Equation-Based Object-Oriented Languages and Tools
Report on the Workshop EOOLT 2007 at ECOOP 2007

Peter Fritzson[1], David Broman[1], François Cellier[2],
and Christoph Nytsch-Geusen[3]

[1] Linköping University, Sweden
{davbr,petfr}@ida.liu.se
[2] ETH Zurich, Switzerland
fcellier@inf.ethz.ch
[3] Fraunhofer FIRST, Germany
christoph.nytsch@first.fraunhofer.de

Abstract. EOOLT'2007 was the first edition of the ECOOP-EOOLT workshop. The workshop is intended to bring researchers associated with different equation-based object-oriented (EOO) modeling languages and different application areas making use of such languages together. The aim of the workshop is to explore common grounds and derive software design principles that may make future EOO modeling languages more robust, more versatile, and more widely accepted among the various stakeholders. At EOOLT'2007, nineteen researchers with diverse backgrounds and needs came together to present and discuss fourteen different concept papers grouped into the four topic areas of integrated system modeling approaches; hybrid modeling and variable structure systems; modeling languages, specification, and language comparison; and tools and methods.

1 Objectives and Call for Papers

Computer aided modeling and simulation of complex systems, using components from multiple application domains, such as electrical, mechanical, hydraulic, control, etc., have in recent years witnessed a significant growth of interest. In the last decade, novel equation-based object-oriented (EOO) modeling languages, (e.g., Modelica, gPROMS, and VHDL-AMS) based on acausal modeling using equations have appeared. Using such languages, it has become possible to model complex systems covering multiple application domains at a high level of abstraction through reusable model components.

The interest in EOO languages and tools is rapidly growing in the industry because of their increasing importance in modeling, simulation, and specification of complex systems. There exist several different EOO language communities today that grew out of different application areas (multi-body system dynamics, electronic circuit simulation, chemical process engineering). The members of these disparate communities rarely talk to each other in spite of the similarities of their modeling and simulation needs.

M. Cebulla (Ed.): ECOOP 2007 Workshop Reader, LNCS 4906, pp. 27–39, 2008.

The workshop is concerned with, but not limited to, the following themes:

- Acausality and its role in model reusability.
- Component systems for EOO languages.
- Database lookup and knowledge invocation.
- Discrete-event and hybrid modeling using EOO languages.
- Embedded systems.
- EOO language constructs in support of simulation, optimization, diagnostics, and system identification.
- EOO mathematical modeling vs. UML modeling.
- Equation-based languages supporting DAEs and/or PDEs.
- Formal semantics of EOO related languages.
- Multi-resolution / multi-scale modeling using EOO languages.
- Numerical coupling of EOO simulators and other simulation tools.
- Parallel execution of EOO models.
- Performance issues.
- Programming / modeling environments.
- Real-time simulation using EOO languages.
- Reflection and meta-programming.
- Reuse of models in EOO languages.
- Table lookup and interpolation.
- Type systems and early static checking.
- Verification.

The EOOLT workshop series aims at bringing these different communities together to discuss their common needs and goals as well as the algorithms and tools that best support them.

The workshop is intended to become recurrent since this is an important and growing area of research and technology development.

The EOOLT Workshop addresses the current state of the art of EOO modeling languages as well as open issues that currently still limit the expression power and usefulness of such languages through a set of full-length presentations, short position papers, and forum discussions.

Papers and contributions are welcome that offer presentations and discussions of existing languages and tools, their capabilities and limitations; reports on practical experience; demonstrations of languages, tools, ideas, and concepts; positions related to relevant questions; and discussion topics.

Despite the short deadlines and the fact that this is a new not very established workshop series, there was a good response to the call-for-papers. Thirteen papers and one presentation were accepted to the workshop program. All papers were subject to reviews by the program committee.

The workshop program started with a welcome and introduction to the area of equation-based object-oriented languages, followed by paper presentations and discussion sessions after presentations of each set of related papers. EOOLT'2007 was hosted by the Technical University of Berlin, in conjunction with the ECOOP'2007 conference.

