Gesellschaft für Informatik e.V. (GI)

publishes this series in order to make available to a broad public recent findings in informatics (i.e. computer science and information systems), to document conferences that are organized in cooperation with GI and to publish the annual GI Award dissertation.

Broken down into
- seminar
- proceedings
- dissertations
- thematics

current topics are dealt with from the vantage point of research and development, teaching and further training in theory and practice. The Editorial Committee uses an intensive review process in order to ensure high quality contributions.

The volumes are published in German or English.

Information: http://www.gi-ev.de/service/publikationen/lni/

ISSN 1617-5468
ISBN 978-3-88579-269-7

The volumes P175, P176 and P177 contain the Proceedings of INFORMATIK 2010 – the 40th Annual Conference of the Gesellschaft für Informatik e.V. (GI) which took place in Leipzig from September 27 to October 1, 2010.

The conference featured invited keynote talks of leading experts from industry and academia and selected workshops from different fields of computer science, mostly covering the general conference topic "Service Science – New Perspectives for Computer Science" ("Service Science – Neue Perspektiven für die Informatik").

K.-P. Fähnrich, B. Franczyk (Hrsg.): INFORMATIK 2010

175

GI-Edition

Lecture Notes in Informatics

Klaus-Peter Fähnrich, Bogdan Franczyk (Hrsg.)

INFORMATIK 2010

Service Science – Neue Perspektiven für die Informatik
Band 1

27.09. - 01.10.2010
Leipzig

Proceedings

Design Approaches for IEC 61499 Control Applications

Sebastian Preuße, Christian Gerber and Hans-Michael Hanisch

University of Halle-Wittenberg
Institute of Computer Science, Chair of Automation Technology
Kurt-Mothes-Straße 1, 06120 Halle (Saale), Germany
{sebastian.preusse, christian.gerber, hans-michael.hanisch}@informatik.uni-halle.de

Abstract: Current challenges require automation industry to be more flexible and tailored to a wide range of production scenarios. The trend shows up that the quantity of products is alternating, while the amount of possible product configurations increases. For this, manufacturing systems as well as control software applications have to be reconfigurable and reusable to meet these demands. The international standard IEC 61499 offers a lot of advantages to face these challenges. This contribution therefore proposes different design approaches for IEC 61499 control applications that are tailored to the demands of reconfiguration and reusability.

1 Introduction

The international standard IEC 61499 [IEC05] provides advanced possibilities and ways to design control applications. Crucial features are, amongst others, real object-orientation, event-driven execution behavior and distributed controller design. Furthermore, it claims vendor-independency, so that once designed control applications can be executed on any engineering environment.

Current demands, as for example system's reconfiguration and reusability, can only partially be fulfilled by the application of well-established methods, which are used for control software development following the international standard IEC 61131 [IEC03]. Usually, the control code consists of one main program that calls function blocks or functions and is processed cyclically or time-triggered. This design approach requests a complete re-engineering of the control program, when rebuilding or migrating the attached plant. However, plant operators demand a high grade of flexibility in combination with minimal downtimes. The IEC 61499 provides promising advances for tackling these problems. Anyway, application engineers have to be aware of the new functionality and have to reconsider the way of control software design.

This contribution therefore proposes IEC 61499 controller design approaches that have been developed during research work of the authors' workgroup. It is structured as follows. Section 2 gives a short introduction on IEC 61499. Afterward, a plant example is introduced in Section 3 that serves as a testbed to apply the design approaches, which are described in Section 4. Finally, the contribution is concluded in Section 5.

