Deutsches Zentrum
für Luft- und Raumfahrt e.V.

Bericht DLR-IB-515-01-02

Higher Intelligence in Embedded Systems Design and Operation: A research and development proposal

Pieter J. Mosterman

Institute of Robotics and
Mechatronics
Oberpfaffenhofen

DLR

Institut für: Robotik und Mechatronik (RM)

Bl.:

IB.Nr.: *515-01-02*

Higher Intelligence in Embedded Systems Design and Operation: A research and development proposal

IB 515-01-02

Freigabe: **Die Bearbeiter:**

Unterschriften:

Pieter J. Mosterman

Der Abteilungsleiter:

Dr.-Ing. J. Bals

Der Institutsdirektor:

Prof. Dr.-Ing. G.Hirzinger

Der Institutsdirektor:

Dieser Bericht enthält:

12 **Blatt davon**

Bilder

Diagramme

Ort: OP	Datum: 27. Juni 2001	Bearbeiter: ja	Zeichen:

Higher Intelligence in Embedded Systems Design and Operation: A research and development proposal

Pieter J. Mosterman

Institute of Robotics and
Mechatronics
Oberpfaffenhofen

12	Seiten
0	Bilder
0	Tabellen
67	Literaturstellen

Contents

List of Figures

List of Tables

Higher Intelligence in Embedded Systems Design and Operation: A research and development proposal

Deutsches Zentrum
für Luft- und Raumfahrt e.V.

Abstract

Models are used in systems engineering for knowledge representation, structured analysis, and to bridge conceptual differences between domains, e.g., to facilitate hardware/software co-design. Additionally, with the advent of ubiquitous computing, model-based applications, e.g., in control, diagnosis, and maintenance, will become pervasive and ultimately become as proliferated as embedded computing power. To handle the multitude of formalisms for model design and analysis and to combine and relate these, the emerging field of computer automated multi-paradigm modeling relies on the notion of meta-modeling, i.e., modeling the model. When extended with sophisticated model transformation facilities, the multi-paradigm modeling notions can thus be exploited further to facilitate a suite of technologies and applications that implement a form of *higher intelligence.*

1 Introduction

The use of models has found widespread application in systems engineering as a (semi-)formal method to manage the complexity and heterogeneity of large scale systems and their design teams and tools because they are amenable to analysis and synthesis tasks. For example, models are used for knowledge representation, requirements engineering [53, 67], structured analysis [25, 65], to manage complexity and achieve high quality of engineered systems [66], to handle the heterogeneous nature of embedded systems [22], as a high level programming method [24, 23, 59], and to bridge conceptual differences between domains [10, 58]. With the advent of ubiquitous computing, model-based applications, e.g., in control, diagnosis, and maintenance, will become pervasive and ultimately become as proliferated as embedded computing power [13, 40].

To avoid overspecification and attain optimal performance, new design paradigms based on holistic views (e.g., mechatronics [60, 61]) are a necessity to analyze subtle interaction between information processing components and the physical environment as well as between the different design tasks. This requires tight integration of the separate individual design activities. However, each of the engineering disciplines involved in system design and operation have developed domain and problem specific (often proprietary) formalisms that match their needs optimally but complicate the integration process. The goal of this research is to develop and prototype a core of next generation multi-paradigm modeling[1] methods and technologies that address this incompatibility and enable the development of novel applications. This is a powerful approach that allows the generation (instantiation) of domain and problem specific methods, formalisms, and tools and because of a common meta language, these different instances can be integrated by combination, layering, heterogeneous refinement, and multiple views [19, 32, 38, 64]. At present this is still very much a topic of on-going research that breaks down into two types of activities: (i) heterogeneous modeling [15, 30, 34, 52] and formalism [6] and tool coupling [16, 17], and (ii) behavior generation [7, 31, 41, 51, 55, 62]. The first is mainly concerned with the symbiotics (symbols, syntax, and static semantics) of modeling formalisms, whereas the second addresses analysis and behavior generation using the dynamic semantics of such heterogeneous models, in general this behavior is of a mixed continuous/discrete, i.e., *hybrid*, nature. Important issues include but are not limited to the design of system engineering ontologies [8], integrated development environment design, heterogeneous

---

[1] The special sessions on computer automated multi-paradigm modeling at the IEEE Symposium on Computer Aided Control System Design in September 2000 present a good overview of the state-of-the-art in this field. See http://www-er.df.op.dlr.de/kondisk/campam.html for more information.

execution models [39], hybrid dynamic systems,[2] code synthesis (software and hardware description language) [27, 35, 26], and formal methods [12].