2 Organizers

Peter A. Fritzson received his M.Sc. in engineering 1975 and Ph.D. in computer science 1984, both from Linköping University. He is Professor and Director of the Programming Environment Laboratory (Pelab), at the Department of Computer and Information Science, Linköping University, Sweden. Peter Fritzson is vice chairman of the Modelica Association, an organization he helped to establish, and during 1999-2007 served as chairman of the Scandinavian Simulation Society, and secretary of the European simulation organization, EuroSim. His main area of interest is software engineering, especially languages, programming and debugging tools and environments; during recent years with special emphasis on modeling and simulation, and is currently leading the OpenModelica modeling and simulation open source tool effort. Professor Fritzson has authored or co-authored more than 180 technical publications, including 13 books/proceedings. In 1994 he published a textbook "Principles of Object-Oriented Modeling and Simulation with Modelica", 939 pages, Wiley-IEEE Press. He has served as chair of a number of international conferences and workskops, and took the initiative to start the AADEBUG and EOOLT workshop series.

Prof. Dr.-Ing. Peter Fritzson
Programming Environment Laboratory (PELAB)
Linköping University
SE-581 83 Linköping
Sweden
Phone: +46(13)281484
Fax: +46(13)285899
Mobile: +46(708)281484
Email: petfr@ida.liu.se
URL: http://www.ida.liu.se/labs/pelab/

François E. Cellier received his BS degree in electrical engineering in 1972, his MS degree in automatic control in 1973, and his PhD degree in technical sciences in 1979, all from the Swiss Federal Institute of Technology (ETH) Zurich. Dr. Cellier worked at the University of Arizona as professor of Electrical and Computer Engineering from 1984 until 2005. He recently returned to his home country of Switzerland where he assumed a position with ETH Zurich. Dr. Cellier's main scientific interests concern modeling and simulation methodologies, and the design of advanced software systems for simulation, computer aided modeling, and computer-aided design. Dr. Cellier has authored or co-authored more than 200 technical publications, and he has edited several books. He published a textbook on Continuous System Modeling in 1991 and a second textbook on Continuous System Simulation in 2006, both with Springer-Verlag, New York. He served as general chair or program chair of many international conferences, and served recently as president of the Society for Modeling and Simulation International.

Prof. Dr. François E. Cellier
Institute of Computational Science

CAB G82.1
ETH Zürich
CH-8092 Zürich
Switzerland
Phone: +41(44)632-7474
Fax: +41(44)632-1374
Mobile: +41(79)416-7546
Email: fcellier@inf.ethz.ch
URL: http://www.inf.ethz.ch/~fcellier/

Christoph Nytsch-Geusen received his PhD in Engineering at Technology University of Berlin in 2001. Since 2001 he is responsible for the development of simulation methods and simulation tools for complex technical systems at Fraunhofer FIRST. These research activities are focused on object oriented modeling, model compilers, simulation runtime systems, simulator coupling and the integration of simulation tools in the design process of technical systems. He was the leader of a joint project of six Fraunhofer Institutes (2004–2007), within the Modelica-based simulation tool MOSILAB was developed. At University of Arts Berlin he holds a chair in "Building Services Engineering" since 2007, where his research activities are focused on modeling and simulation of complex energy supply systems for single buildings and whole districts.

Prof. Dr.-Ing. Christoph Nytsch-Geusen
Fraunhofer Institute for Computer Architecture and Software Technology
D-12489 Berlin, Germany
Phone +49 (0) 30/6392-1919
Telefax: +49 (0) 30/6392-1805
Email: christoph.nytsch@first.fraunhofer.de
Internet: http://www.first.fraunhofer.de

David Broman is currently pursuing his PhD in computer science at Linköping University, Sweden, where he also received his M.Sc. degree in 2001. Before he started his PhD work, he worked as a software engineer and technical project manager for a security company in Stockholm. David's current research interest is focusing on language semantics and type systems of equation-based object-oriented languages. He is a member of the Modelica Association and has been active in the Modelica design group since 2005.

David Broman
Department of Computer and Information Science
Linköping University
SE-581 83 Linköping
Sweden
Phone: +46(0)13-285724
Fax: +46(0)13-285899
Mobile: +46(0)707-909075
URL: http://www.ida.liu.se/~davbr/

3 Participants

The number of participants of this first EOOLT workshop was 19, of which 18 were physically present and one connected by electronic web conferencing; 17 are present in the table below.

