Preprints of the 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, Russia, June 3 - 5, 2009

One Decade of IEC 61499 Modeling and Verification - Results and Open Issues

Hans-Michael Hanisch * **Martin Hirsch** * **Dirk Missal** *
Sebastian Preuße * **Christian Gerber** *

* *University of Halle-Wittenberg, Germany*
Institute of Computer Science, Chair for Automation Technology
Email: hans-michael.hanisch@informatik.uni-halle.de

Abstract: The contribution summarizes the development of formal modeling and verification of Function Blocks following the IEC 61499. We provide a critical review on what has been done so far and open the view for further challenges in the development of formal techniques for IEC 61499.

1. INTRODUCTION

The work Vyatkin and Hanisch [1999] was to our best knowledge the first approach to deal with the IEC 61499 (that was just a draft at this time) in a formalized way. That was in 1999, ten years ago, and one of the authors of this contribution who presented the work in 1999 had the impression that almost nobody in the audience had any idea what the work was all about.
Fortunately, the situation has changed remarkably since that time. Unfortunately, a huge bag of unsolved problems and questions remains.
Therefore, the intention of this contribution is a revision of the work accomplished up to now and a projection of the upcoming issues in future.
This contribution is mainly based on the experiences that have been made in our group. It includes approaches of other groups, but only to the extent that conclusions can be drawn from that. It is therefore structured as follows.
Section 2 describes the general framework of a model-based controller design of IEC 61499 (IEC [2005]) and the results that are available. Section 3 provides an overview of existing tools and benchmark applications.
Section 4 discusses some open issues and gives a conclusion.

2. MODEL-BASED CONTROLLER DESIGN

The rising size and complexity of controlled systems as well as growing safety and quality requirements make high demands on controller design. Within the framework of computer aided engineering (CAE) the model-based controller design provides instruments to meet the modern demands. Figure 1 shows the general framework for model-based controller design as it is commonly used in control engineering. One sees that generally there are two ways to get a correct controller. The most elegant one is synthesis. For this, we take a model of the uncontrolled plant behavior and a formal specification of the desired behavior of the plant. By applying a synthesis algorithm that is proven to be correct, we generate a model of the controller that would fulfill the specifications in closed loop. Since the model of the controller is not the implementation of the controller, we need to generate the control code from the controller model. This should be done automatically.

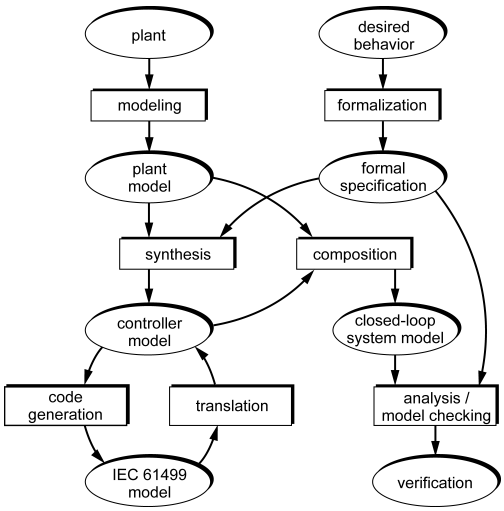

Fig. 1. Formal methods on controller design.

We will come back to this in Section 2.6. The other way is controller verification. For this, we need a controller design, a model of the uncontrolled plant behavior and a formal specification of the desired plant behavior under control. We have to generate a formal model of the controller, have to interconnect this model with the plant model to a model of the closed loop and prove the specification by means of model-checking. This may end up in iterations if the controller is not designed correctly.
In both cases we need models of the uncontrolled plant behavior and formal specifications. The models as well as the specifications must be designed by engineers who are experts in designing and operating manufacturing systems but by no means experts in formal methods. Applying such methods even to traditionally PLC-based control systems is by no means easy. Nonetheless, it is the keystone in future engineering methodologies for distributed control systems. We therefore describe in this section what has been done in this field for distributed control systems following the IEC 61499.

