IEC 61499 – Enabling Control of Distributed Systems beyond IEC 61131-3

Magnus Holm, Göran Adamson and Lihui Wang
University of Skövde, Virtual Systems Research Centre, Skövde, Sweden
magnus.holm@his.se

ABSTRACT

Any global manufacturer needs to reduce production costs by continuously improving the efficiency of its production systems. To keep a high availability in the production systems with increasing complexity, operability and configurability are important variables for the control systems. Each device in the control loop holds more and more intelligence and computing power which can be used for a distributed automation system with improved efficiency and redundancy.

Today's device controllers, Programmable Logic Controllers (PLCs), in the production systems use the well-know and well-spread programming standard IEC 61131-3. However, this standard is not suitable for a distributed system since its definition is a system with centralized control (an "ordinary" PLC). To meet the requirements of tomorrow's control systems, the programming standard IEC 61499 was officially introduced in 2005. The IEC 61499 standard is designed to respond to the requirements of interoperability, reconfigurability and portability which are missing in the IEC 61131-3 standard. The foundation of the IEC 61499 standard is a distributed control system using event-driven function blocks where little effort is needed to program inter-device communication and message exchange.

In this paper, the two programming standards IEC 61131-3 and IEC 61499 are compared. Benefits of IEC 61499 are highlighted and examples of IEC 61499-based control are provided for better understanding.

Keywords: IEC 61499, Distributed Control System, Function Blocks

1 INTRODUCTION

Today's global economy is constantly demanding the manufacturers to adapt to changes in the market. These frequent changes expose the production systems to an increasing degree of uncertainty. To perform efficiently the controllers in the production systems must be able to cope with variations in the production. The system and its controllers must have a high adaptability and flexibility to meet the changing demands of the global market.

Traditionally, PLCs are the "brain" of the production systems since they hold the general intelligence when controlling the devices of the systems. In modern production systems more and more computing power and intelligence are encapsulated in different devices such as smart sensors and actuators. These intelligent devices open the possibility to meet requirements of interoperability, reconfigurability and portability which are missing in the traditional ways of programming a production system using the standard IEC 61131-3 [1]. In this standard the programming language Function Block Diagrams (FBD) is introduced. FBD encapsulates code enhancing easy-to-use and reuse but is not able to make use of the increasing intelligence of the spread devices in the production system.

The international standard IEC 61499 [2] introduced the concept of event-driven function blocks. It was officially published in 2005 as an IEC standard meeting the demands of adaptability, reconfigurability and flexibility for production systems and its automation using a distributed control system model. PLC-based control systems are the main application area of the IEC 61499 standard but it is also relevant to and can be applied in other industrial control systems such as robotic or CNC control. Event-driven function blocks can be used to encapsulate control code or machining data and can be used to generate and execute process plans. By using the event-driven function blocks in a distributed control system, as stated by the IEC 61499 standard, a device or machine becomes more intelligent and autonomous, facilitating decision making, at run-time. This ability is enabled by the embedded algorithms controlling the actual machine. A control system based on event-driven function blocks can also handle, for example, process monitoring, scheduling of dynamic resources and execution control [3].

The use of function blocks in programming were neither invented nor announced when publishing the IEC 61499 standard. It was an established programming concept offering robust and reusable components for the programmers working with industrial processes.

Inside the function block a software solution is encapsulated. The algorithm can control a small task, a conveyer, or a complex industrial task, a production line handling several products. Thanks to the easy-to-use nature of function blocks, a programmer does not need to have the full knowledge of the embedded algorithms but only the overall functionality of the function block when designing and developing a new system.

Through the IEC 61499 standard, a generic model for distributed systems is provided. The model describes a distributed control system, including processes and communication networks, for embedded devices, resources and applications. Event-driven function blocks facilitate an approach for a distributed control system enabling interoperability, reconfigurability and portability.

The remainder of the paper is organized as follows. In Section 2, we provide a literature review of the general research done based on IEC 61499 and also how function blocks have been used for control of manufacturing equipment. The IEC 61499 architecture is covered in Section 3 and a comparison with the previous PLC programming standard IEC 61131-3 is presented in Section 4. A production cell controlled by function blocks, including both milling operations and a gantry robot, is presented in the case study in Section 5. Finally, the conclusions are outlined in Section 6.