Name	Affiliation	Country	Email
Bernhard Bachmann	University of Applied Sciences, Bielefeld	Germany	bernhard.bachmann@fh-bielefeld.de
Felix Breitenecker	Vienna University of Technology	Austria	felix.breitenecker@tuwien.ac.at
David Broman	Linköping University	Sweden	davbr@ida.liu.se
François E. Cellier	ETH	Switzerland	fcellier@inf.ethz.ch
Peter Fritzson	Linköping University	Sweden	petfr@ida.liu.se
Ramine Nikoukhah	Inria	France	ramine.nikoukhah@inria.fr
Henrik Nilsson	University of Nottingham	UK	nhn@cs.nott.ac.uk
Christoph Nytsch-Geusen	University of Fine Arts, Berlin,	Gerrmany	christoph.nytsch@first.fraunhofer.de
Adrian Pop	Linköping University	Sweden	adrpo@ida.liu.se
Olaf Enge-Rosenblatt	Fraunhofer Institute for Integrated Circuits	Germany	olaf.enge@eas.iis.fraunhofer.de
Miguel A. Rubio	UNED	Spain	marubio@dia.uned.es
Carl-Johan Sjöstedt	Royal Institute of Technology	Sweden	carlj@md.kth.se
Günther Zauner	Vienna University of Technology	Austria	guenther.zauner@drahtwarenhandlung.at
Dirk Zimmer	ETH	Switzerland	dzimmer@inf.ethz.ch
Johan Åkesson	Lund University	Sweden	jakesson@control.lth.se
Michael Cebulla	TU Berlin	Germany	mce@cs.tu-berlin.de
Gilad Bracha	Cadence	U.S.A.	gilad@bracha.org

4 Contributions

All papers are published electronically by Linköping University Electronic Press and available in the electronic proceedings at `http://www.ep.liu.se/ecp/024/`.

All presentations (together with the papers) are also available at the EOOLT'2007 web site: `http://www.ida.liu.se/labs/pelab/conf/eoolt07/`.

The workshop sessions are briefly described below. Each session started with paper presentations, followed by a discussion related to the topic of that particular session. Some discussion also took place during the paper presentations.

4.1 Integrated System Modeling Approaches

This session grouped paper that especially emphasized integrated modeling tools for complex systems and integrated modeling environments aimed towards the whole development process. Session chair: Peter Fritzson.

In "The use of the UML within the modeling process of Modelica models," Christoph Nytsch-Geusen presented work on an integration of a subset of UML and Modelica called UML^{H}. The UML class diagrams, state chart diagrams, and collaboration diagrams are integrated in an extended subset of Modelica including special UML^{H}, graphical annotations, and language extensions for statechart support. An example model and simulation of a Pool-Billiard game using this approach was shown.

In "Towards Unified System Modeling with the ModelicaML UML Profile," Adrian Pop, David Akhvlediani, and Peter Fritzson presented the new ModelicaML UML profile, based on both SysML and Modelica, with (currently incomplete) implementation in Eclipse based on the Eclipse Modeling Framework EMF. This integrates the graphical software modeling of UML with Modelica modeling of physical systems, thus giving a rather complete integrated approach for full software-hardware complex system development.

In "Developing Dependable Automotive Embedded Systems using the EAST-ADL," Carl-Johan Sjöstedt, De-Jiu Chen, Phillipe Cuenot, Patrick Frey, Rolf Johansson, Henrik Lönn, David Servat and Martin Törngren presented the EAST-ADL modeling language (Embedded Automotive Systems – Architectural Description Language), which is structured in five levels: vehicle, analysis, design, implementation, and operation. Since EAST-ADL is partly implemented as a UML2 profile, an attempt to model a small electrical circuit using SysML parametric diagrams was presented. Difficulties were observed because of the absence of flow variables in SysML. The solution of using flow-split is rather cumbersome.

During the following discussion some questions were raised. For example, why not use the SysML profile instead of UML^{H}? The explanation is that UML^{H} was developed much earlier than SysML, partly based on the Smile system and later the MOSILAB effort. The second talk also presented an integration approach between Modelica and UML, now based on SysML and Eclipse. There was a question regarding the difference between UML and Modelica connection diagrams. Why do you have both? The UML-style is less compact but better known to software developers, the Modelica-style is common among engineers and usually more compact.

Regarding the third talk, there were some questions – why is EAST-ADL needed, since we have Modelica, VHDL-AMS, SysML etc. EAST-ADL is more specific to the automotive sector, but there are similar efforts and library developments, e.g., for automotive modeling based on Modelica. The EAST-ADL implementation is currently an UML profile.

