Advances in Soft Computing 39

Editor-in-Chief: J. Kacprzyk

Advances in Soft Computing

Editor-in-Chief

Further volumes of this series can be found on our homepage: springer.com

Marek Kurzynski, Edward Puchala,
Michal Wozniak, Andrzej Zolnierek (Eds.)
Computer Recognition Systems, 2005
ISBN 978-3-540-25054-8

Abraham Ajith, Yasuhiko Dote,
Takeshi Furuhashi, Mario Köppen,
Azuma Ohuchi, Yukio Ohsawa (Eds.)
Soft Computing as Transdisciplinary Science and Technology, 2005
ISBN 978-3-540-25055-5

Barbara Dunin-Keplicz, Andrzej
Jankowski, Andrzej Skowron,
Marcin Szczuka (Eds.)
Monitoring, Security, and Rescue Techniques in Multiagent Systems, 2005
ISBN 978-3-540-23245-2

Frank Hoffmann, Mario Köppen,
Frank Klawonn, Rajkumar Roy (Eds.)
Soft Computing Methodologies and Applications, 2005
ISBN 978-3-540-25726-4

Mieczyslaw A. Klopotek, Slawomir T.
Wierzchon, Kryzysztof Trojanowski
(Eds.)
Intelligent Information Processing and Web Mining, 2005
ISBN 978-3-540-25056-2

Abraham Ajith, Bernard de Bacts,
Mario Köppen, Bertram Nickolay (Eds.)
Applied Soft Computing Technologies: The Challenge of Complexity, 2006
ISBN 978-3-540-31649-7

Mieczyslaw A. Klopotek, Slawomir T.
Wierzchon, Kryzysztof Trojanowski
(Eds.)
Intelligent Information Processing and Web Mining, 2006
ISBN 978-3-540-33520-7

Ashutosh Tiwari, Joshua Knowles,
Erel Avineri, Keshav Dahal,
Rajkumar Roy (Eds.)
Applications and Soft Computing, 2006
ISBN 978-3-540-29123-7

Bernd Reusch, (Ed.)
Computational Intelligence, Theory and Applications, 2006
ISBN 978-3-540-34780-4

Miguel López-Díaz, María ç. Gil,
Przemysław Grzegorzewski, Olgierd
Hryniewicz, Jonathan Lawry
Soft Methodology and Random Information Systems, 2006
ISBN 978-3-540-34776-7

Ashraf Saad, Erel Avineri, Keshav Dahal,
Muhammad Sarfraz, Rajkumar Roy (Eds.)
Soft Computing in Industrial Applications, 2007
ISBN 978-3-540-70704-2

Ashraf Saad, Erel Avineri, Keshav Dahal,
Muhammad Sarfraz, Rajkumar Roy (Eds.)

Soft Computing in Industrial Applications

Recent and Emerging Methods and Techniques

Springer

Editors

Dr. Ashraf Saad
Department of Computer Science
Armstrong Atlantic State University
11935 Abercorn Street
Savannah, Georgia 31419-1997
USA
E-mail: ashraf@cs.armstrong.edu

Dr. Erel Avineri
Centre for Transport & Society
Faculty of the Built Environment
University of the West of England
Frenchay Campus
Coldharbour Lane
Bristol BS16 1QY
UK
E-mail: Erel.Avineri@uwe.ac.uk

Dr. Keshav Dahal
MOSAIC Research Group
University of Bradford
Department of Computing
Bradford BD7 1DP
UK
E-mail: K.P.Dahal@Bradford.ac.uk

Dr. Muhammad Sarfraz
Information & Computer Science Department
King Fahd University of Petroleum & Minerals
KFUPM #1510
Dhahran 31261
Saudi Arabia
E-mail: sarfraz@ccse.kfupm.edu.sa,
sarfraz@kfupm.edu.sa

Prof. Rajkumar Roy
Decision Engineering Centre
Manufacturing Department
Cranfield University
Bedford MK43 OAL
UK
E-mail: r.roy@cranfield.ac.uk

Library of Congress Control Number: 2007923718

ISSN print edition: 1615-3871
ISSN electronic edition: 1860-0794
ISBN-10 3-540-70704-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-70704-2 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: by the authors and SPS using a Springer LaTeX macro package

Printed on acid-free paper SPIN: 11585275 89/SPS 5 4 3 2 1 0

Preface

On behalf of all members of the International Technical Program Committee of the 11th Online World Conference on Soft Computing in Industrial Applications (WSC11), we would like to extend our sincere welcome to you. The conference continues a tradition started over a decade ago by the World Federation of Soft Computing (WFSC) to bring together researchers interested in advancing state of the art in the field. Continuous technological improvements since then continue to make this online forum a viable gathering format for a world class conference.