2.1 Plant Modeling

If control is seen decoupled from the object that has to be controlled, it is no longer control. It is then merely

information processing, in which way ever. It is therefore a domain for Computer Scientists that is called "embedded systems" or "reactive systems". Control, however, means always a closed-loop interaction of the controller with the object that is controlled (we call it plant).
From this point of view, the plant is the dominating part, and the controller itself is in any case a means to realize a behavior of the plant that is desired. That, in turn, means that the design of a controller without a minimum knowledge of the plant is a very doubtful activity. That does not mean that it is not a severe scientific problem, but related to control, its results can be only as good as the knowledge of the plant behavior.
We focus on approaches that use a formal plant model that is not only used for simulation but for formal verification. Obviously, designing a plant model is time consuming and therefore a cost factor that must not be underestimated. Designing a model from scratch is therefore not appropriate. A methodology of engineering models in a systematic way rather than designing them is required. Such methodologies are described in Vyatkin and Hanisch [2005], Missal and Hanisch [2007] for example. A modular and compositional approach is helpful although it is not the ultimate solution of all problems.
The formalism of Net Condition/Event Systems (NCES) has been developed in our group. It is often seen as "only" an extension of ordinary Petri nets by some kind of new arcs that connect transitions. From a formal point of view and after complete composition, this may be true.
The major point of interest, however, is not yet another enrichment of Petri net formalisms but a yet formal methodology that is tailored to the needs of engineers and allows real modularity, object orientation (to some extent), behavior encapsulation, block diagram orientation, a concept of signals as usual in control engineering and last but not least a way to engineer models rather than to design them from scratch.
While the modeling of plant equipment behavior can be eased through repeatedly using building blocks (e.g., for binary sensors, magnetic valves, cylinders etc.), the model of work piece properties is mostly process specific. Obviously, such properties must be included in the model since the goal of the whole manufacturing/production process is to perform stepwise changes of these properties, and a major part of the specifications are expressed in terms of properties of manufactured goods. Modeling of work piece properties in manufacturing is relatively simple, at least if compared with chemical engineering. In most manufacturing processes, work pieces are naturally discrete and do not have their own dynamic behavior. If we look to Batch processes in chemical engineering that have the strongest similarities with manufacturing, things are much more complicated. Manufactured goods are not naturally discrete, the batch size can be changed continuously, Batches can be mixed (Blending operations), can be separated (Filtration, Distillation etc.) and so on. Furthermore, the processed substances have their own dynamic behavior driven by the laws of chemical kinetics, thermodynamics etc. It seems to be surprising, but it is a fact, that such processes are quite well (to a certain extent) understood and controlled. The basis is the concept of Unit Operations that is about 100 years old. Nonetheless, it is the conceptual basis for the method of Recipe Control (Namur [1993], ISA [1995]) that is commonly accepted and applied in Batch process control.
The methodology as well as implementations (see for example Fisher [1990]) show some strong similarities with object-oriented approaches in manufacturing.
It is obvious that object orientation is a very helpful issue in plant modeling. It is clear that a plant is a composition of similar or even identical physical components that can be identified and modeled separately. It leads in a very natural way to the concept of Automation Objects.
However, the need of manual modeling of work piece properties remains.