There was a general discussion regarding continued work in the area, because of the parallel efforts, e.g., Modelica and SysML, as to which approach is more fruitful, standardizing or providing translations between the formalisms. One problem concerns the fuzzy semantics of many modeling languages, especially UML/SysML, which need to be made more precise. There is also a trend with increasing importance to have integrated system development approaches for whole systems including both software and hardware.

It was also remarked that too early standardization and efforts at adoption might be harmful. Instead, focus should be on the hard technical problems, try to solve those, and exchange experiences and results between the different groups.

4.2 Hybrid Modeling and Variable Structure Systems

As apparent from the title, the talks in this session concern two topics: hybrid continuous-time and discrete-time modeling and simulation, and variable structure systems where the structure and number of equations can change at run-time. Session chair was Bernhard Bachmann.

In "Hybrid dynamics in Modelica: Should all events be considered synchronous," Ramine Nikoukhah compared the event semantics of Modelica and Scicos and argued that the Modelica semantics is ambiguous, leading to distinct interpretations by different implementations (e.g., Dymola and Scicos) as to whether certain events are synchronous or not. It is argued that it is better for events to be considered synchronous only if they can be traced back (e.g., through equations) to the same event source, and that this leads to a more efficient and simpler implementation.

In "Extensions to Modelica for efficient code generation and separate compilation," Ramine Nikoukhah complemented the topic of the previous presentation by presenting a consistent way of separate compilation of (Modelica) models as black boxes, which then would have event inputs and event outputs as well as regular inputs and outputs. This would make it possible to connect separately compiled Modelica models, e.g., to models implemented in other formalisms, such as Scicos, Simulink or plain C code.

The next two talks concerned the topic of modeling and simulation of variable structured systems (structural dynamics), i.e., systems where objects can be introduced, removed, and connected/disconnected at run-time. MOSILAB has previously developed a solution to the modeling of structural dynamics by a language extension to Modelica supporting UML state charts.

In "Enhancing Modelica towards variable structure systems," Dirk Zimmer presented ideas and ongoing work on a new Modelica-like modeling language, called Sol, where several restrictions in standard Modelica have been removed and new features added, in particular being able to handle variable structured systems. Sol is intended as a research language to explore ideas, with an interpretive implementation. A model of a machine with a flywheel is sketched in the Sol language.

In "Functional Hybrid Modeling from an Object-Oriented Perspective," Henrik Nilsson, John Peterson, and Paul Hudak presented ideas for how to combine functional programming and modeling to create more powerful modeling languages for hybrid systems. In earlier work, they created a framework for causal hybrid modeling called Functional Reactive Programming (FRP). The goal now is to generalize this towards equation-based modeling (Functional Hybrid Modeling, FHM), in particular to create a declarative language that supports highly structurally dynamic models, based on the power of functional programming concepts combined with run-time code generation. The concept of first class signal relations is introduced. Ideas are sketched for an equation-based language (called Hydra), and a broken pendulum example model is shown. The work is still in an early stage.

In the following discussion, a number of issues were raised. One point is that there is concern for increased model complexity if we remove the Modelica rule that the number of variables must equal the number of equations, as in the first talk for discrete-time variables. Partial answer: these are essentially proposals for intermediate language constructs for hybrid system modeling. They allow to introduce the valuable feature of separate compilation of models/blocks, e.g., in conjunction with Modelica.

Regarding the Sol language, there were some questions about the implementation strategy. Why are programs coded in Sol being interpreted rather than compiled? Motivation: the approach makes it easy to handle structural variation at run-time, i.e., to create/exchange/destruct components on the fly. Sol is designed as a research tool only. The language serves as a proof-of-concept tool, and its implementation must be considered a prototype environment only. Will this approach not limit the tool to dealing with toy problems only? Could just-in-time code generation be used instead of interpretation? Right now, the language is being interpreted, and the flattened system of equations is updated at run-time.

The talk on functional hybrid modeling introduced many interesting ideas. It focused on relations of signals. At a structural change, use a switching function to add or remove signal relation functions in a collection. Hierarchical modeling is done through signal relations. Classes are nice but not needed for hierarchical modeling. One question: why not use constraints? Answer: functions are essential. Relations describe Modelica equations. Another question: functional programming is not widely used, what should be done about that? Answer: this is primarily aimed at a semantic language for dynamic modeling environments.