2 LITERATURE REVIEW

Since the end of the 1990s event-driven function blocks have been used and applied in distributed control systems during the emergence of the IEC 61499 standard. The focus of the absolute majority of the previous work done on event-driven function blocks have been on low-level process control for PLCs and syntax issues when executing distributed control systems [4]. There has not been much research done handling consequences due to uncertainty caused by changes in the production system configuration and layout nor process planning from a high-level production system view. Besides from the authors' own research team only partial research has been focusing on event-driven function blocks for adaptive process planning and machine control.

2.1 General Research on Function Blocks

Without doubt, the IEC 61499 standard is an important development of current practice in software engineering concerning production system control. Even though the standard was published some years ago, there is still a long road ahead before it will be adopted by the industry [4, 5 and 6].

When designing an autonomous distributed system, intelligent control components are often used together with function blocks. Early research by Wang et al. [7] pointed out how function blocks could be used for holonic control. In [8], Schwab et al. presented a web-based methodology for engineering and maintenance of distributed control systems. Other early research on engineering support systems covering architecture for development of function blocks was presented by Tramboulides and Tranoris [9]. A reconfigurable model for functions blocks and its implementation was presented by Brennan [10], and in [11] an automatic verification of a function block based industrial control system is described.

In [12], Dubinin and Vyatkin summarizes the semantic challenges of IEC 61499, suggesting a formal language for researchers when presenting and implementing verification systems and execution environments based on function blocks. In [13], Vyatkin and Dubinin discuss alternative semantics for the execution of IEC 61499 systems.

Aspects of real-time constraints during the verification process are discussed in [14] by Sünder et al. and also information concerning the behaviour of the function block run-time environment. A prototype model generator which automatically translates function blocks into net condition/event system models is presented by Pang and Vyatkin [15] who also developed a method using a generic data exchange mechanism together with function blocks to seamlessly integrate engineering tools used in the design process [16]. A method for improving the performance of the state machine, the execution control chart, inside the function block was developed by Vyatkin and Dubinin [17].

Rules for converting user-owned function blocks into IEC 61499 compliant function blocks are discussed in [18]. Using a model driven development approach as a foundation, an implementation method for transforming PLC code written according to IEC 61131-3 into IEC 61499 is presented by Wenger et al. [19]. In [20], a guide how to migrate function blocks from IEC 61131-3 to IEC 61499 is presented covering limitations and cautions during the migration process. In [21], Wenbin Dai and Vyatkin continue discussing how centralized PLC-control, based on IEC 61131-3, can be migrated to a distributed environment based on IEC 61499.

The well-known software IsaGRAFs, showing new potential of applying function block programming according to IEC 61499, is analysed in [22] along with the software FBDK/FBRT. IsaGRAF is found to be somewhere in the borderland between the two function block standards while FBDK/FBRT is solely focusing on IEC 61499 function blocks. Yang and Vyatkin [23] use the commercial software MATLAB when developing a simulation environment supporting the design of complex distributed systems. Generic function blocks, developed in a run time environment called FORTE, is used by Ebenhofer et al. [24]. These generic function blocks can be used independent of the hardware platform (platforms from Bachmann, Beckhoff, Siemens and Digi Connect are used in the test case) and be modified at runtime.

2.2 Function Blocks Used for Execution Control of Manufacturing Equipment

The demands of today´s manufacturing industries are the driving force in the process of evolving the programming techniques used for CNC systems.

Traditionally, G-code (ISO 6983) is implemented when programming manufacturing equipment using computer aided manufacturing (CAM) software together with computer aided design (CAD) tools. Even though the capability of G- and M-code has evolved significantly in parallel to the development of computer systems over the years, portability, the ability to use the same code for different CNC-machines, is still limited. Native and machine-specific G- and M-code has to be used. This strongly limits the possibility of sharing and transferring information, hindering an adaptive system to be configured. Methodologies aiming to enhance flexibility and interoperability for CNC machines have been developed during the past decade. One such approach is the standard for exchange of product model data (STEP) and the closely related data model STEP-NC. In [25], an approach using STEP and STEP-NC in an open architecture CNC-controller is presented which is programmed using an object oriented programming language. How a distributed architecture according to IEC 61499 can be used as a base for an open CNC controller is presented in [26] by Minhat et al. Wang et al. [27] presented an adaptable CNC system combining function blocks and STEP-NC having a system independent of the CAD/CAM system used. By translating STEP-NC code into the CNC controller's native G- and M-code the need of reconfiguration of the controller is eliminated. The approach made product data interchangeable and enabled a seamless information flow in the CNC system. An enhanced CNC controller using an extended STEP-NC data model is presented by Huang [28]. The STEP-NC data model enables the controller to work in a reconfigurable environment. Wang et al. [29] also used function blocks for process planning and adaptive control of a CNC machine. A function block based approach to process planning and scheduling with execution control is proposed by Wang et al. [30]. The function, enabled by the use of function blocks, covers distributed process planning and adaptive control.