2.2 Controller Modeling

In early work, we have modeled only the execution control of Function Block (abbr. FB) applications. The algorithms inside the FB's, however, change and transmit not only Boolean data but integer-valued data as well.
We therefore need transformation rules from the designed FB application to formal models for the execution control as well as for the algorithms. Because of the wide range of functions being available in IEC 61499 algorithm languages, that can only be done for a restricted subset. A first step is done for variable value operation for the Boolean and integer data type by defining such transformation rules shown in Gerber et al. [2008], where models for Boolean and integer-valued data processing algorithms can be automatically retrieved from Function Blocks. Currently, work is done on proving the correctness and implementation.
Beside the issue of algorithm modeling, the issue of execution control modeling includes some critical problems. The execution of FBs and in this case the scheduling of FBs inside a FB Network differs between different runtime implementations (Sünder et al. [2006]) because the IEC 61499 does not exactly define it. As any transformation from a semi- or non-formal description, as IEC 61499, to a formal one, all transformation rules depend on a formal interpretation of the non-formal source. As long as every event cycle is interrupted by a Service Interface Function Block (abbr. SIFB), there will be no problem and the results of the closed-loop verification correspond with the real controller behavior. But using event cycles from one FB through other or direct back to itself, the execution model of Ivanova-Vasileva et al. [2008] has to be adopted to the different runtimes. Obviously, a general verification method needs analysis for many different controller models, which is not manageable for an engineer in everyday's practice.
It is not surprising that also other researchers found this issue interesting. We therefore mention the work proposed a couple of years later in Čengić and Åkesson [2008]. A discussion about these issues could be fruitful, as soon as the authors would compare their approaches with our - earlier - done work.

2.3 Composition

Plant and controller models have to be interconnected to closed-loop systems. In reality, plant and controller are interconnected via sensor/actuator signals that carry Boolean or integer-valued data. Models should describe this explicitly. If an interested engineering person has to

examine the interaction among controllers and plant, she or he should see it explicitly in the model.
Plant and controllers do not exchange any tokens (as it is assumed in the Petri net community), and they do not share common variables or events (as the automata community assumes). They are interconnected via signals that must be physically represented, transmitted etc. In general that signals do not provide the complete state information of the plant to the controller. NCES provide very easy means to compose plant and controller models by interconnecting them by condition signals and event signals.
Figure 2 shows the general scheme of signal interactions that is assumed. It is common to any control engineer, and it is the standard paradigm of automatic control that has been shown to be appropriate since decades. Hence, the authors do not see any need to change it.

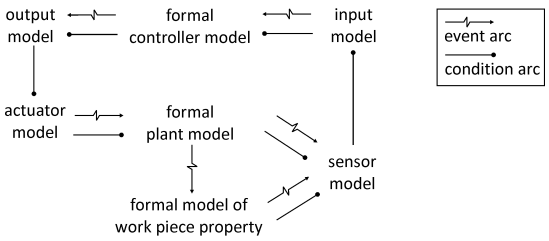

Fig. 2. Closed loop model scheme.

2.4 Specification

Designing a technical plant starts with the description of the desired plant behavior. Based on this specification, the plant is constructed and the control software is implemented. Previously, it was described, how the closed-loop model of plant and controller can be developed. The point of interest is now, how to obtain a formal and well-defined specification, which is free from misunderstandings. This requirement is vitally important, if the closed-loop model shall be verified in connection with the specification by the use of a model-checking tool. A lot of work groups deal with the specification of behavior of IEC 61499 control systems. For example, the executing behavior of Function Blocks is modeled with timed automata in Khalgui et al. [2004] or with activity diagrams of the Unified Modeling Language (abbr. UML) in Panjaitan and Frey [2005]. Certainly, we focus on specification of plant behavior, namely safety and production requirements. In the following, the main ideas of different approaches concerning this aspect are summarized.
In Vyatkin et al. [2000] it is described how a specification of plant behavior could be formulated with the Timed Computation Tree Logic (abbr. TCTL). This temporal logic allows combining the qualitative temporal assertions together with real-time constraints, what is important for a specification of dynamic behavior. To support the user in creating such a description of plant behavior, a graphical specification method was proposed in Hanisch and Vyatkin [2001]. Here, Timing Diagrams (abbr. TD) are used to specify production sequences. A Timing Diagram Editor (abbr. TDE) was implemented in Bouzon et al. [2005] to translate the diagrams to TCTL formulas. This editor is part of the Verification Environment for Distributed Applications (abbr. VEDA) (Vyatkin and Hanisch [2003]), which works as a tool chain for the verification of distributed industrial controllers. Another graphical method for the specification of production sequences was proposed in Preuße and Hanisch [2008]. The authors use Symbolic Timing Diagrams (abbr. STD) to describe plant behavior in an intuitive way. For the translation of the diagrams to temporal logic formulas, the Symbolic Timing Diagram Editor was implemented. It derives Computation Tree Logic (abbr. CTL) formulas automatically, so that the engineer has not to be familiar with the complex theory.
The UML offers a further graphical specification method. In Missal et al. [2007] activity diagrams are used to specify production sequences. The Systems Modeling Language (abbr. SysML) is an extension based on UML and provides description methods that are important for modeling of a system. In Hirsch and Hanisch [2008] the application of the different diagram types and facts on simulation, which are provided by SysML, are described.
Normally, the specification comes from an engineer who is an expert in manufacturing but not in the field of formal modeling or model-checking. The concern of all approaches is to support a user in creation of a formal description of plant behavior. This specification can be used to provide a well-defined documentation but it is reasonable as well to be used for the verification of the closed-loop model.