Three orthogonal dimensions of multi-paradigm modeling are *multi-abstraction* modeling, concerned with the relationship between models at different levels of abstraction, possibly described in different formalisms, *multi-formalism modeling*, concerned with the coupling of and transformation between models described in different formalisms, and *meta-modeling*, concerned with the description of model representations and instantiation of domain specific formalisms [50]. When extended with sophisticated model transformation facilities, the multi-paradigm modeling notions can be exploited to facilitate a suite of technologies and applications that manipulate a model into a different representation, possibly changing the abstraction, partitioning, and hierarchical structure to render it suitable for particular tasks, i.e., it is operated on the model rather than its generated information.

Though some model transformation schemes exist within [14, 29, 36, 63] and between formalisms [4, 5, 57, 62], a hiatus in this multi-paradigm modeling effort is the prevalent need to manually design models in different representations for analyses, consistency checks, and execution. The model transformations that are available and current development efforts tend to focus on the goal of system realization from design (e.g., automatic code synthesis) while models embody knowledge, and as such they also form the core of intelligent applications (e.g., model-predictive control [1], model-based diagnosis [11, 28, 42, 44, 45, 48], and self maintenance). When extended with sophisticated model transformation facilities, the multi-paradigm modeling notions can thus be exploited further to facilitate a suite of technologies and applications that implement a form of *higher intelligence*: Where present intelligent applications utilize a formal representation of some form of a process or system to derive information about its state and predict future behavior, higher intelligence manipulates this model into a different representation, possibly changing the abstraction, partitioning, and hierarchical structure to render it suitable for required tasks, i.e., it operates on the model rather than its generated information.

---

[2] See, e.g., web-site of the Virtual Action Group on Hybrid Dynamic Systems of the IEEE Computer Aided Control System Design Technical Committee at http://www-er.df.op.dlr.de/cacsd/hds/index.shtml.

2 The Proposal

It is proposed to support a coherent pervasive model-based technology paradigm that addresses three branches of research to be pursued to facilitate higher intelligence in embedded systems design and operation: (i) model manipulation, e.g., for reuse, (ii) cross correlating models, and (iii) exploiting and further developing model-based technologies.

2.1 Model Complexity Transformation

It is investigated how to systematically derive models of different complexity. These may be simplified models in the same formalism [54] but also more abstract models in a different representation [49]. This methodology can be applied, e.g.,

- to perform optimization with increasingly complex models, which may be more likely to find a global optimum [21],
- to allow vendor models (destined to replace electronic data sheets) to be combined into one extensive and complex *base model* that can be used in a reduced form for the different design and operation tasks (e.g., control design, performance assessment, and model-based diagnosis) [43],
- to design intelligent numerical solvers that adapt the complexity of the model to the efficiency requirements (e.g., real-time simulation constraints) [33],
- to support reactive learning environments (so called *microworlds*) by increasingly adding detail to the world model [47], and
- to infer the required level of detail of model parts in different representations to ensure consistency of analysis results of the overall combined model against given criteria.

Note that in general it may be possible to automatically add model detail as well as to automatically reduce complexity of a base model [9].