2 The IEC 61499 standard

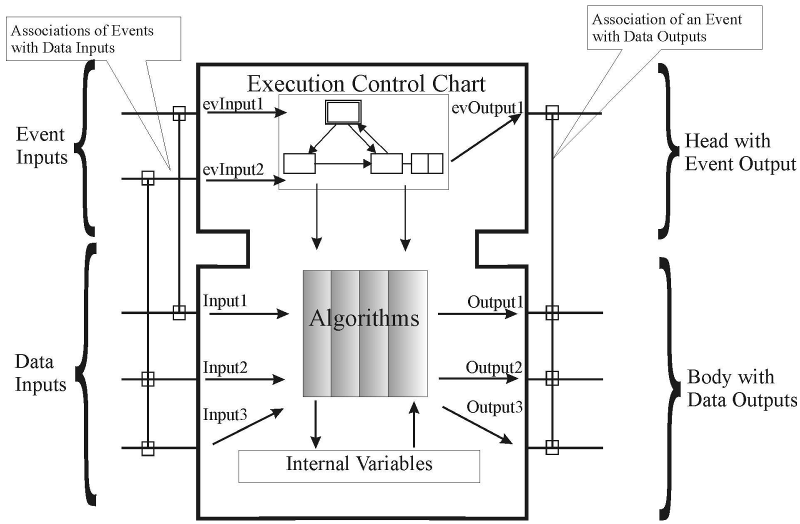

Figure 1: Basic Function Block.

The IEC 61499 is an industrial standard that defines an open architecture for the design of distributed control applications. The standard provides a generic model for distributed systems, which contains processes and communication networks as a basis for the distribution of applications on devices and their resources. It got finally standardized in 2005 and before that, it has been available as a Public Available Specification since 2000. Currently, the standard is under review for the second edition.

An IEC 61499 control system is based on Function Blocks (FBs) encapsulating functionalities and their signal interconnection. The functionality is invoked depending on events. The event-driven architecture represents a light-weight component solution that provides all essential features such as encapsulation of semantics from a particular execution platform, portability, reconfiguration and a holistic view on distributed applications.

Function Blocks provide an interface with event and data inputs and outputs plus associations between events and data variables. Different general FB types are defined according to their insides. Basic FBs are the basic building blocks. As shown in Figure 1, they consist of a body with data inputs and outputs and of a head with event inputs and outputs. Furthermore, they contain an Execution Control Chart (ECC) that controls the execution of algorithms depending on input events, input data and internal data and that produces output events. Composite FBs contain multiple component FBs - Basic FBs and/or Composite FBs.

The applications, represented by a FB network, can be distributed to any (set of) controller(s) within the system. Finally, Service Interface FBs (SIFBs) explicitly define mechanisms for the interaction of FB applications with hardware resources. That is for example reading the input variables from an input module, which is physically connected to the process that shall be controlled.

The encapsulation of controller functions into objects is a very intuitive approach because plants are constructed in a modular and hierarchical way as well. Therefore, the control application for each plant part is represented by its own FB and, as in real world, the FBs are hierarchically composed to build the whole controller structure. Figure 2 displays the generic system model of IEC 61499. The controlled process is connected to devices that execute the control software. Each device has several resources and the different applications can be mapped to these resources. As the structure is distributed, applications run allocated on one or more devices.
The standard features vendor-independency, amongst others, by an XML-based interchange format for FBs and control applications. There is no restriction concerning the applied controller and communication networks, while specific SIFBs are provided. This marks very clearly the benefit of IEC 61499 because in contrast to IEC 61131, the controller structure is really object-oriented and distributed and allows reconfiguration at runtime [SZFB^{+}06] and optimal utilization of resources [KHMH08].

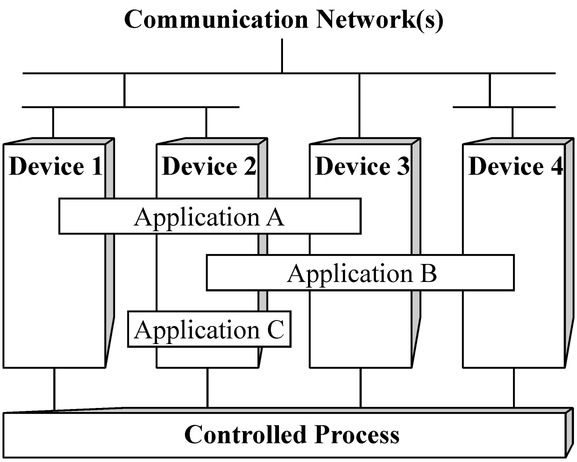

Figure 2: IEC 61499 generic system model.