The program committee received a total of 63 submissions, of which 61 papers qualified for peer review by the International Program Committee. Each paper was then reviewed by at least three referees, culminating in the acceptance of 30 papers for publication. Authors of all accepted papers were then notified to prepare and submit their final manuscripts and conference presentations. This resulted in a total of 28 final submissions by 73 authors that comprise the six sessions of the conference program. Based on the reviewers' reports, the authors provided revised versions of the papers – all of them are featured in this book. Also featured is an invited paper based on a keynote presentation. The authors of several outstanding papers have been invited to submit significantly revised and extended versions of their papers to the Applied Soft Computing Journal.

We extend our sincere thanks to all authors and to all members of the International Program Committee for their clear and unwavering commitment to the success of WSC11. Reflecting the worldwide nature of WSC11, authors, members of the program committee and the conference organizers are from over 20 countries and five continents. We also extend our thanks to our keynote speaker, Dr. Pieter Mosterman of the MathWorks for his contributed talk.

November 29, 2006

Ashraf Saad
General Chair of WSC11
Savannah, Georgia, USA

Erel Avineri
Program Chair of WSC11
Bristol, UK

Message from the WSC11 General Chair and Program Chair

It is our pleasure to officially announce the start of the conference. The official WSC11 web site has been relocated since August to the following URL: http://www.cs.armstrong.edu/wsc11/. Please make the necessary changes to any web pages that you maintain with reference to the conference. That will increase the chances of search engines pointing to the correct WSC11 web site.

An opening note has been posted to the conference web site along with the final pdf version of all accepted papers. With regard to the presentation of papers and the keynote, we will be able to support (for the first time in WSC's history) real-time presentations via audio conferencing. This is made possible through a kind three-week trial offer (for the duration of the conference) of Elluminate (http://www.Elluminate.com), a Java-based (http://java.sun.com/products/javawebstart/) webinar environment. In return, we will provide feedback about the use of this web-based conferencing tool in support of our worldwide conference. In order to get an idea of the use of this tool, please visit the following URL: https://sas.elluminate.com/m.jnlp?sid=1125&password=M.161974A26FAAF95DB6C50F2C6CFF05 where an image version of the opening note is currently posted for testing purposes.

Therefore, we request from each correspondence author to email us back by Friday, September 22, with his/her availability to make a 25-30 minutes presentation during the upcoming two weeks (Sep 25-Oct 6). Please provide us with 2-3 possible times, and indicate your local time zone as it relate to GMT (e.g., EST in the US is GMT-5, while Brazil should be GMT-4). A presenter will need a Java-enabled computer, with a reasonable high quality connection to the Internet, and which is also equipped with a speaker and a microphone (or a headset). We will schedule all presentations and upload into Elluminate the presentation slides that have been submitted in August. A final schedule of presentations will be posted and emailed to all by Monday, September 25. All interested participants will then be able to connect to a presentation at the scheduled time, up to a maximum of 30 seats per session. We will expect session chairs to attend as many of the presentations of their sessions as possible.

It is indeed an exciting development for us to be able to support a synchronous mode of interaction for WSC11 given our global community. We also hope to witness a strong level of participation in the sessions by researchers from all four corners of the globe.

September 18, 2006

Ashraf Saad
General Chair of WSC11
Savannah, Georgia, USA

Erel Avineri
Program Chair of WSC11
Bristol, UK

WSC11 Organization and International Program Committee

General Chair

Ashraf Saad, Armstrong Atlantic State University**, USA
** Formerly with the Georgia Institute of Technology

Program Chair

Erel Avineri, University of the West of England, Bristol, UK

Advisory Board

Hisao Ishibuchi, Osaka Prefecture University, Japan
Rajkumar Roy, Cranfield University, UK
Ajith Abraham, Chung-Ang University, Korea
Mario Köppen, Fraunhofer IPK, Berlin, Germany

International Co-chairs

Lakhmi Jain, University of South Australia, Australia
Serge Popov, Kharkiv University of Radio Electronics, Ukraine
Muhammad Sarfraz, King Fahd University of Petroleum and Minerals, Saudi Arabia
Ashitosh Tiwari, Cranfield University, UK

Publicity Chair

Keshav Dahal, University of Bradford, UK

International Technical Program Committee

Janos Abonyi, University of Veszprem Folyamatmérnöki Tanszék, Hungary
Bart Baesens, Catholic University of Leuven, Belgium
Valeriu Beiu, United Arab Emirates University, UAE
Sugato Bagchi, IBM Research, USA
Soumya Banerjee, BITS Mesra, India
Christian Blum, Universitat Politecnica de Catalunya, Spain
Ulrich Bodenhofer, Software Competence Center, Austria
Andrea Bonarini, Politecnico de Milano, Italy