2.2 Model Representation Transformation

Another part of the research investigates transforming a model representation based on the specification of its initial and target formalism [23]. This allows one to:

- Generate a functional model from software or even execution trace (e.g., a solver procedure can be synthesized from the concepts that are part of the domain specific ontology, i.e., function calls, and their respective execution ordering);
- Automate generation of different views on a system (e.g., scenario diagrams from a functional model) or even an implementation model when translating to a domain specific formalism;
- Automate design by generating specifications from requirements, ultimately leading to automated code synthesis (or at least stub generation), which, in turn, can be integrated in an automated optimization and run-time architecture reconfiguration scheme for hardware and software, or software only (e.g., for System-on-Chip applications [56]);
- Automatically derive a reconfiguration model for guiding run-time system changes from functional and architectural models [3, 37];
- Use best-of-class methods and tools by generating the required data and model representation format to prevent inconsistencies at the boundaries between engineering teams, engineering software, and multiple modeling paradigms, and to enable the sharing and coordinating of information flow with minimal overhead [18].

2.3 Formalism Modeling

The third branch of research investigates the theory and application of meta-modeling [2, 20], the enabling technology for (i) the design of tailored formalisms and tools by constituting an infinitely fine grained spectrum of formalisms, (ii) the use of domain specific formalisms and tools to facilitate high level model-based programming, and (iii) finding analogies, similarities, and differences between models of different system views and aspects.

2.4 Model Execution

At the execution level, the dynamic semantics of such systems combine continuous with discrete behavior to form so-called hybrid dynamic systems. These require dedicated tools and algorithms to handle a number of idiosynchracies in behavior generation such as event detection and location, chattering, and consistent initial value computation [46]. This research will investigate and advance state of the art in the hybrid dynamic systems field by addressing (i) sequences of discrete transitions, (ii) consistent semantics of hybrid dynamic systems formalisms, (iii) sensitivity to initial conditions, (iv) sliding mode behavior, and (v) hybrid models for diagnosis and to design observers.

3 Deliverables

The deliverables of the research will be a meta-modeling tool to facilitate the model transformation research activities and a collection of application modules. The tool will be designed to generate tailored formalisms and integrated development environments. One or more transformation formalisms will be developed that allow model transformation all the way to an executable specification. This tool forms the basis for additional functionality based on artificial intelligence techniques to perform model complexity and representation transformation tasks. In turn, this will be coupled to software modules that implement the application tasks such as optimization and simulation. Apart from the research aspects involved in this, the specific applications of the inter- and intra-formalism model transformation will be further developed and investigated.

Bibliography

[1] Karl J. Åström and Björn Wittenmark. *Computer Controlled Systems: Theory and Design.* Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[2] Colin Atkinson. Metamodeling for Distributed Object Environments. In *First International Enterprise Distributed Object Computing Workshop (EDOC'97)*, pages 90–101, Brisbane, Australia, 1997.

[3] Ted Bapty, Sandeep Neema, Jason Scott, Janos Sztipanovits, and Sameh Asaad. Model-Integrated Tools for the Design of Dynamically Reconfigurable Systems. Technical Report ISIS-99-01, Vanderbilt University, Nashville, TN, 2000.

[4] Luciano Baresi, Alessandro Orso, and Mauro Pezz'e. Introducing Formal Specification Methods in Industrial Practice. In *19th International Conference on Software Engineering (ICSE'97)*, pages 56–66, Boston, MA, May 1997.

[5] Luciano Baresi and Mauro Pezz'e. On Formalizing UML with High-Level Petri Nets. In F. De Cindio and G. Agha, editors, *Concurrent Object-Oriented Programming and Petri Nets*, pages 271–300. Springer-Verlag, Berlin, 1999.

[6] Fernando J. Barros. Modeling formalisms for dynamic structure systems. *ACM Transactions on Modeling and Computer Simulation*, 7(4):501–515, 1997.

[7] Paul I. Barton. Modeling, Simulation, and Sensitivity Analysis of Hybrid Systems: Mathematical Foundations, Numerical Solutions, and Sofware Implementations. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 117–122, Anchorage, Alaska, September 2000.

[8] Willem Nico Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse.* PhD dissertation, University of Twente, The Netherlands, 1997.

[9] Arno P. J. Breunese, Theo J. A. de Vries, Job van Amerongen, and Peter C. Breedveld. Maximizing Impact of Automation on Modeling and Design. In *ASME Dynamic Systems & Control Div. '95*, pages 421–430, San Francisco, CA, 1995.