4.3 Modeling Languages, Specification, and Language Comparison

This session contained presentations and discussions on modeling languages, tools, and comparisons, as well as the topic of more precise specifications of modeling language semantics. Session chair was Christoph Nytsch-Geusen.

In “Important characteristics of VHDL-AMS and Modelica with respect to model exchange,” Olaf Enge-Rosenblatt, Joachim Haase, and Christoph Clauß presented a modeling language comparison between the IEEE-standardized language VHDL-AMS and the language Modelica to describe analog and mixed-signal systems. The underlying modeling approaches were compared. Further, the potential to transform models written in one language into models of the other language was discussed.

In “Modeling Structural Dynamics Systems in Modelica/Dymola, Modelica /Mosilab, and AnyLogic”, Günther Zauner, Daniel Leitner, and Felix Breitenecker presented a tool comparison between the three state-of-the-art DAE simulation environments Dymola, Mosilab, and AnyLogic regarding the possibilities of coupling of different state spaces. For this purpose, the three modeling approaches “parallel model setup,” “serial model setup,” and “combined model setup” were discussed. The analogies and discrepancies between these approaches were discussed by use of the classical constrained pendulum example as defined in the ARGESIM comparison C7.

In “Abstract Syntax Can Make the Definition of Modelica Less Abstract,” David Broman and Peter Fritzson discussed different aspects of formulating a Modelica language specification. They proposed a "middle-way" strategy, which can make the specification both clearer and easier to reason about. For this purpose, a proposal

was formulated, whereby the current informally specified Modelica semantics are complemented with several grammars, specifying intermediate representations of abstract syntax.

In "Physical Modelling with ModelVision, a DAE Simulator with Features for Hybrid Automata," Günther Zauner, Yuri Senichenkov, and Yuri Kolesov presented the modeling capabilities of the hybrid simulator ModelVision. The simulation technology of this simulation tool is based on hybrid state charts, which makes parallel, serial, and conditional combination of continuous models possible, described by DAEs as mathematical equations. State models themselves can be instantiated and replaced as objects during the simulation experiment for modeling and simulating structural dynamic systems. The talk was given by Felix Breitenecker in proxy for the authors.

Regarding the first presentation, one question was regarding VHDL-AMS models: are they compatible between different tools? Is the objective a full model translation tool between VHDL-AMS and Modelica? The answer: not really at this stage, this would be too difficult, and modeling language semantics are still too imprecise.

The second presentation compared three different tools using a constrained pendulum example. The question arose, how these tools can be fairly compared with each other? For example, Modelica allows DAEs, and its implementations usually support index reduction, but AnyLogic currently does not support either index reduction or event handling, which is a serious limitation. Another question raised in the context of MOSILAB that supports UML state charts: are such asynchronous UML state charts really desirable since they prevent the modeler from being able to prove the absence of deadlocks in models? Would it not be more fruitful to work with synchronous model approaches like the Modelica StateGraph library?

The third talk presented ideas how to make model language specifications more precise by combining informal specification, abstract syntax, and additional grammar fragments. One question concerned itself with the kind of grammars used: are they context free or context dependent? Answer: they are context dependent, since certain meta variables range over names and identifiers.

The presentation on the ModelVision tool was an interesting example of a long-term tool development effort in Russia going on in parallel to the mainstream developments, and now supporting many advanced features such as hybrid simulation and index reduction.

In general, it was concluded that more precise language semantics definitions are needed to make progress in model exchange between different modeling languages. The third talk presented some ideas how to make progress in this direction. However, much work is needed before models can be translated automatically, and in many cases there are low-level properties that make automatic model exchange really hard to attain.

4.4 Tools and Methods

The tools and methods session presented three papers of different aspects of modeling and simulation tools. Session chair was Peter Fritzson.

In "An Approach to the Calibration of Modelica Models," Miguel A. Rubio, Alfonso Urquía, and Sebastian Dormido presented a new Modelica library GAP*/*Lib based on genetic algorithms used for identification of unknown model parameter

values from measurement data. An application of estimating parameters in a fuel cell model was presented.

In "Dynamic Optimization of Modelica Models – Language Extensions and Tools," Johan Åkesson presented work on extending the Modelica language also for optimization, with a language extension called Optimica. This superimposes four aspects on a model: information about Modelica variables, specification of a grid, definition of a cost function, and specification of constraints. Its use in formulating and solving a start-up problem for a plate reactor system was also presented.