Oscar Castillo, Instituto Tecnológico de Tijuana, Mexico
Siam Charoenseang, King Mongkut's University of Technology, Thailand
Leandro Coelho, Pontifical Catholic University of Parana, Brazil
Carlos A. Coelho, CINVESTAV, Mexico
Oscar Cordon, University of Granada, Spain
Gaspar Cunha, University of Minho, Potugal
Suash Deb, National Institute of Science & Technology, India
Guy De Tré, Ghent University, Belgium
Mauro Dell'Orco, University of Bari, Italy
Giuseppe Di Fatta, University of Konstanz, Germany
Katrin Franke, Fraunhofer IPK, Germany
Aureli Soria Frisch, Universitat Pompeu Fabra, Spain
Xiao-Zhi Gao, Helsinki University of Technology, Finland
Takeshi Furuhashi, Nagoya University, Japan
Crina Grosan, Babes-Bolyai University, Romania
Roderich Gross, Universite Libre de Bruxelles, Belgium
Hani Hagras, University of Essex, UK
Ioannis Hatzilygeroudis, University of Patras, Greece
Ayanna Howard, Georgia Institute of Technology, USA
Yaochu Jin, Honda Research Institute Europe, Germany
Uri Kartoun, Ben Gurion University of the Negev, Israel
Okyay Kaynak, Bogazici University, Turkey
Frank Klawonn, University of Applied Sciences, Germany
Joshua Knowles, University of Manchester, UK
Andreas König, Technische Universitat Kaiserslautern, Germany
Renato Krohling, University of Dortmund, Germany
Reza Langari, Texas A&M, USA
Luis Magdalena, Universidad Politecnica de Madrid, Spain
Max Manfrin, Universite Libre de Bruxelles, Belgium
Christophe Marsala, Universite P. et M. Currie, France
Patricia Melin, Instituto Tecnológico de Tijuana, Mexico
Sanaz Mostaghim, ETH-Zurich, Switzerland
Mehmet K Muezzinoglu, University of Louisville, USA
Lakshmi Narasimhan, The University of Newcastle, Australia
Detlef D Nauck, British Telecom, UK
Nadia Nedjah, State University of Rio de Janeiro, Brazil
Andreas Nuernberger, Universität Magdeburg, Germany
Jae C. Oh, Syracuse University, USA
Sankar K. Pal, Indian Statistical Institute, India
Vasile Palade, Oxford University, UK
Gerardo Rossel, Universidad Abierta Interamericana, Argentina
Yos Sunitiyoso, University of the West of England, Bristol, UK
Vicenc Torra, AI Research Institute, CSIC, Spain
Edward Tunstel, Jet Propulsion Lab/NASA, USA
Marley Vellasco, Pontifical Catholic University of Rio de Janeiro, Brazil
Christian Woehler, DaimlerChrysler AG, Germany
Berend Jan van der Zwaag, University of Twente, The Netherlands

Contents

Invited Keynote

Part I: Soft Computing in Computer Graphics, Imaging and Vision

Part II: Control Systems

Part III: Pattern Recognition

Part IV: Classification

Part V: Soft Computing for Modeling, Optimization and Information Processing

Part VI: Soft Computing in Civil Engineering and Other Applications

Part I

Soft Computing in Computer Graphics, Imaging and Vision

Hybrid Dynamic Systems in an Industry Design Application

Pieter J. Mosterman and Elisabeth M. O'Brien

The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760, USA
`{pieter_j_mosterman,elisabeth.obrien}@mathworks.com`

Abstract. The term *hybrid dynamic system* is a term for a mathematical system that combines behavior of a continuous nature with discontinuous changes. Such systems are often formed by the underlying computational representation of models used in the design of control and signal processing applications, for example in the automotive and aerospace industries. This paper outlines the benefits of Model-Based Design and illustrates how many different formalisms may be essential in model elaboration, such as time-based block diagrams, state transition diagrams, entity-flow networks, and multi-body diagrams. The basic elements of the underlying hybrid dynamic system computational representation are presented and it is shown how these elements combine to form different classes of behaviors that need to be handled for simulation.

Keywords: Model-Based Design; Hybrid Dynamic Systems; Hybrid Systems; Multi-Formalism Modeling; Embedded Control Systems; Networked Embedded Systems.