[10] Jan F. Broenink and Gerald H. Hilderink. A structured approach to embedded control systems implementation. In *Proceedings of the IEEE International Conference on Control Applications*, Mexico City, Mexico, September 2001.

[11] R.N. Clark, P.M. Frank, and R.J. Patton. Introduction. In Ron Patton, Paul Frank, and Robert Clark, editors, *Fault Diagnosis in Dynamic Systems: Theory and Applications*, chapter 1, pages 1–19. Prentice-Hall, UK, 1989.

[12] Edmund M. Clarke and Jeannette M. Wing. Formal Methods: State of the Art and Future Directions. *ACM Computing Surveys*, 28(4):626–643, December 1996. Report by the Working Group on Formal Methods for the ACM Workshop on Strategic Directions in Computing Research.

[13] Rance Cleaveland, John Anton, Jim Krause, Bruce Krogh, Harold Mortazavian, and Feng Zhao. DARPA-ITO workshop on software enabled control. http://www.dyncorp-is.com/darpa/meetings/sw/software-enabled.html, December 1997.

[14] G. Dauphin-Tanguy, P. Borne, and M. Lebrun. Order reduction of multi-time scale systems using bond graphs, the reciprocal system and the singular perturbation method. *Journal of the Franklin Institute*, 319(1/2):157–171, 1985.

[15] Eric Engstrom and Jonathan Krueger. A Meta-Modeler's Job is Never Done: Building and Evolving Domain-Specific Tools With DOME. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 83–88, Anchorage, Alaska, September 2000.

[16] Johannes Ernst. Data interoperability between CACSD and CASE tools using the CDIF family of standards. In *Proceedings of the 1996 International Symposium on Computer Aided Control System Design*, pages 346–351, Dearborn, MI, September 1996.

[17] Johannes Ernst and Scott Washburn. Zero-latency Engineering^{TM} for Control Design. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 71–76, Anchorage, Alaska, September 2000.

[18] Michael Fisher. Zero-latency engineering^{tm}. Aviatis Corp., White Paper, 1999.

[19] Paul A. Fishwick. Heterogeneous decomposition and inter-level coupling for combined modeling. In *1991 Winter Simulation Conference*, pages 1120–1128, Phoenix, AZ, 1991.

[20] R. Geisler, M. Klar, and C. Pons. Dimensions and Dichotomy in Metamodeling. Technical Report 98-5, TU Berlin, Berlin, Germany, 1998.

[21] Andrew Gelsey, Mark Schwabacher, and Don Smith. Using modeling knowledge to guide design space search. *Artificial Intelligence*, 101:35–62, 1998.

[22] Alain Girault, Bilung Lee, and Edward A. Lee. Hierarchical Finite State Machines with Multiple Concurrency Models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(6):742–760, June 1999.

[23] David Harel. From play-in scenarios to code: An achievable dream. *Computer*, 34(1):53–60, January 2001.

[24] David Harel and Eran Gery. Executable object modeling with statecharts. *Computer*, pages 31–42, July 1997.

[25] Derek J. Hatley and Imtiaz Pirbhai. *Strategies for Real-Time Systems Specification*. Dorset House Publishing Co., New York, New York, 1988.

[26] Robert Höpler and Pieter J. Mosterman. Model Integrated Computing in Robot Control to Synthesize Real-time Embedded Code. In *Proceedings of the IEEE International Conference on Control Applications*, Mexico City, Mexico, September 2001.

[27] Christian Hote, Alain Roan, and Florence Flores-Fahrnow. Case study: A simple altitude management system with SCADE. Technical report, CS Verilog, March 1999.

[28] Rolf Isermann. A review on detection and diagnosis illustrate that process faults can be detected when based on the estimation of unmeasurable process parameters and state variables. *Automatica: IFAC Journal*, 20(4):387–404, 1989.

[29] Yumi Iwasaki and Inderpal Bhandari. Formal basis for commonsense abstraction of dynamic systems. In *AAAI-88*, pages 307–312, 1988.

[30] John James. Thoughts on Information Operation Detection as a Nonlinear, Mixed-Signal Identification Problem: A Control Systems View. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 77–82, Anchorage, Alaska, September 2000.