The function blocks' ability to embed algorithms rather than executing fixed data such as G-code facilitates an adaptive, generic and portable alternative. Adaptive CNC controllers can make decisions at runtime when a change occurs in the production system implementing the new conditions.

Of course, manufacturing equipment other than CNC machines can be controlled by function blocks. Doukas et al. [31] present how the motion of a robotic arm with three degrees of freedom can be controlled using a function block based controller application. Some years ago, Thramboulidis [4] compiled the use of IEC 61499 in factory automation, considering possible inefficiencies of the function block paradigm supporting the development process of distributed control applications. A state of the art review covering IEC 61499 as enabler of distributed and intelligent automation was recently published by Vyatkin [32]. His conclusion is that the promising improvements related to IEC 61499 may lead to control systems that are automatically generated from the design documentation using integrated design methodologies.

A conclusion to be drawn from the presented research is that when manufacturing equipment is able to use function block based controllers the need of any vendor specific machine code is eliminated. The capability of executing the embedded algorithms of the function blocks directly in the machine controllers enhances adaptability, flexibility and portability.

3 IEC 61499 ARCHITECTURE

The IEC 61499 standard defines models supporting architecture for distributed control of industrial processes. There are five models in the standard: an application model, a system model, a device model, a resource model and a function block model. These models enable the engineer to use a graphical approach when developing applications for distributed control systems [3].

3.1 The system model

The system model is the top level of the IEC 61499 architecture. It defines the relationship between all physical communicating devices (PLCs, controllers, smart sensors, etc.) and the applications in the production system. An application can be distributed over several devices or exist in a single device (Figure 1). A distributed application is designed as a function block network where fragments of the network are placed in different devices. Since the controlling code is distributed to the devices, using their processing capacity, no main controller can be defined in the network, and the system becomes a truly distributed system.

3.2 The device model

The second level of the IEC 61499 architecture is the device model. A device is able to support one or several applications and hosts one or several resources. Each device has a process interface facilitating the exchange of data with the physical inputs and output (I/O) points. The communication interface provides data exchange with resources in external devices (Figure 2).

3.3 The resource model

To model how the function blocks behave within each resource is the main focus of IEC 61499. A resource and its properties, as defined in IEC 61499, are not far from the resource defined in IEC 61331-3. The resource provides independent execution and control of the function block network within it. Loading, configuration and also start/stop procedures can be done within the resource without affecting other resources in the same device or network. The resource holds, besides the function block networks, also scheduling functions and communication and process interfaces.

Within the resource, a network of function blocks, linked together by data and event flows, are shown in Figure 3. The correct execution order is assured by the scheduling function in the resource. One kind of function blocks defined in the standard IEC 61499 is the service interface function block (SIFB) which are used

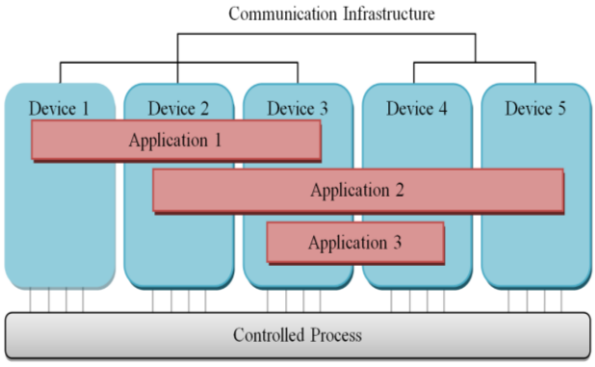

Figure 1. A system with devices and applications

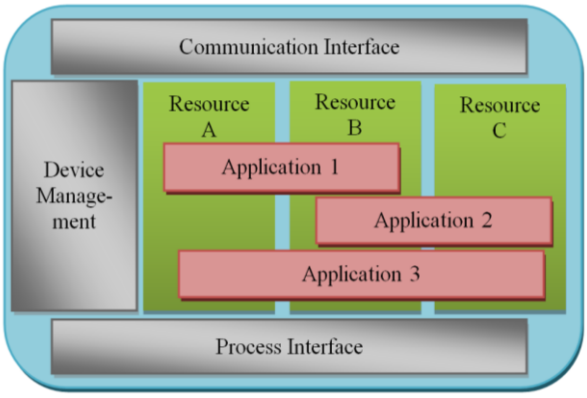

Figure 2. Resources in a device

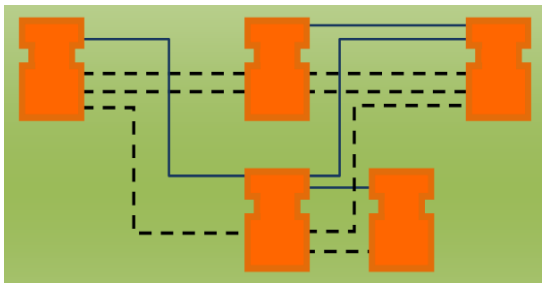

Figure 3. Function block network

when exchanging data or events to function blocks in another resource.

