y. Sakakibara (Eds.), Grammatical Inference: Algorithms and Applications. XI, 291 pages. 2004.

Vol. 3259: J. Dix, J. Leite (Eds.), Computational Logic in Multi-Agent Systems. XII, 251 pages. 2004.

Vol. 3257: E. Motta, N.R. Shadbolt, A. Stutt, N. Gibbins (Eds.), Engineering Knowledge in the Age of the Semantic Web. XVII, 517 pages. 2004.

Vol. 3249: B. Buchberger, J.A. Campbell (Eds.), Artificial Intelligence and Symbolic Computation. X, 285 pages. 2004.

Vol. 3248: K.-Y. Su, J. Tsujii, J.-H. Lee, O.Y. Kwong (Eds.), Natural Language Processing – IJCNLP 2004. XVIII, 817 pages. 2005.

Vol. 3245: E. Suzuki, S. Arikawa (Eds.), Discovery Science. XIV, 430 pages. 2004.

Vol. 3244: S. Ben-David, J. Case, A. Maruoka (Eds.), Algorithmic Learning Theory. XIV, 505 pages. 2004.

Vol. 3238: S. Biundo, T. Frühwirth, G. Palm (Eds.), KI 2004: Advances in Artificial Intelligence. XI, 467 pages. 2004.

Vol. 3230: J.L. Vicedo, P. Martínez-Barco, R. Muñoz, M. Saiz Noeda (Eds.), Advances in Natural Language Processing. XII, 488 pages. 2004.

Vol. 3229: J.J. Alferes, J. Leite (Eds.), Logics in Artificial Intelligence. XIV, 744 pages. 2004.

Vol. 3228: M.G. Hinchey, J.L. Rash, W.F. Truszkowski, C.A. Rouff (Eds.), Formal Approaches to Agent-Based Systems. VIII, 290 pages. 2004.

Vol. 3215: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. LVII, 906 pages. 2004.

Vol. 3214: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. LVIII, 1302 pages. 2004.

Vol. 3213: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LVIII, 1280 pages. 2004.

Vol. 3209: B. Berendt, A. Hotho, D. Mladenic, M. van Someren, M. Spiliopoulou, G. Stumme (Eds.), Web Mining: From Web to Semantic Web. IX, 201 pages. 2004.

Vol. 3206: P. Sojka, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue. XIII, 667 pages. 2004.

Vol. 3202: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Knowledge Discovery in Databases: PKDD 2004. XIX, 560 pages. 2004.

Vol. 3201: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Machine Learning: ECML 2004. XVIII, 580 pages. 2004.

Vol. 3194: R. Camacho, R. King, A. Srinivasan (Eds.), Inductive Logic Programming. XI, 361 pages. 2004.

Vol. 3192: C. Bussler, D. Fensel (Eds.), Artificial Intelligence: Methodology, Systems, and Applications. XIII, 522 pages. 2004.

Vol. 3191: M. Klusch, S. Ossowski, V. Kashyap, R. Unland (Eds.), Cooperative Information Agents VIII. XI, 303 pages. 2004.

Vol. 3187: G. Lindemann, J. Denzinger, I.J. Timm, R. Unland (Eds.), Multiagent System Technologies. XIII, 341 pages. 2004.

Vol. 3176: O. Bousquet, U. von Luxburg, G. Rätsch (Eds.), Advanced Lectures on Machine Learning. IX, 241 pages. 2004.

Vol. 3171: A.L.C. Bazzan, S. Labidi (Eds.), Advances in Artificial Intelligence – SBIA 2004. XVII, 548 pages. 2004.

Vol. 3159: U. Visser, Intelligent Information Integration for the Semantic Web. XIV, 150 pages. 2004.

Vol. 3157: C. Zhang, H. W. Guesgen, W.K. Yeap (Eds.), PRICAI 2004: Trends in Artificial Intelligence. XX, 1023 pages. 2004.

Vol. 3155: P. Funk, P.A. González Calero (Eds.), Advances in Case-Based Reasoning. XIII, 822 pages. 2004.

Vol. 3139: F. Iida, R. Pfeifer, L. Steels, Y. Kuniyoshi (Eds.), Embodied Artificial Intelligence. IX, 331 pages. 2004.

Vol. 3131: V. Torra, Y. Narukawa (Eds.), Modeling Decisions for Artificial Intelligence. XI, 327 pages. 2004.

Vol. 3127: K.E. Wolff, H.D. Pfeiffer, H.S. Delugach (Eds.), Conceptual Structures at Work. XI, 403 pages. 2004.

Vol. 3123: A. Belz, R. Evans, P. Piwek (Eds.), Natural Language Generation. X, 219 pages. 2004.