3 Plant Example

The framework presented in this contribution shall be demonstrated using the example of a manufacturing prototype. The demonstrator in Figure 3 is part of the EnAS project [EnA10], which deals with energy self-sufficient actuators and sensors.
The testbed consists of two identical plant modules, which are rotated by 180° to each other. Conveyor ① marks the wait position and has one light barrier. It has no mounted processing station, so it can be used as a pallet buffer. Conveyor ② delivers pallets to the Jack Station and has two light barriers. Conveyor ③ transports pallets to the Gripper Station and has two light barriers as well. Furthermore, the figure displays the Jack Station ④, the Slide Station ⑤ and the Gripper Station ⑥.

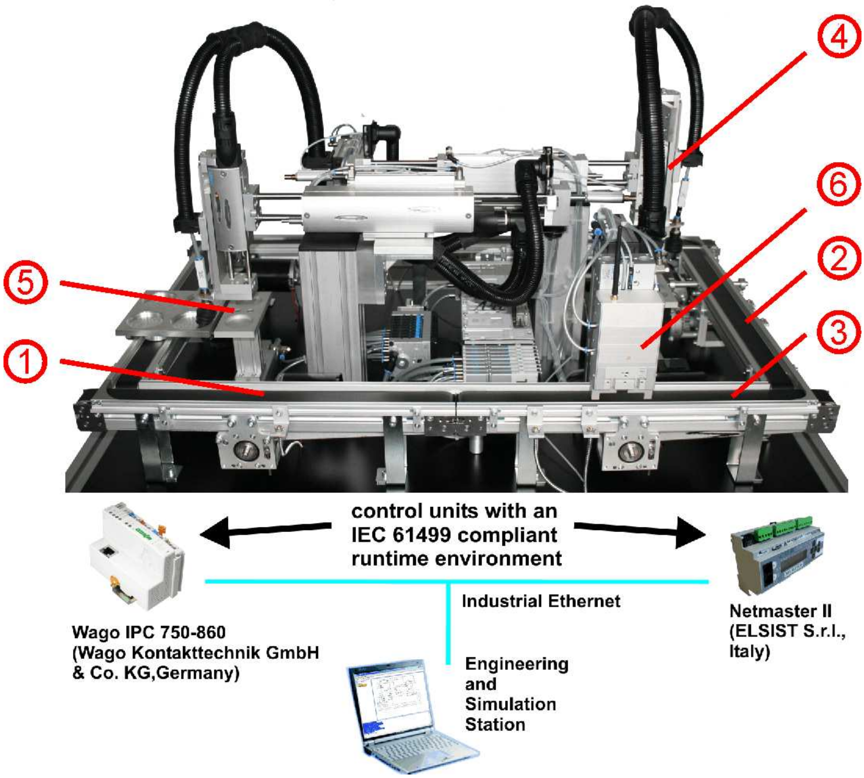

Figure 3: EnAS Demonstrator

To emphasize the effort of using the *Energy-Autarkic Actuator and Sensor System*, only the right plant module is equipped with it. The left one is classically wired. The conveyors (①, ②, ③) of both modules form a circuit and transport pallets cyclically to each processing station in the clockwise direction.

Each *conveyor* is moved by a separate drive and all six conveyors transport pallets with tins and workpieces inside from one processing station to another. Equipped with light barriers, the tins on the pallets are correctly positioned in front of the stations.

The first conveyor ① transports a pallet to the first light barrier, i.e. the wait position. After the light barrier is activated, the controller device switches off the drive and the pallet is stopped. If the drive is started again, the pallet will move on and will disable the light barrier after a certain time. The second conveyor ② brings the pallet by means of two light barriers to the correct position in front of the Jack Station ④. The first photo sensor indicates the first loading position, and the second one marks the second loading position. The third conveyor ③ transfers the pallets to two positions in front of the Gripper Station ⑥. The first position determines the correct positioning of the first tin. The second tin will be positioned if the second sensor switches on. After activating the drive of the last conveyor ③, the two light barriers will be disabled after a certain time. The pallet will