2.5 Model-Checking

Model-checking is used to verify formal models of plant and controller automatically. For this purpose, a model of the closed-loop behavior of plant and controller(s) is analyzed in connection with a formal specification of the desired plant behavior. The procedure provides a mathematical proof of correctness because every reachable state is computed. In contrast to this, simulation only tests a cutout of possible scenarios. Rare but often critical cases are not considered, so that erratic behavior could happen suddenly. For this reason, verification is usually the only possibility to exclude these worst case scenarios.
For formal modeling, we prefer the usage of NCES. That kind of model allows verifying functional and temporal properties. A further advantage is that it corresponds to the modular thinking of engineers. The naturally distributed state formalism guarantees a higher efficiency of model-checking. I.e. combinations of concurrent processes, that are difficult to predict in advance, can be computed. The formal specification must be ready for computing, therefore we choose CTL to verify the system. Automatic translation of verbally formulated safety-requirements is possible (see Sec. 2.4).
However, it is a challenge even for modern computers to handle the calculation of all possible states, because the complexity rises exponentially the more complex models become. Anyway, model-checking is the only possibility to get an assertion about the mathematical correctness of a model.
With the models and the specification one can perform a reachability analysis using appropriate tools, e.g. Starke and Roch [September 2002]. There, malfunctions can be localized and the origin of a malfunction can be tracked.
To verify the system efficiently, it can be an advantage to do hierarchical level-by-level verification. That approach has been introduced in Missal et al. [2007]. The approach

basically is to first verify the tasks of a module (separate and application independent). Afterwards, the pre-verified task controllers can be arranged in various combinations. Then, one can verify the coordination of the system, without considering the detailed behavior model of the task-controllers of the modules.

2.6 Synthesis

Formal synthesis is based on the idea of automatic controller generation. It is a procedure proven to be correct to generate a control model based on a formal plant model and a formal specification of intended or/and forbidden system behavior. Code for controller software can be generated based on the resulting controller model. Synthesis approaches are described for different plant models as finite state machines (Su and Wonham [2004]), Petri nets (Iordache and Antsaklis [2006]) and variations as condition nets (Holloway et al. [2000]) and Net Condition/Event Systems (Missal and Hanisch [2008]).
There is some work on code generation for IEC 61131 languages from formal controller models. The authors are not aware of any results for generation of IEC 61499 function blocks from formal models so fare. In Missal [2007] an approach for synthesis of distributed safety controllers is presented which results in Basic Function Blocks according to IEC 61499. The used synthesis formalism from NCES plant behavior models and state predicate specifications of forbidden states to formal controller models is described in Missal and Hanisch [2006]. The work on code generation is still ongoing. Function Blocks for control components can be generated based on IEC 61499 system descriptions within such framework.