In the paper "Robust Initialization of Differential Algebraic Equation," Bernhard Bachmann, Peter Aronsson, and Peter Fritzson presented a more robust method for initialization of DAE systems based on initialial equation systems. The method allows the initial equation systems to be overdetermined, solved by least square minimization, which gives additional flexibility in specifying initial conditions locally in each component model without concerns for possible problems with overdetermined initial equation systems. Currently, initial equations usually have to be specified in the top level application model to avoid problems with overdetermined systems. An application with a 3-phase electrical system with Park transformations was presented.

In the following discussion, regarding talk 1, identification of unknown parameters, there was a question regarding the choice of optimization algorithms: why genetic algorithms, aren't those inefficient? Answer: genetic algorithms are simple to understand and use and possibly modify. Another advantage is that there is no need to change the model. Another question: how do other methods, e.g., Tabu search, compare? Answer: this has not yet been investigated.

Regarding the optimization talk, there were questions regarding formulating the upper and lower bounds, needs for a special backend. A program called Socks was mentioned that should be somewhat similar to the presented work. Can we switch optimization algorithms without changing the model?

Finally, regarding the robust initialization, there were questions regarding the scalability: will the method work efficiently with 100 or more state variables instead of 6 as in the example? According to the presenter, the approach should scale up. The optimization algorithm is fairly efficient, and even complex applications have seldom more than a hundred or a few hundred state variables. Another question concerned if-equations: which region should be optimized? Answer: keep one branch constant while varying the other, then keep the second branch constant while varying the first.

To summarize: tools and methods for analyses related to modeling and simulation are becoming increasingly important to help prepare models for solution, e.g., parameter identification and initialization, or post processing and additional analysis such as optimization, which uses simulation for a particular purpose.

5 Discussion of Future Directions of Equation-Based Languages

The workshop ended with a general discussion about possible future directions of EOO languages and tools. The discussion was roughly divided into the following topics.

5.1 New Directions

What new directions can be discerned in the area? Optimization was mentioned as one such area. Metamodeling, model unification (see below) is another. Static analysis of models for verification is already established in the embedded systems community, and should increasingly be relevant here.

A current trend is increased emphasis on model-based development for embedded systems, not just simulation. This includes generation of embedded code, e.g., in controllers, perhaps with fixed-point support, and more emphasis on real-time issues in general.

5.2 Tool Integration and Tool Interfaces

There is increasing recognition that tools need to be more modular with clearly defined interfaces/APIs and interface formats between the phases/modules. This will have advantages such as:

- Enables extensible plugin development e.g., as in Eclipse.
- Enables tool certification.

There was also some discussion on the availability of the flat structure of equations. Is this enough? The main reason is that it is easier for many tools to operate on the flat structure.

5.3 Variable Structure Systems

Support for variable structure systems is a hot future topic that needs to find good solutions. The challenge is to combine the advantages of efficient code and static checking of fixed-structure models with the flexibility of variable structured systems. For example, what is the cost of restructuring, and can that be brought down? These, and similar, issues need to be studied more closely.

5.4 Metamodeling, Reflection, Model Unification

Metamodeling, operations on models, and model transformations are topics of increasing importance and tools become more capable and extensible. The models themselves could be made extensible by including new analyses and transformations in the models instead of in monolithic tools.

There is a trend to use separate untyped scripting languages together with modeling languages. Instead, it might be desirable to generalize the modeling languages themselves to handle models, have functions that return models, and operate on models with a type system that includes all this.

5.5 Integrated Modeling Approaches

Integrated modeling approaches could mean approaches for metamodeling by combining black box models, co-simulation, or tool integration in general. This is an increasing trend and strongly dependent on the tool integration mentioned previously.

6 Conclusions

The participants felt that this was a successful workshop, the first in its series. The area of equation-based object-oriented (EOO) languages and tools is of rapidly increasing importance. It is important to engage more computer scientists in this area, which is one of the motivations of co-locating the workshop with ECOOP.

It is important to bring in a wider spectrum of languages and tools, with less dominance of Modelica-related work. For example, why not include also PDE or FEM modeling languages and tools? It was felt that the workshop should be continued and possibly expanded. The discussions in the workshop were good. In case there will be more papers presented at future workshops, the time per presentation has to be reduced in order to keep enough time for discussions.

Some references are given below as a background to this area.