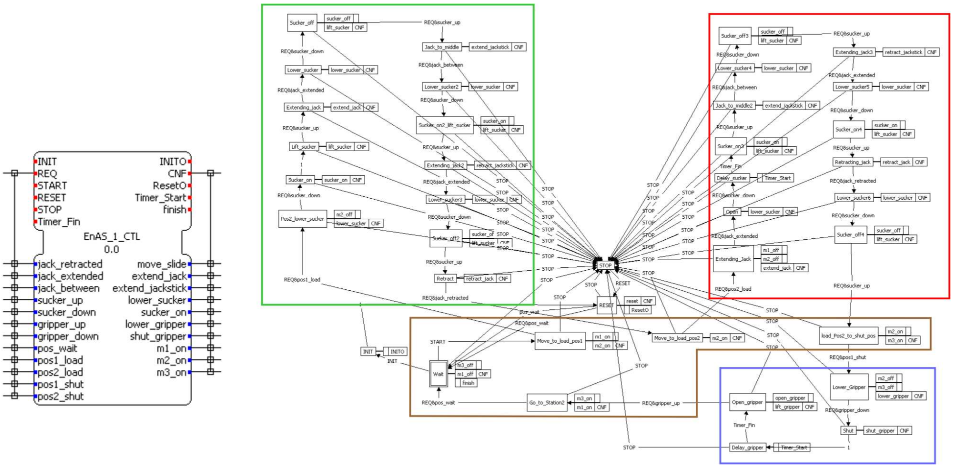

Figure 4: Central Controller of the testbed

move to the wait position of the following plant module if the drive of its first conveyor ① is switched on.

The *Jack Station* puts workpieces from the Slide Station to the tins on the pallet or vice versa. Furthermore, it can open a tin and deposit its lid onto the pallet or back onto the tin again. To do this, the sucker of the Jack Station can move to a high and a low position as well as to three different horizontal positions. The horizontal middle position can only be reached if the Jack Station extends an additional Jack Stick. By means of vacuum, the lid or the workpiece is sucked on and safely lifted when the sucker moves up. Through the combination of these possibilities, the tins can be loaded and unloaded, respectively.

The main function of the *Gripper Station* is to close the lid of a tin, which was previously loaded by the Jack Station. Furthermore, a tin can be lifted and deposed onto another pallet. Through compressed air, the gripper is moved up and down and activates the sensors *up* and *down* thereby.

4 Controller Design Approaches

This section introduces controller design approaches that have been developed during research work of the authors' workgroup. The order is chronological, since the approaches have been advanced due to solving new problems that appeared, when implementing control software for different testbeds. Therefore, the approaches are experimentally approved to be applicable for manufacturing systems and are suitable for large scale plants as well.

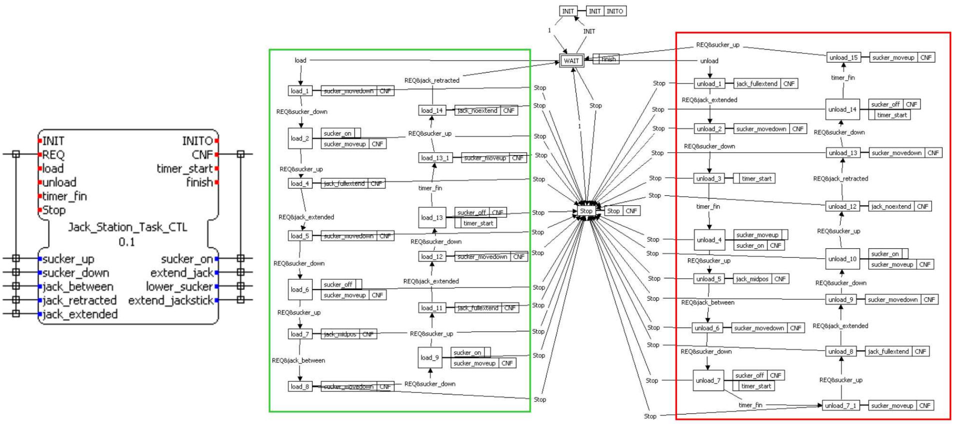

Figure 5: Task-Controller of the Jack Station

4.1 Central-Controller Approach

The Central-Controller approach is quite common since the controller functions are implemented within one FB, what makes it possible to quickly comprehend the sequence of the manufacturing steps. As the authors' testbeds have first been controlled by IEC 61131 conform controllers that usually consist of one main program and several function calls, the idea seems obvious to just map the functionality of the classic controllers to the IEC 61499 conform ones.