[31] Karl Henrik Johansson, John Lygeros, Jun Zhang, and Shankar Sastry. Hybrid Automata: A formal paradigm for heterogeneous modeling. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 123–128, Anchorage, Alaska, September 2000.

[32] M. Jourdan, F. Lagnier, F. Maraninchi, and P. Raymond. A multiparadigm language for reactive systems. In *IEEE International Conference on Computer Languages (ICCL)*, Toulouse, France, 1994.

[33] M. S. Kamel, K. S. Ma, and W. H. Enright. ODEXPERT - an expert system to select numerical solvers for initial value ODE systems. *ACM Transactions on Mathematical Software*, 19(1):44–62, 1993.

[34] Gabor Karsai, Greg Nordstrom, Akos Ledeczi, and Janos Sztipanovits. Specifying Graphical Modeling Systems Using Constraint-based Metamodels. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 89–94, Anchorage, Alaska, September 2000.

[35] Gabor Karsai, Janos Sztipanovits, and Hubertus Franke. Towards Specification of Program Synthesis in Model-Integrated Computing. In *Proceedings of the ECBS-98*, pages 226–233, Jerusalem, Israel, 1998.

[36] Petar V. Kokotović, Hassan K. Khalil, and John O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design.* Academic Press, London, 1986. ISBN 0-12-417635-6.

[37] Sanjaya Kumar, Devesh Bhatt, Steve Vestal, Bill Wren, John Shackleton, Hazel Shirley, Rashmi Bhatt, John Golusky, Mark Vojta, John Fischer, Steve Crago, Brian Schott, Robert Parker, and Gary Gardner. ADAPTERS. In *2nd annual Military and Aerospace Applications of Programmable Devices and Technologies Conference*, Laurel, MD, September 1999.

[38] Akos Ledeczi, Greg Nordstrom, Gabor Karsai, Peter Volgyesi, and Miklos Maroti. On Metamodel Composition. In *Proceedings of the IEEE International Conference on Control Applications*, Mexico City, Mexico, September 2001.

[39] Edward A. Lee. Embedded Software – An Agenda for Research. Technical Report M99/63, Department of EECS, University of California, Berkeley, CA 94720, 1999.

[40] Edward A. Lee. What's Ahead for Embedded Software. *Computer*, 33(9):18–26, September 2000.

[41] Jie Liu and Edward A. Lee. Component-Based Hierarchical Modeling of Systems with Continuous and Discrete Dynamics. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 95–100, Anchorage, Alaska, September 2000.

[42] Jan Lunze. Diagnosis of quantised systems by means of timed discrete-event representations. In Nancy Lynch and Bruce H. Krogh, editors, *Lecture Notes in Computer Science, Hybrid Systems: Computation and Control*, pages 258–271, Berlin, 2000. Springer-Verlag.

[43] Heřman Mann. A versatile modeling and simulation tool for mechatronics control system development. In *1996 IEEE Symposium on Computer Aided Control System Design*, pages 524–529, Dearborn, 1996.

[44] A. Misra, J. Sztipanovits, and R. Carnes. Robust diagnostic system: Structual redundancy approach. In *Proceedings of the SPIE's International Symposium on Knowledge-Based Artificial Intelligence Systems in Aerospace and Industry*, pages 249–260, Orlando, FL, April 1994.

[45] Amit Misra. *Sensor-Based Diagnosis of Dynamical Systems*. PhD dissertation, Vanderbilt University, Nashville, TN, 1994.

[46] Pieter J. Mosterman. An Overview of Hybrid Simulation Phenomena and Their Support by Simulation Packages. In Frits W. Vaandrager and Jan H. van Schuppen, editors, *Hybrid Systems: Computation and Control*, volume 1569, pages 164–177. Lecture Notes in Computer Science; Springer-Verlag, March 1999.

[47] Pieter J. Mosterman. Towards Model Manipulation for Efficient and Effective Simulation and Instructional Methods. In *Distributed Modelling and Simulation of Complex Systems for Education, Training and Knowledge Capitalisation*, Eze, France, May 1999.