Vol. 3120: J. Shawe-Taylor, Y. Singer (Eds.), Learning Theory. X, 648 pages. 2004.

Vol. 3097: D. Basin, M. Rusinowitch (Eds.), Automated Reasoning. XII, 493 pages. 2004.

Vol. 3071: A. Omicini, P. Petta, J. Pitt (Eds.), Engineering Societies in the Agents World. XIII, 409 pages. 2004.

Vol. 3070: L. Rutkowski, J. Siekmann, R. Tadeusiewicz, L.A. Zadeh (Eds.), Artificial Intelligence and Soft Computing - ICAISC 2004. XXV, 1208 pages. 2004.

Vol. 3068: E. André, L. Dybkjær, W. Minker, P. Heisterkamp (Eds.), Affective Dialogue Systems. XII, 324 pages. 2004.

Vol. 3067: M. Dastani, J. Dix, A. El Fallah-Seghrouchni (Eds.), Programming Multi-Agent Systems. X, 221 pages. 2004.

Vol. 3066: S. Tsumoto, R. Słowiński, J. Komorowski, J.W. Grzymała-Busse (Eds.), Rough Sets and Current Trends in Computing. XX, 853 pages. 2004.

Vol. 3065: A. Lomuscio, D. Nute (Eds.), Deontic Logic in Computer Science. X, 275 pages. 2004.

Vol. 3060: A.Y. Tawfik, S.D. Goodwin (Eds.), Advances in Artificial Intelligence. XIII, 582 pages. 2004.

Vol. 3056: H. Dai, R. Srikant, C. Zhang (Eds.), Advances in Knowledge Discovery and Data Mining. XIX, 713 pages. 2004.

Vol. 3055: H. Christiansen, M.-S. Hacid, T. Andreasen, H.L. Larsen (Eds.), Flexible Query Answering Systems. X, 500 pages. 2004.

Vol. 3048: P. Faratin, D.C. Parkes, J.A. Rodríguez-Aguilar, W.E. Walsh (Eds.), Agent-Mediated Electronic Commerce V. XI, 155 pages. 2004.

Vol. 3040: R. Conejo, M. Urretavizcaya, J.-L. Pérez-de-la-Cruz (Eds.), Current Topics in Artificial Intelligence. XIV, 689 pages. 2004.

Vol. 3035: M.A. Wimmer (Ed.), Knowledge Management in Electronic Government. XII, 326 pages. 2004.

Vol. 3034: J. Favela, E. Menasalvas, E. Chávez (Eds.), Advances in Web Intelligence. XIII, 227 pages. 2004.

Vol. 3030: P. Giorgini, B. Henderson-Sellers, M. Winikoff (Eds.), Agent-Oriented Information Systems. XIV, 207 pages. 2004.

Vol. 3029: B. Orchard, C. Yang, M. Ali (Eds.), Innovations in Applied Artificial Intelligence. XXI, 1272 pages. 2004.

Vol. 3025: G.A. Vouros, T. Panayiotopoulos (Eds.), Methods and Applications of Artificial Intelligence. XV, 546 pages. 2004.

Vol. 3020: D. Polani, B. Browning, A. Bonarini, K. Yoshida (Eds.), RoboCup 2003: Robot Soccer World Cup VII. XVI, 767 pages. 2004.

Vol. 3012: K. Kurumatani, S.-H. Chen, A. Ohuchi (Eds.), Multi-Agents for Mass User Support. X, 217 pages. 2004.

Vol. 3010: K.R. Apt, F. Fages, F. Rossi, P. Szeredi, J. Váncza (Eds.), Recent Advances in Constraints. VIII, 285 pages. 2004.

Vol. 2990: J. Leite, A. Omicini, L. Sterling, P. Torroni (Eds.), Declarative Agent Languages and Technologies. XII, 281 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004.

Vol. 2977: G. Di Marzo Serugendo, A. Karageorgos, O.F. Rana, F. Zambonelli (Eds.), Engineering Self-Organising Systems. X, 299 pages. 2004.

Vol. 2972: R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (Eds.), MICAI 2004: Advances in Artificial Intelligence. XVII, 923 pages. 2004.