The FB presented in the left of Figure 4 is the central controller of the left plant part of the EnAS testbed. It controls all 3 conveyors and all processing stations. Therefore, the ECC shown in the right of Figure 4 gets huge and hard to maintain. Even though this FB is portable between several engineering environments conform to IEC 61499, it will not be reusable if the production scenario changes, and it will completely have to be reengineered.

Regarding the ECC, one can see four different controller tasks, which are marked in Figure 4. The upper right ECC part controls the opening and unloading of a tin, whereas the upper left part controls the loading and closing of a tin. The down center part controls the conveyors of the plant, and the down right part controls the movement and the closing of the gripper.

Using this information, several actions can be identified for every used mechanical component and distributed to separate FBs as described in [VHH06, HGVH07] for another plant. This leads to the Master-Task-Controller approach, presented in the following.

4.2 Master-Task-Controller Approach

The Task-Controller is the reusable part of the control application and can be stored in a library. It is once developed and verified for every mechanical component and controls the specific functions of it. If a component is removed from or is added to the plant, this Task-Controller FB will just have to be deleted from or inserted into the control application. Although the components are rearranged, the FB has only to be reconnected.
Figure 5 shows the FB and the ECC of the Task-Controller of the Jack Station. According to the ECC parts in Figure 4, the right rectangle borders all ECStates controlling the opening and unloading of a tin, while the left rectangle controls the loading and closing of a tin.
The control application is composed of all Task-Controllers, which handle the functionality. The program operation is coordinated by a Master-Controller. Such a controller does not have any algorithm inside. It is therefore very easy to develop and to understand. Even a concept of distributed master controllers was developed (for a different testbed) in [MHH07]. The Master-Controller FB triggers the execution of every task by events. As the Master-Controller is specific, it - but only it - will have to be re-engineered if the production scenario changes. But in contrast to the Central-Controller approach, the FBs controlling the elementary tasks are reusable. This makes the Master-Task-Controller approach more flexible and allows quick reconfiguration. Anyway, every new or changed FB has to be uploaded to the control device. Since not every runtime environment supports this dynamic reconfiguration as reported in [OWRSB05, VHH06], the application has possibly to be stopped and restarted again with the new configuration. The Parameterized Master-Task-Controller approach presents a solution for this problem and is presented in the following.

4.3 Parametrized Master-Task-Controller Approach

The approach of the Parameterized Master-Task-Controller uses Master and Task-Controllers, too. The Master-Controller incorporates every possible production scenario and coordinates it with the other Master-Controllers of the plant. The control application can be reconfigured by implementing all possible production scenarios and switching between them through changed parameters.
Figure 6 shows the Parameterized Master-Task-Controller and its corresponding ECC of the Jack Station. Through the data input *Actions*, an array of sequentially performed actions can be parameterized. Whenever the FB receives the input event *Jack* and is in EC-State *START*, the next action is read out from the array, and the connected Task-Controllers are accordingly triggered. The number of actions is counted by an internal counter *i*. If this number equals to the value of the data input *Length*, the counter is reset, and the planned production scenario starts again from the beginning. As described in [GHH09], each production scenario is specified by an activity diagram of the Systems Modeling Language (SysML). The specific scenarios are transferred to the controller via the human machine interface, so that a high degree of flexibility is reached because the process can be adapted