3. APPLICATION

3.1 Tools

For implementing IEC 61499 compliant applications, during the last decade a couple of tools have been publically made available. Thanks to those tools, the emerging standard became famous in the cosmos of researchers and, significantly to be mentioned, their students, just because the trial implementations made the entrance of the standard into the teaching area possible.
In the following, we would like to present a short survey of the most common tools that have been developed in the last years, without claiming completeness of the list.
First of all, one needs to mention the Function Block Development Kit (abbr. FBDK) implementation by J. H. Christensen [1]. That tool is widely used in research trial implementations, and also our group mainly uses that development environment. A lot of Service Interface Function Block implementations have been developed, so the tool can be used in combination with lots of different Java-based control devices. A short introduction can be found in Cai et al. [2003] and Hirsch et al. [2007].
The CORFU Engineering Support System by the workgroup of K. Thramboulidis [2] supports an UML-based design approach of IEC 61499 compliant system design. Also to mention is the O^{3}NEIDA workbench [3], which includes the concept of Automation Objects. Last but not least we list the 4DIAC development environment [4], developed at the Vienna University of Technology, which is a C-Code based variation. A remarkable industrial, commercial tool implementation is ISaGRAF. This software combines both IEC 61131 and IEC 61499 compliant implementation methods [5].
In the following section we briefly specify existing benchmarks of IEC 61499 compliant test bed implementations.

3.2 Existing Benchmarks

Application of the IEC 61499 Standard is not only confined to the industrial manufacturing sector. Although developed for that field, one also comes across example implementations from automobile comfort electronics sector (Hirsch et al. [2006]) and aviation electronics sector (Insaurralde et al. [2006]).
But basically the standard has been developed for industrial manufacturing systems, and basically the most benchmarks have been developed on that background, although they have not yet reached it completely.
Many workgroups have been dealing with IEC 61499 compliant test beds. Due to those activities, naturally a certain amount of benchmarks has been developed and commissioned. Thereby one can differentiate between pure academic test beds and on the other hand, rare industrial-like prototypes. We begin with the academic test beds.
Our group started implementing prototype benchmarks for IEC 61499 compliant controllers in 2003 (Cai et al. [2003]). Through the years, our methodologies advanced in many directions. Starting with systematic building of models of the plant for a sound visualization, also the controller design methodologies were further developed. Based on simple centralized controllers we developed systematic methodologies for implementing distributed controllers.
Controller design methods have been published by our group in Cai et al. [2003], Vyatkin et al. [2006], Hirsch et al. [2007] and Missal et al. [2007] (chronological). Above all, the issue of reusing controllers in flexible automation unavoidably led to the design of hierarchical control patterns. Current state of the art benchmarks of our group can be found on our site [6] [7]. Also the group of G. Frey developed so called functional mechatronic controllers (Panjaitan [2007]), where the design approach is similar to ours.
Further, we want to specify academic test beds from the University of Auckland and Vienna University of Technology. In a few years, the group of V. Vyatkin created an IEC 61499 compliant test bed laboratory [8]. Reconfiguration on the fly, reusability of distributed control components and verification of the control software are major points of interest in the research of this group, that actually originated in the work done in Halle. Furthermore, a prototype-industrial luggage conveyor system case study taken from an airport-vendor is to be mentioned (Vyatkin

[1] Holobloc FBDK. URL, http://www.holobloc.com, March 2009.
[2] Corfu ESS 1.0. URL, http://seg.ee.upatras.gr/corfu/dev/download.htm, March 2009.
[3] O^{3}NEIDA. URL, http://www.oooneida.org, March 2009.
[4] 4DIAC. URL, http://www.fordiac.org, March 2009.
[5] ICS Triplex ISAGRAPH. URL, http://www.isagraf.com, March 2009.
[6] EnAS Project Demonstrator. URL, http://aut.informatik.uni-halle.de/forschung/enas_demo, March 2009.
[7] IEC 61499 Festo test bed. URL, http://aut.informatik.uni-halle.de/forschung/testbed, March 2009.
[8] MITRA. URL, http://www.ece.auckland.ac.nz/∼vyatkin/mitra_lab.html, March 2009.