[48] Pieter J. Mosterman and Gautam Biswas. Diagnosis of continuous valued systems in transient operating regions. *IEEE Transactions on Systems, Man, and Cybernetics*, November 1999.

[49] Pieter J. Mosterman and Gautam Biswas. Towards Procedures for Systematically Deriving Hybrid Models of Complex Systems. In Nancy Lynch and Bruce Krogh, editors, *Hybrid Systems: Computation and Control*, pages 324–337, 2000. Lecture Notes in Computer Science.

[50] Pieter J. Mosterman and Hans Vangheluwe. ACM transactions on modeling and computer simulation, 2003. Special Issue on Computer Automated Multi-Paradigm Modeling.

[51] Simin Nadjm-Tehrani. Formal Methods for Analysis of Heterogeneous Models of Embedded Systems. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 141–146, Anchorage, Alaska, September 2000.

[52] Gregory G. Nordstrom. *Metamodeling – Rapid Design and Evoluion of Domain-Specific Modeling Environments*. PhD dissertation, Vanderbilt University, Electrical Engineering, May 1999.

[53] Bashar Nuseihbeh, Jeff Kramer, and Anthony Finkelstein. A framework for expressing the relationships between multiple views in requirements specification. *IEEE Transactions on Software Engineering*, 20(10):760–773, October 1994.

[54] George J. Pappas, Gerardo Lafferriere, and Shankar Sastry. Hierarchically consistent control systems. Technical Report UCB/ERL M98/16, University of California at Berkeley, Berkeley, California, 1998.

[55] Taeshin Park. Verification of Large-Scale Hybrid Systems Using Implicit Model Representation. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 135–140, Anchorage, Alaska, September 2000.

[56] Kenneth H. Peters. Migrating to single-chip systems. *Embedded Systems Programming*, 12(4):30–45, April 1999.

[57] Manuel A. Pereira Remelhe. Simulation and Visualization Support for User-defined Formalisms Using Meta-Modeling and Hierarchical Formalism Transformation. In *Proceedings of the IEEE International Conference on Control Applications*, Mexico City, Mexico, September 2001.

[58] J. Sztipanovits, G. Karsai, C. Biegl, T. Bapty, A. Ledeczi, and A. Misra. MULTIGRAPH: An Architecture for Model-Integrated Computing. In *Proceedings of the International Conference on Engineering of Complex Computer Systems (ICECCS'95)*, pages 361–368, Ft. Lauderdale, Florida, November 1995.

[59] Janos Sztipanovits, Gabor Karsai, and Hubertus Franke. Model-integrated program synthesis environment. In *IEEE Symposium on Engineering of Computer Based Systems*, Friedrichshafen, Germany, March 1996.

[60] Job van Amerongen. Mechatronic Design. In *The 7th Mechatronics Forum International Conference*, Atlanta, GA, August 2000. plenary paper.

[61] Job van Amerongen. The Role of Control in Mechatronics. *Engineering Science and Educational Journal*, 9(3):105–112, 2000.

[62] Hans Vangheluwe. DEVS As a Common Denominator for Multi-formalism Hybrid System Modeling. In *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*, pages 129–134, Anchorage, Alaska, September 2000.

[63] Andreas Varga. On modal techniques for model reduction. Technical Report TR R136-93, Institute of Robotics and System Dynamics, DLR Oberpfaffenhofen, P.O. Box 1116, D-82230 Wessling, Germany, 1993.

[64] Steve Vestal. Software Architecture Workshop, July 1994.

[65] Paul T. Ward and Stephen J. Mellor. *Structured Development for Real-Time Systems.* Prentice-Hall, Englewood Cliffs, New Jersey, 1985.

[66] K.C.J. Wijbrans. *Twente Hierarchical Embedded Systems Implementation by Simulation: a structured method for controller realization.* PhD dissertation, University of Twente, Enschede, The Netherlands, 1993. ISBN 90-9005933-4.

[67] Pamela Zave and Michael Jackson. Where do operations come from? a multiparadigm specification technique. *IEEE Transactions on Software Engineering,* 22(7):508–528, July 1996.