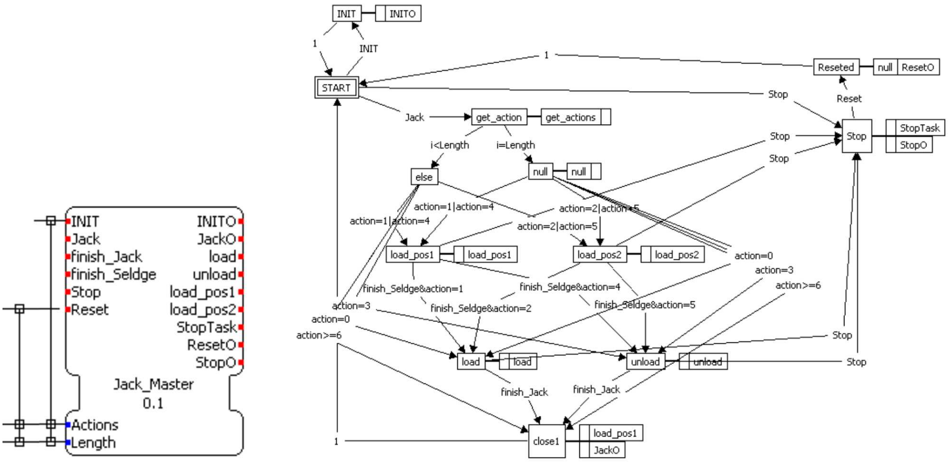

Figure 6: Parameterized Master-Controller of the Jack Station

to new demands without having to restart or even to change the code of the control device.

4.4 Workpiece-Controller Approach

The complexity of the planned production scenario can increase because of several reasons. For example, pallets are added to or removed from the manufacturing system, or there exist production alternatives, which influence each other for some reason. Consequently, the planning process will become the most time-consuming part of the reconfiguration.

The procedure can be simplified by developing the production scenario separately for every pallet. For this, each scenario is represented by one FB, namely the Workpiece-Controller, which controls the actions of the Task-Controllers. Each Workpiece-Controller allocates the Task-Controllers, which are necessary for the next production step, and releases them afterwards. If not all required Task-Controllers are available and a production alternative exists, this one will be chosen. Regarding the idea of prioritizing the ECTransition presented in [TD06], it is further possible to prioritize the production alternatives.

Figure 7 shows the FB Network, which controls the Gripper Station. It receives the *RUN* event, which is split and propagated according to the value of the event qualifiers *store* and *take* by the *E_Switch* FBs. Doing so, the actions *close, hold* and *deposite* are triggered. The event qualifier values are provided by the Workpiece-Controller FBs. For this, the production process is handled by the pallets that ”know” what actions have to be executed to process their workpieces.