Table of Contents

Knowledge Management and Decision Support

Optimizing the Evaluation of XPath Using Description Logics
Peter Baumgartner, Ulrich Furbach, Margret Gross-Hardt, Thomas Kleemann 1

Declaratively Querying and Visualizing Knowledge Bases in XML
Dietmar Seipel, Joachim Baumeister, Marbod Hopfner 16

SQL-Based Frequent Pattern Mining with FP-Growth
Xuequn Shang, Kai-Uwe Sattler, Ingolf Geist 32

Incremental Learning of Transfer Rules for Customized Machine Translation
Werner Winiwarter 47

Quality Measures and Semi-automatic Mining of Diagnostic Rule Bases
Martin Atzmueller, Joachim Baumeister, Frank Puppe 65

An Evaluation of a Rule-Based Language for Classification Queries
Dennis P. Groth 79

Deductive and Inductive Reasoning on Spatio-Temporal Data
Mirco Nanni, Alessandra Raffaetà, Chiara Renso, Franco Turini 98

Mining Semantic Structures in Movies
Kimiaki Shirahama, Yuya Matsuo, Kuniaki Uehara 116

Solving Alternating Boolean Equation Systems in Answer Set Programming
Misa Keinänen, Ilkka Niemelä 134

Constraint Programming and Constraint Solving

Effective Modeling with Constraints
Roman Barták 149

A Local Search System for Solving Constraint Problems of Declarative Graph-Based Global Constraints
Markus Bohlin 166

Realising the Alternative Resources Constraint
Armin Wolf, Hans Schlenker 185

Integrating Time Constraints into Constraint-Based Configuration Models
Ulrich John, Ulrich Geske 200

Distributed Constraint-Based Railway Simulation
Hans Schlenker 215

Declarative Programming and Web-Based Systems

Concurrent Engineering to Wisdom Engineering
Shuichi Fukuda 227

Web Services Based on PROLOG and XML
Bernd D. Heumesser, Andreas Ludwig, Dietmar Seipel 245

A Contribution to the Semantics of Xcerpt, a Web Query and Transformation Language
François Bry, Sebastian Schaffert, Andreas Schroeder 258

DialogEngines – Dialog Agents for Web-Based Self Service Consulting
Oskar Bartenstein 269

Towards Ubiquitous Maintenance – Defining Invocation of Plant Maintenance Agents in Real Workspace by Spatial Programming
Hiroki Takahashi, Osamu Yoshie 278

A Pragmatic Approach to Pre-testing Prolog Programs
Christoph Beierle, Marija Kulaš, Manfred Widera 294

Author Index 309

Optimizing the Evaluation of XPath Using Description Logics

Peter Baumgartner[1,2], Ulrich Furbach[1], Margret Gross-Hardt[1], and Thomas Kleemann[1]

[1] Institut für Informatik, Universität Koblenz-Landau, D-56070 Koblenz, Germany
[2] Max-Planck-Institut für Informatik, D-66123 Saarbrücken
{peter, uli, margret, tomkl}@uni-koblenz.de

Abstract. The growing use of XML in commercial as well as non-commercial domains to transport information poses new challenges to concepts to access this information. Common ways to access parts of a document use XPath-expressions. We provide a transformation of DTDs into a knowledge base in Description Logic. We use reasoning capabilities grounded in description logics to decide if a given XPath can be satisfied by a document, and to guide the search of XML-Processors into possibly successful branches of the document, thus avoiding parts of the document that will not yield results. The extension towards object oriented subclassing schemes opens this approach towards OODB-queries. In contrast to other approaches we do not use any kind of graph representing the document structure, and no steps towards incorporation of the XML/OODB-processor itself will be taken.

Keywords: XML, XPath, Description Logics, automated reasoning, DTD, Schema.

1 Introduction

Within a short period of time XML has become a widely accepted standard for information interchange. Starting as a subset of SGML to transport structured text, the ease of understanding and using XML has promoted its use as an interchange format of rather large documents. This evolution has created the needs for a validation of documents against an according definition. Most common and basic validation is accomplished by XML-processors referring to a Document Type Definition (DTD). A DTD defines the structure of elements in the document, that references the DTD. We will detail the terms element and structure in the following chapters.

Beyond the need to validate data, several attempts have been made or are currently made to access parts of a document. One common idea in these attempts is the access of parts of a document following a path from its root to some subtrees. In general, these paths are not specified completely from the root to the subtree, but leave unspecified gaps to be filled by the XML query processor. Usually XML processors have to traverse the whole document tree to find instances of the specified parts of a path.

Based on DTD and XPath expressions we will provide a way to optimize this traversal. Actually, a XPath expression may be seen as a pattern that describes certain subtrees

D. Seipel et al. (Eds.): INAP/WLP 2004, LNAI 3392, pp. 1–15, 2005.

of the complete document. As already mentioned, current processors traverse the complete document in order to determine the appropriate subtrees. Instead, our goal is to follow only those parts that may result in desired data, but omit those paths that are guaranteed to fail. The idea is to exploit structural knowledge in order to determine the relevant parts of a document for a certain query. This allows to enhance other approaches based on indexing techniques in query processing. To this aim, we translate the DTD into a set of description logics (DL) formulae and the XPath query into a query in a logical representation. Questions about the satisfiability of a XPath in a document will be answered as well as queries of the starting element of subtrees, which may contain fillers for the path specification.