1 Introduction

Model-Based Design improves the design workflow of engineered sytems by employing computational models. In the embedded control systems realm, these models often are designed using Simulink® [20]. An embedded control system typically consists of a controller and a plant, where the plant is a physical system that is controlled to operate according to desired behavior.

The elements of Model-Based Design, illustrated in Fig. 1, can be summarized as:

- Executable specifications from models allow immediate feedback on the behavior of a specification, as opposed to documented behavior that often is misinterpreted.
- Design with simulation supports a faster exploration of the design space as opposed to constructing physical prototypes.
- Automatic code generation reduces the tedious and error-prone process of translating a design into a specification for the software engineers and manually writing the corresponding computer code.
- Test and verification can be performed in a much earlier stage in the design as a computational model is available with access to all internal variables, including those that may be difficult to obtain on a physical prototype.

A. Saad et al. (Eds.): Soft Computing in Industrial Applications, ASC 39, pp. 1–16, 2007.
springerlink.com

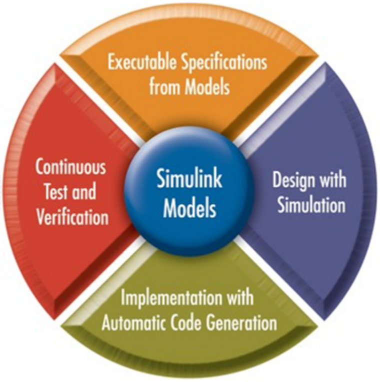

Fig. 1. Model-Based Design elements leverage Simulink® models

The adoption of Model-Based Design has enterprise-wide implications [1]. For example, the extensive use of models throughout the design process has created the desire to facilitate model reuse. This reuse, in turn, requires design tools that support the exchange of models between engineering teams. For example, to obtain a high-fidelity plant model, a SolidWorks [21] computer-aided design (CAD) model of the geometry can be exported into a SimMechanics [18, 25] multibody model of the dynamics. Thus, modeling effort is reused as models are shared across teams.

A controller model may initially be a discrete state based model that is then extended to include implementation effects such the validation of input data. This approach implies that an execution engine for a supporting tool set such as Simulink and Stateflow® [22] has to efficiently handle both a data driven approach as well as an event driven approach. Because of the widely differing execution semantics that different models may employ, execution engines are required to be versatile and powerful such that efficient algorithms, tailored to the needs of a specific model, can be invoked.

An important distinction in execution semantics can be made between those that require continuity of variables, possibly in higher derivatives, and those that allow discrete changes. Combining those two execution semantics results in *hybrid dynamic systems*, or *hybrid systems* for short (e.g., [2, 10, 23]).

The modeling formalisms that capture the discrete part of a hybrid system often are state transition diagrams [9], for example, a high-level language such as statecharts [7] may be employed. Statecharts are state transition diagrams that include language features such as hierarchy, parallelism, and event broadcasting.

The modeling formalisms that capture the continuous part of a hybrid system often are designed for plant modeling, i.e., the modeling of physics [5, 8], and they typically rely on differential equations, possibly combined with algebraic constraints.

The combination of state transition diagrams and differential and algebraic equations may be desired if, for example, there are widelydiffering time scales at which physical phenomena occur. In such a situation, it may be beneficial to abstract fast continuous behavior into a discrete change. The slower continuous behavior is then modeled by differential equations, while the discrete behavior may be modeled by a state transition diagram [12].

For example, a nonelastic collision between two bodies can be modeled in detail by accounting for dissipation effects that occur from when the bodies initiate contact to when they achieve the same velocity. Alternatively, detailed behavior from the dissipative effects can be disregarded, and the velocities can be instantaneously set to be equal.

This paper provides the elements that constitute a hybrid dynamic system. Complications and idiosyncrasies in the behavior of such hybrid dynamic systems and an ontology of mode transition behavior are presented. It is illustrated how instantaneous changes in variables, in combination with the inequalities that define mode switching, can lead to rich and complex mode transition behavior [13].

Section 2 provides a more detailed introduction to Model-Based Design. Section 3 illustrates the use of Model-Based Design for a power window control system, which concretely shows a number of different modeling formalisms that are employed throughout the design. Section 4 introduces the underlying computational representation across different modeling formalisms as a hybrid dynamic system and discusses the characteristics of such a system. Section 5 presents the conclusions of this work.

2 Model-Based Design

The benefits of Model-Based Design are manifold and mostly stem from the use of computational technologies. In addition, rather than isolated usages of computational models, it is important that a tool infrastructure is available to move a model through the design stages while elaborating it along the way.