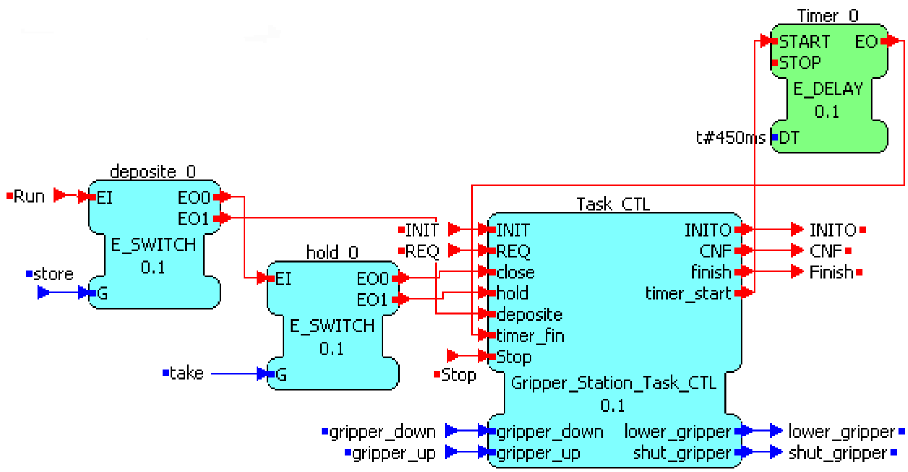

Figure 7: Function Block network to control the Gripper Station

5 Conclusion

The international IEC 61499 standard features numerous advantages for current demands of automation industry. The need for more flexibility requires reconfigurable and reusable control applications, what is facilitated by applying the standard. However, one will only benefit from the advantages if the design of control software is adapted to the improvements of the standard. All approaches, except the Central-Controller one, have in common that they use the same Task-Controller. Each Task-Controller implements all elementary actions, which can be performed by the corresponding mechanical component. Thus, each vendor of mechanical components can supply the Task-Controller as his intellectual property and encapsulate it inside a FB [VCL05].
A control application can easily be developed by inserting the Task-Controllers of the components and by interconnecting them according to the presented approaches. Following this procedure, the benefits of real object-orientation and distributed controller design show up and expose the advantages of applying the IEC 61499 for control software development.
Although the IEC 61499 standard supports issues as reconfiguration, reusability, and flexibility of controller design, it does not automatically guarantee that the designed controllers have these capabilities [PMG$^+$10]. An engineering methodology must be provided to guide the designer through the development process to really achieve that goal.
This contribution has shown some steps towards these goals that have been shown to be useful. There are other approaches as well in this quickly emerging field of technology. What has not been mentioned in this short contribution is the fact that the described controller design methodologies are further enhanced by means of formal verification [HMH08]. This is a rather significant means to ensure correctness and reliability of controller design. This goes, however, far beyond the scope of this contribution.

Acknowledgment

This work was partially funded by the Federal Ministry of Economics and Technology (BMWi) under reference 16 IN 0651 and 01 MG 566 on account of a decision of the German Bundestag and by the German Research Foundation (DFG) under reference Ha 1886/16-1 and Ha 1886/16-2. The authors are responsible for the contents of this contribution.

References

[EnA10] EnAS project website. URL, http://www.energieautark.com/, May 2010.

[GHH09] C. Gerber, M. Hirsch, and H.-M. Hanisch. Automatisierung einer energieautarken Fertigungsanlage nach IEC 61499. *Automatisierungstechnische Praxis (atp)*, 51:44–52, 2009.

[HGVH07] 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 *Proceedings of the Conference on Industrial Informatics (INDIN)*, pages 829–834, Vienna, Austria, July 2007. INDIN.

[HMH08] M. Hirsch, D. Missal, and H.-M. Hanisch. Design and Verification of Distributed Industrial Manufacturing Control Systems. In *34^{th} Annual Conference of IEEE Industrial Electronics (IECON)*, pages 152–157, November 2008.

[IEC03] IEC 61131, Programmable Logic Controllers - Part 3: Programming languages, 2003.

[IEC05] IEC 61499, Function Blocks - Part 1: Architecture, 2005.

[KHMH08] M. Khalgui, M. Hirsch, D. Missal, and H.-M. Hanisch. Reconfiguration of Embedded Systems. In *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 157–162, 2008.

[MHH07] 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.

[OWRSB05] S. Olsen, J. Wang, A. Ramirez-Serrano, and R. W. Brennan. Contingencies-based reconfiguration of distributed factory automation. *Robotics and Computer-Integrated Manufacturing*, 21(2005):379–390, 2005.

[PMG$^+$10] S. Preuße, D. Missal, C. Gerber, M. Hirsch, and H.-M. Hanisch. On the Use of Model-Based IEC 61499 Controller Design. *Int. J. of Discrete Event Control Systems (IJDESC)*, 2010. Manuscript submitted for publication.

[SZFB$^+$06] C. Sünder, A. Zoitl, B. Favre-Bulle, T. Strasser, H. Steininger, and Thomas S. Towards Reconfiguration Applications as basis for Control System Evolution in Zero-downtime Automation Systems. In *Intelligent Production Machines and Systems (IPROMS)*, pages 523–528, 2006.

[TD06] K. Thramboulidis and G. Doukas. IEC61499 Execution Model Semantics. In *Proceedings of the Conference on Industrial Electronics, Technology & Automation (CISSE-IETA)*, 2006.

[VCL05] V. Vyatkin, J.H. Christensen, and J.L.M. Lastra. OOONEIDA: an open, object-oriented knowledge economy for intelligent industrial automation. *Transactions on Industrial Informatics*, 1 (1):4 – 17, 2005.

[VHH06] V. Vyatkin, M. Hirsch, and H.-M. Hanisch. Systematic Design and Implementation of Distributed Controllers in Industrial Automation. In H.-M. Hanisch V. Vyatkin, M. Hirsch, editor, *Proceedings of the Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 633–640. IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2006), September 2006.