Recently, XML Schema [14] has evolved as a successor of DTDs. Basically, XML Schema addresses some shortcomings of DTDs; in particular XML Schema supports besides others user defined types and some aspects of object orientation. We will show the compatibility of our translation and reasoning with these sorts of object oriented extensions. This opens our approach towards the use of path completion techniques in object oriented databases as well as schema-based definitions of XML documents.

2 XML Documents

Starting as a specialisation of the Standard Generalized Markup Language (SGML) the eXtensible Markup Language (XML) was supposed to provide a better way of document markup than the widespread HyperText Markup Language (HTML). The *normative* definition of XML is available from the W3C [6]. In contrast to the fixed markup and its interpretation of HTML, the XML approach offers a standardized way to markup arbitrary documents. This may include redefined HTML documents but is not limited to this application.

A XML document consists of a prologue and an element, optionally followed by miscellaneous comments and processing instructions, that are not in the scope of this paper. An element is either an empty element or a sequence of a starting tag followed by content and an ending tag. Taken from [6]:

[1]	document	::=	prolog element Misc*
[39]	element	::=	EmptyElemTag | STag content Etag
[40]	STag	::=	'<' Name Attribute* '>'
[41]	Attribute	::=	Name '=' Attvalue
[42]	ETag	::=	'</' Name '>'
[43]	content	::=	CharData? ((element | Reference | CDSect | PI | Comment) CharData?)*
[44]	EmptyElemTag	::=	'<' Name Attribute* '/>'

Content by itself may again contain so called child-elements. Tags are identified by their names. We will use this name as the name of an element.

The W3C cares about character codings, white spaces and miscellaneous components. Because we are merely interested in the structure of the document we will omit these otherwise important details. The topmost element will be called root element. It spans almost all of the document, especially it contains all other elements. We expect all documents to be well-formed and valid, as explained in the following section.

Sample Document

According to the above mentioned productions and constraints a well-formed document may look like this:

```
<?xml version="1.0"?>
<!DOCTYPE university SYSTEM "university.dtd">
<university name="Universität Koblenz">
  <library>
    <book isbn="978123123">
      <author> </author>
      <title>  </title>
    </book>
    <book isbn="978234234">
      <author> </author>
      <title>  </title>
    </book>
  </library>
  <department name="cs"/>
  <department name="math"/>
  ...more descriptions...
</university>
```

The prologue specifies this document to be a XML document according to version 1.0. This is currently the only possible version. The second line specifies a document type definition. University is the root element of the document. This university element contains a library element, that contains several book elements, and several department elements. The department elements are empty elements. Empty elements are empty with respect to the content, but may contain attributes. The university, department and book elements contain attributes, i.e. name and isbn.

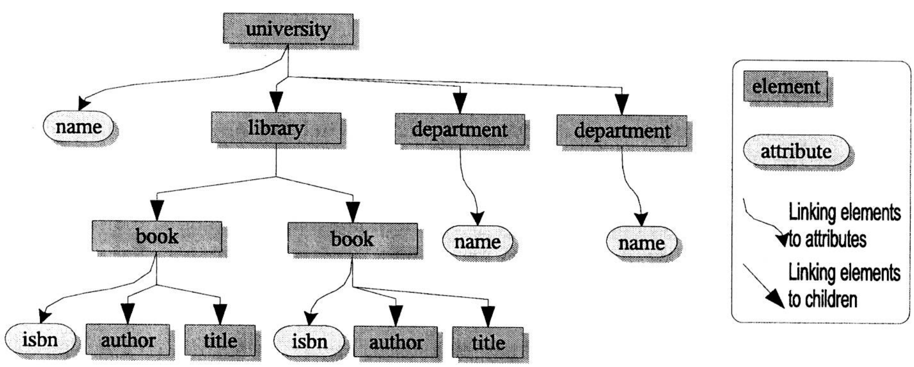

Fig. 1. Graphical Representation of Document

A XML document may be represented as a tree, see figure 1. The root element corresponds to the root of this tree. The elements are represented as nodes of the tree. An element is linked to its child elements and attributes. With respect to our query optimization task, we may omit the data of elements. Instead, we concentrate on the structure of documents. The following section introduces a common way to define possible structures.