2.1 Why Model-Based Design?

Model-Based Design uses an executable specification, which facilitates communication across engineering groups and enables rapid design iterations which greatly decreases development time. This approach contrasts with a more traditional approach in which the specification typically consists of a paper document. The document needs to be shared among many engineers or groups of engineers, and is often miscommunicated or distributed copies are not kept up to date.

The model that results from an executable specification is not only the repository for all of the information about the concept and design but also the design implementation. Once the specification has been made executable, Model-Based Design enables the exploitation of simulation so that the design space can be searched for an optimal design efficiently. Moreover, this search may now be automated.

Following simulation, implementation is achieved through automatic code generation. Transforming a paper specification of a design into software such as C-code is an error-prone process. Automatic code generation can reduce both design and hand-coding errors while substantially alleviating the tediousness of the coding task.

Model-Based Design further enables unambiguous communication between everyone involved in the overall design, within one company and across companies, such as between suppliers and the original equipment manufacturer (OEM). When everyone works off the same model, or at least an elaborated form of a core model, they can speak the same language and communicate more effectively.

Another key benefit of Model-Based Design is early test and verification. If a model is available early on in the design process, and it is executable, it is possible to design the tests to ensure that the final product complies with the original requirements based on the model. Therefore, design testing can be performed early on in the design process, as opposed to having to wait until the physical product has become available.

As a result, Model-Based Design eliminates the need for physical prototypes in the early design phases. Their use can be deferred much longer than in a traditional design approach, which decreases the reworking of a prototype because it has already been tested in much greater detail in a computational setting.

2.2 Practicing Model-Based Design

Model-Based Design relies heavily on model elaboration, as shown in Fig. 2. On the left of the diagram is the core control algorithm, which is often designed using synthesis techniques based on simplified plant models, such as low-order linear versions of more complex plant models.

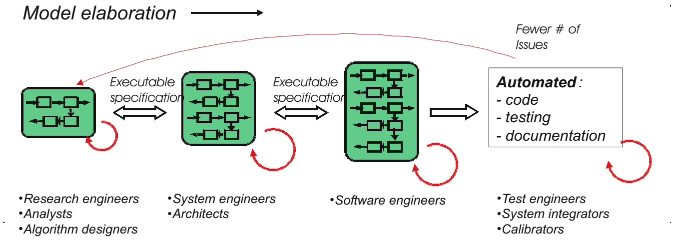

Fig. 2. Model elaboration

Once the core control algorithm has been derived, it is handed to the system engineers who embed it into an overall system. At this point, data validation, input/output (I/O) functionality, redundancy management, and testing functionality will be included.

The next step is implementation, in which the control algorithm needs to be coded in C, Ada, or any other desired target language, to embed the control algorithm into a physical environment as software that executes on a hardware target. This step is typically done by software engineers. Operating system issues may arise here; for example, computations that have been designed for the algorithm as well as for the system must fit into the computational resources available. The algorithm may need to fit onto a number of microprocessors; there may be high priority tasks, low priority tasks, and different sample rates, which are all coded into tasks or multiple tasks; and it is necessary to verify and validate that the system still operates according to specification.

Finally, the system must be integrated with other systems that have been built. This requirement leads to the notion of "systems of systems." Using an automobile power window as an example, it may be necessary to validate that the window operates properly in concert with the electrical system by not drawing electrical power when the engine is started. This is achieved by combining and integrating the system of systems, as well as calibrating it to make sure that it operates properly.

Model elaboration, then, is the process of moving the model through a number of phases where increasing detail is included. This facilitates communication between the engineering teams responsible for the separate phases. As mentioned previously, data validation and analysis need to be performed, I/O and interfaces need to be established, and redundancy management all need to be included in the design. With Model-Based Design–and its use of executable models–testing happens every time a model is simulated, and thus is an integrated aspect of the design process. This integration enables continuous testing and validation that the model satisfies the requirements and is working according to specifications.

3 A Power Window

To provide a concrete example of the use of Model-Based Design, the design of a power window (see Fig. 3) is outlined. The power window is an example of Model-Based Design for embedded control system development from concept through to implementation. It illustrates the use of different modeling formalisms that have different models of computation, the combination of which results in a hybrid dynamic system.

3.1 System Requirements

Electronics are used in automobiles to control various functions such as the opening and closing of windows and sun-roof, adjusting the mirrors/headlights, and locking and unlocking the doors. These systems are subject to stringent operating constraints, as failure may result in dangerous and possibly life-threatening situations. Therefore, careful design and analysis is mandatory before deployment.

Some quantitative requirements for the control of a power window may be as follows:

- The window must be fully opened and closed within 4 s.
- If the down or up command is issued for at least 200 ms and at most 1 s the window has to be fully opened or closed, respectively (auto-up/auto-down).