References

[1] Accellera, Cadence: Verilog-AMS Language Reference Manual Version 2.2, Published by: Accellera, 1370 Trancas Street, #163, Napa, CA 94558 (November 2004)

[2] Augustin, D.C., Fineberg, M.S., Johnson, B.B., Linebarger, R.N., Sansom, F.J., Strauss, J.C.: The SCi Continuous System Simulation Language (CSSL). Simulation (9), 281–303 (1967)

[3] Birtwistle, G.M., Dahl, O.J., Myhrhaug, B., Nygaard, K.: SIMULA BEGIN. Auerbach Publishers, Inc. (1973)

[4] Breunese, A.P.J., Broenink, J.F.: Modeling Mechatronic Systems Using the SIDOPS+ Language. In: Proceedings of ICBGM 1997, 3rd International Conference on Bond Graph Modeling and Simulation, Phoenix, Arizona. Simulation Series, vol. 29(1), pp. 301–306. SCS Publishing, San Diego, California (1997), http://www.rt.el.utwente.nl/proj/modsim/modsim.htm

[5] Cellier, F.E.: Continuous System Modelling, p. 755. Springer, New York (1991)

[6] Cellier, F.E., Kofman, E.: Continuous System Simulation, p. 643. Springer, New York (2006)

[7] Christen, E., Bakalar, K.: VHDL-AMS – A Hardware Description Language for Analog and Mixed-Signal Applications. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 46(10), 1263–1272 (1999)

[8] Clabaugh, J., Tolsma, J.E., Barton, P.I.: Abacuss II: Advanced Modeling Environment and Embedded Simulator, and Abacuss II Syntax Manual. Massachusetts Institute of Technology, Chemical Engineering System Research Group (1999), Available at http://yoric.mit.edu/abacuss2/abacuss2.html

[9] Elmqvist, H.: A Structured Model Language for Large Continuous Systems. Ph.D. thesis, TFRT-1015, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden (1978)

[10] Ernst, T., Jähnichen, S., Klose, M.: The Architecture of the Smile/M Simulation Environment. In: Proceedings 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Berlin, Germany, vol. 6, pp. 653–658 (1997)

[11] Fritzson, P.: Principles of Object-Oriented Modeling and Simulation with Modelica 2.1, p. 940. Wiley-IEEE Press (2004) ISBN 0-471-471631

[12] Fritzson, P., Viklund, L., Fritzson, D., Herber, J.: High Level Mathematical Modeling and Programming in Scientific Computing, IEEE Software, pp. 77–87 (July 1995)
[13] Mattsson, S.-E., Andersson, M.: The Ideas Behind Omola. In: CADCS 1992. Proceedings of the 1992 IEEE Symposium on Computer-Aided Control System Design, Napa, California, March 17-19, 1992, pp. 23–29 (1992)
[14] Oh, M., Pantelides, C.C.: A modelling and Simulation Language for Combined Lumped and Distributed Parameter Systems. Computers and Chemical Engineering 20(6–7), 611–633 (1996)
[15] Piela, P.C., Epperly, T.G., Westerberg, K.M., Westerberg, A.W.: ASCEND: An Object-Oriented Computer Environment for Modeling and Analysis: The Modeling Language. Computers and Chemical Engineering 15(1), 53–72 (1991)
[16] Sahlin, P., Sowell, E.F.: A Neutral Format for Building Simulation Models. In: Proceedings of the Conference on Building Simulation, IBPSA, Vancouver, Canada, pp. 147–154 (1989)
[17] Sargent, R.W.H., Westerberg, A.W.: Speed-Up in Chemical Engineering Design. Chemical Engineering Research and Design 42a, 190–197 (1964)
[18] The Mathworks. Simulink – Simulation and Model-Based Design (Last accessed: March 6, 2007), http://www.mathworks.com/products/simulink/
[19] The Modelica Association. The Modelica Language Specification Version 3.0 (September 2007), http://www.modelica.org
[20] Tiller, M.: Introduction to Physical Modeling with Modelica, p. 368. Springer, New York (2001)
[21] UML Homepage: http://www.uml.org
[22] van Beek, D.A., Man, K.L., Reniers, M.A., Rooda, J.E., Schiffelers, R.R.H.: Syntax, and consistent equation semantics of hybrid Chi. The Journal of Logic and Algebraic Programming 68, 129–210 (2006)