et al. [2007]).
The group of the Vienna University of Technology possesses a big lab, dealing with distributed control methods, among others the IEC 61499 standard [9] . Furthermore the group makes a point demonstrating future scenarios for application IEC 61499 (Baier et al. [2007]).
A rather industrial-like application of the standard can be found in Colla et al. [2006]. In contrast to IEC 61131 Frameworks, the authors recognized an effective high-level view of the application. Furthermore, the development process showed up to be more rapidly due to reusing and adapting existing Function Blocks.
Concluding this section, IEC 61499 has reached the threshold to be applied in industrial applications. If we take a realistic view on all the benchmarks, we cannot see an application to systems where real manufacturing is done. This point of view leads us to the critical remarks and the conclusion.

4. CONCLUSION

Despite the need for better runtime platforms and development tools, there are also other issues that may play an important role in future.
The use of IEC 61499, UML and other emerging technologies does not automatically generate good and flexible automation solutions. These technologies must be accompanied by engineering methodologies that would eventually enable more reliable, flexible and reconfigurable systems. Anyway, the initiative to develop such methodologies must come from manufacturing industries themselves. Universities, research centers etc. may help, but they do not have the power to impose the pressure to control systems vendors to develop new generations of control systems that would be able to satisfy the above mentioned requirements. As long as big manufacturing companies do not feel the urgent need to take advantage of the emerging technologies, research in academia will get stuck in playing around with benchmark examples and possibly some kind of "exotic" applications.
Recipe Control in Batch process engineering is an excellent example for control systems engineering. Batch process systems in multipurpose plants have always been extremely flexible and reconfigurable. Recipe control is an engineering concept that actually came from a consortium of big (German) companies that released just a recommendation (Namur [1993]) that eventually ended up in an International Standard (ISA [1995]).
Based on the economic power of their customers, almost all big providers of Process Control Systems realized means to fulfill this recommendation within a couple of years. Therefore, Recipe Control has become the unified approach for control design in Batch process engineering. Up to now, we cannot see something comparable in manufacturing.
Some of the concepts developed in academia, as for example, agent-based technologies, Automation Objects and other approaches seem to be appropriate for developing control software, but from the point of view of shop-floor requirements, they are a bit over-sophisticated. The reason is a mutual misunderstanding of experts in embedded systems (focused on software) and manufacturing engineers (focused not only on their manufacturing hardware but on the goal to provide products).
Another huge issue is the problem of migration. It is completely unrealistic to assume that a company will make a strict and complete transition from its traditional PLC-based automation concepts to new technologies. Therefore, heterogeneous systems - both in hardware and software - will appear and will exist a rather long time. This generates completely new questions of engineering, modeling, verification and operation of such systems.
Also the transition from traditional (sub-) systems to IEC 61499-based ones without changing the behavior of the control loops is an issue of highest significance. We also observe missing communication between researchers of different background (manufacturing control, computer science) that leads to misunderstanding or misinterpretation. How to cross this gap is an open question left to the decision of the reader.
Instead of promising too much, the community should ask itself how far we have gone, what has been accomplished and what needs to be done in nearer or further future.

[9] Odo Struger Laboratory, Vienna Technical University. URL, http://www.acin.tuwien.ac.at/forschung/Projekte/255/, March 2009.

ACKNOWLEDGEMENTS

This work has been supported by several research funding institutions in Germany. The authors would like to acknowledge the support of German Research Foundation, German Ministry for Education and Research, German Ministry for Economy and Technology and Alexander von Humboldt Foundation.

REFERENCES

IEC 61499 - 1, Function Blocks - Part 1 Architecture, International Standard, 2005.