3 Validation of Documents

Whenever XML is used to transport information between independent applications, which is most common in business-to-business communication, there is a need to validate the document structure. Validation of document structure means that the processor checks if the XML documents obey to the structure defined in the appropriate DTD. Validating XML processors offer standardized ways for this validation process. These processors use a Document Type Definition or a schema as a description of accepted elements, nesting of elements and type information. In the example document above (2^{nd} line) we already introduced the reference to an external DTD. This Document Type Declaration names the root element, university in this case. DTDs may as well be inline. The advantage of external DTDs is that they are stored centrally and only once. Beyond these differences both kinds of DTDs provide the same set of definitions. DTDs are already known to SGML [12] and HTML documents. They do not provide significant type information or any aspect of object orientation. To overcome these insufficiencies XML schema has been introduced by the W3C. A schema introduces some basic type information and a limited support of object oriented concepts. We will start with DTDs and demonstrate afterwards the opportunities of extended type information.

A DTD for the sample document above may look like this

```
<!ELEMENT university (library, department*, #PCDATA)>
<!ATTLIST university name CDATA #REQUIRED)>
<!ELEMENT library (book)+>
<!ELEMENT department EMPTY>
<!ATTLIST department name CDATA #REQUIRED)>
<!ELEMENT book (author+, title, abstract?)>
<!ATTLIST book isbn CDATA #REQUIRED)>
<!ELEMENT author #PCDATA>
<!ELEMENT title #PCDATA>
```

Focusing on the structure of a document, we will not use any information about the data of the document that is described. So #PCDATA, CDATA and so on will not be in the scope of this paper. Crucial to our target of optimization and completion of path expressions are the definitions of the child elements and attributes of elements. All elements mentioned in the definition of an element are child elements. The sequence operator ',' may be used to establish a sibling relation among child elements. In this example author, title and abstract are child elements of book. isbn is an attribute of book. author is a sibling of author and title and so on. Figure 2 provides an overview.

For the purpose of this paper we limit our presentation to the abbreviated syntax for XPath. In this syntax, one does not have the possibility to express sibling relationships

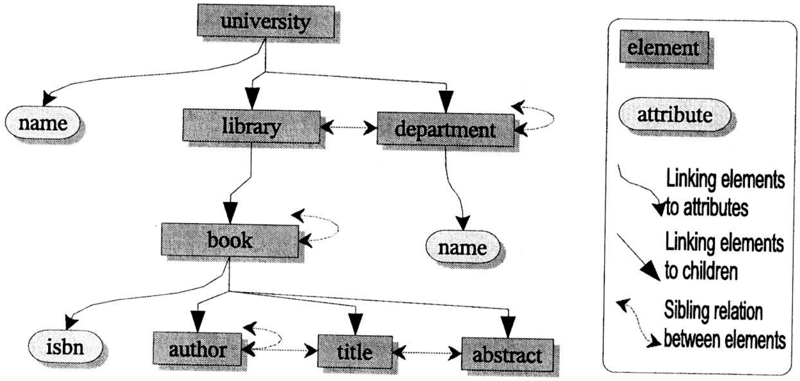

Fig. 2. Graphical Representation of DTD

directly. Therefore we omit these relations in the remaining of this paper although these relations between elements are covered by our approach.

Different from the tree like structure of the XML documents themselves, the description may contain cycles. A well known example of a cycle is the HTML table. The TD element is a child of an TR element, that is a child of the TABLE element. Because TABLE is part of arbitrary HTML content, TABLE is a possible child of TD. The reasoning capabilities used in our query optimization approach are robust against these cyclic definitions.

Even in our quite small example, DTD cycles can be found, e.g. the element author is a sibling of itself.

4 Picking the Parts

Common to almost all processing of documents is the addressing of parts of these. The basic idea of all addressing schemes is a path expression, that specifies the navigation through the document. These path expressions may be rooted or relative to an existing position in the document tree. Several notational variants have been developed. We will use the abbreviated XPath 2.0 notation, that is covered by a W3C-working-draft [15]. Path expressions following these recommendations are incorporated in XSLT, XQuery, CSS2 and other standards.

Path expressions are explained by the following rules taken from [15]:

```
        path ::= '//' relativePath | '/' relativePath | relativePath
relativePath ::= stepExpr ( '/' | '//' ) stepExpr
    stepExpr ::= '.' | '@' nameTest | '..' | nodeTest
```

The leading '/' and '//' construct a path starting at the document out of an relative path. '//' will expand to a path of zero or more steps. Step expressions access the current context node, its attributes by a preceding '@' the parent of an element, or perform node tests, ranging from simple element names to more complex expressions. Especially a wildcard '*' will match all child elements. A detailed description can be found in [15].