T. Baier, J. Fritsche, G. Keintzel, D. Loy, R. Schranz, H. Steininger, T. Strasser, and C. Sünder. Future scenarios for application of downtimeless reconfiguration in industrial practice. In *International Conference on Industrial Informatics (INDIN)*, pages 1129–1134, Vienna, Austria, June 2007.

G. Bouzon, V. Vyatkin, and H.-M. Hanisch. Timing Diagram Specifications in Modular Modeling of Industrial Automation Systems. In *16th IFAC World Congress*, pages 208–221, Prague, Czech Republic, July 2005.

X. Cai, V. Vyatkin, and H.-M. Hanisch. Design and Implementation of a Prototype Control System According to IEC 61499. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 269–276, Lisbon, Portugal, 2003.

M. Colla, A. Brusaferri, and E. Carpanzano. Applying the IEC-61499 Model to the Shoe Manufacturing Sector. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1301–1308, Prague, Czech Republic, 2006.

T. G. Fisher. *Batch Control Systems: Design, Application, and Implementation*. ISA, 1990. ISBN 1556171315.

C. Gerber, I. Ivanova-Vasileva, and H.-M. Hanisch. A Data processing Model of IEC 61499 Function Blocks with Integer-Valued Data Types. In *Workshop on Intelligent Manufacturing Systems (IMS)*, pages 239–244, Szczecin, Poland, 2008.

H.-M. Hanisch and V. Vyatkin. Application of Visual Specifications for Verification of Distributed Controllers.

In *International Conference on Systems, Man and Cybernetics (SMC)*, pages 646–651, Tucson, Ariz, USA, October 2001.

M. Hirsch and H.-M. Hanisch. Systemspezifikation mit SysML für eine Fertigungstechnische Laboranlage. In *Entwurf komplexer Automatisierungssysteme (EKA)*, pages 23–34, Magdeburg, Germany, April 2008.

M. Hirsch, V. Vyatkin, and H.-M. Hanisch. IEC 61499 Function Blocks for Distributed Networked Embedded Applications. In *International Conference on Industrial Informatics (INDIN)*, pages 670–675, Singapore, August 2006.

M. Hirsch, C. Gerber, V. Vyatkin, and H.-M. Hanisch. Design and Implementation of Heterogeneous Distributed Controllers according to the IEC 61499 Standard - A Case Study. In *International Conference on Industrial Informatics (INDIN)*, Vienna, Austria, 2007.

L. E. Holloway, X. Guan, R. Sundaravadivelu, and J. Ashley Jr. Automated synthesis and composition of taskblocks for control of manufacturing systems. *Transactions on Systems, Man and Cybernetics (SMC)*, 30(5): 669–712, 2000.

C. C. Insaurralde, M. A. Seminario, J. F. Jimenez, and J. M. Giron-Sierra. IEC 61499 Model for Avionic Distributed Fuel Systems with Networked Embedded Holonic Controllers. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 388–396, Prague, Czech Republic, September 2006.

M. V. Iordache and P. J. Antsaklis. *Supervisory Control of Concurrent Systems: A Petri Net Structural Approach (Systems & Control: Foundations & Applications)*. Birkhauser, 2006. ISBN 0817643575.

ISA. S88 Batch Control Part 1: Models and Terminology, ISA-S88.01-1995, February 1995.

I. Ivanova-Vasileva, C. Gerber, and H.-M. Hanisch. Basics of Modelling IEC 61499 Function Blocks with Integer-Valued Data Types. In *Workshop on Intelligent Manufacturing Systems (IMS)*, pages 233–238, Szczecin, Poland, 2008.

M. Khalgui, X. Rebeuf, and F. Simonot-Lion. A behavior model for IEC 61499 function blocks. In *Third Workshop on Modelling of Objects, Components and Agents (MOCA)*, pages 71–88, Aarhus, Denmark, October 2004.

D. Missal. Synthese Verteilter Sicherheitssteuerungen. In *Beiträge zum 41. Regelungstechnischen Kolloquium Boppard*, 2007.

D. Missal and H.-M. Hanisch. Synthesis of Distributed Controllers by Means of a Monolithic Approach. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 356–363, Prague, Czech Republic, September 2006.

D. Missal and H.-M. Hanisch. Modular Plant Modelling for Distributed Control. In *International Conference on Systems, Man and Cybernetics (SMC)*, pages 3475–3480, Montréal, Canada, October 2007.

D. Missal and H.-M. Hanisch. Synthesis of Distributed Forcing/Locking Safety Controllers. In *Conference of the IEEE Industrial Electronics Society (IECON)*, pages 383–390, Orlando Fl., USA, 2008.

D. Missal, M. Hirsch, and H.-M. Hanisch. Hierarchical Distributed Controllers - Design and Verification. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 657–664, Patras, Greece, 2007.

Namur. NE33 - Requirements to be met by systems for recipe-based operations, 1993.

S. Panjaitan. *Development Process for Distributed Automation Systems based on Elementary Mechatronic Functions*. PhD thesis, Technical University of Kaiserslautern, Germany, November 2007.

S. Panjaitan and G. Frey. Functional Design for IEC 61499 Distributed Control Systems using UML Activity Diagrams. In *International Conference on Instrumentation, Communications and Information Technology (ICICI)*, pages 64–70, Bandung, Indonesia, August 2005.

S. Preuße and H.-M. Hanisch. Specification and Verification of Technical Plant Behavior with Symbolic Timing Diagrams. In *International Design and Test Workshop (IDT)*, pages 313–318, Monastir, Tunisia, December 2008.

Ch. Sünder, A. Zoitl, J. H. Christensen, V. Vyatkin, R. W. Brennan, A. Valentini, L. Ferrarini, Th. Strasser, J. L. Martinez-Lastra, and F. Auinger. Usability and Interoperability of IEC 61499 based distributed automation systems. In *International Conference on Industrial Informatics (INDIN)*, pages 31–37, Singapore, August 2006.

P.H. Starke and S. Roch. Analysing Signal-Net Systems. *Informatikberichte, Humboldt-Universität zu Berlin*, vol. 162, September 2002.

R. Su and W. M. Wonham. Supervisor Reduction for Discrete-Event Systems. *Discrete Event Dynamic Systems*, 14(1):31–53, 2004. ISSN 0924-6703.

G. Čengić and K. Åkesson. A Control Software Development Method Using IEC 61499 Function Blocks, Simulation and Formal Verification. In *17th IFAC World Congress*, pages 22–27, COEX, Korea, South, July 2008.

V. Vyatkin and H.-M. Hanisch. A Modeling Approach for Verification of IEC1499 Function Blocks Using Net Condition / Event Systems. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 261–269, Catalonia, Spain, October 1999.

V. Vyatkin and H.-M. Hanisch. Verification of distributed control systems in intelligent manufacturing. In *Journal of Intelligent Manufacturing*, pages 123–136, 2003.

V. Vyatkin and H.-M. Hanisch. Reuse of Components in Formal Modelling and Verification of Distributed Control Systems. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 129–134, Catania, Italy, September 2005.

V. Vyatkin, H.-M. Hanisch, P. Starke, and S. Roch. Formalisms For Verification Of Discrete Control Applications On Example Of IEC 61499 Function Blocks. In *Fachtagung Verteilte Automatisierung*, pages 72–79, Magdeburg, Germany, 2000.

V. Vyatkin, M. Hirsch, and H.-M. Hanisch. Systematic Design and Implementation of Distributed Controllers in Industrial Automation. In *Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 633–640, Prague, Czech Republic, September 2006.

V. Vyatkin, Z. Salcic, P. S. Roop, and J. Fitzgerald. Now that's smart! *IEEE Industrial Electronics Magazine*, 1 (4):17–29, 2007.