3.4 The application model

A network of interconnected function blocks linked by events and data flows is the IEC 61499 definition of an application. An application is a part of the production system, for example a PID-controller with sensors and an actuator. An application can be distributed over one or more devices and resources. The required event and data flows between the function blocks, resources and devices are defined by the application.

3.5 The function block model

The formal description of the data structure and the embedded algorithms of the function block model are the fundamentals of the IEC 61499 architecture. The standard defines several different function blocks. Lewis [3] summarizes their main features as:

- A type name and a unique instance name should be given to all function blocks.
- Each function block have a set of event inputs, which receive events from other function blocks, and one or more event outputs passing on events to succeeding function blocks.
- Each function block have a set of data inputs, receiving values from other function blocks, and a set of data outputs to pass data values produced by the internal algorithms to other function blocks in a network.
- Each function block has a set of internal variables for holding values until the internal algorithms are invoked next time.
- The behavior of the function block is defined by the internal algorithms and the finite state information (the execution control chart). When defining which internal algorithm to execute as a reaction to a specific incoming event, various strategies can be modeled.

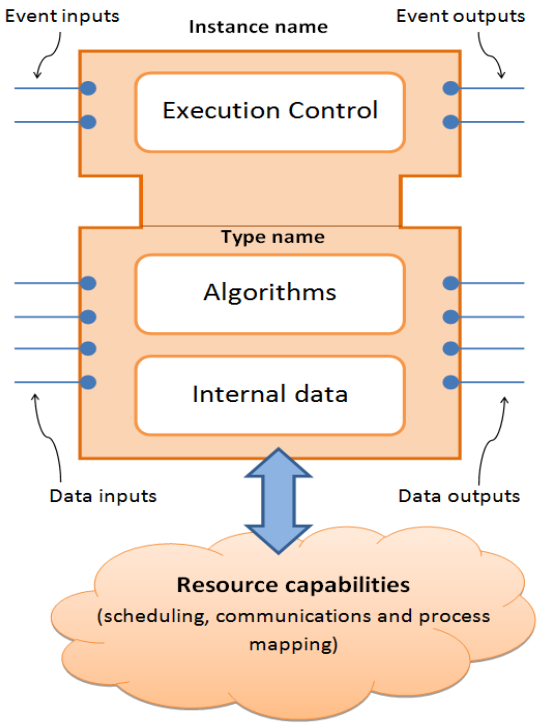

Figure 4. Characteristics of function blocks

The function block characteristics are shown in Figure 4. The “execution control” in the top part of the function block stipulates how the embedded algorithms, in the lower part, are activated on the arrival of incoming events. This is done by a finite state machine, or execution control chart. The resource hosting the function block facilitates the process with communication, scheduling and process mapping capabilities.

4 COMPARISON BETWEEN IEC 61131-3 AND IEC 61499

Looking into the PLCs of today's production systems, one can find that the programming standard IEC 61131-3 is used with few exceptions. Five programming languages are introduced in IEC 61131-3, one of them being Function Block Diagrams (FBD). At a first glimpse, the function blocks in both standards seem to be identical, both in look and in functionality. Since IEC 61499 has somewhat its origin in IEC 61131-3, this similarity is understandable. IEC 61131-3 is focused on how to program single processors and smaller systems using a low number of closely connected micro-processors. Increasing demands of distributed functionality for production systems pointed out the need of further standards beyond IEC 61131. IEC 61499 defines an architecture not only for design of functionality in distributed systems using many processors but also how data and information models should be defined to support an integration of system tools. Since the standard does not specify any programming language, an algorithm in the function block can be programmed using any applicable programming language such as C# or Java.

The FBD language defined by IEC 61131-3 has its limitations in execution control. The function blocks in FBD are linked together in a network by simply connecting the input and output variables. Each function block provides an internal algorithm and the function blocks are executed normally from left to right due to the dependency of the output values from the function blocks to the left (Figure 5).

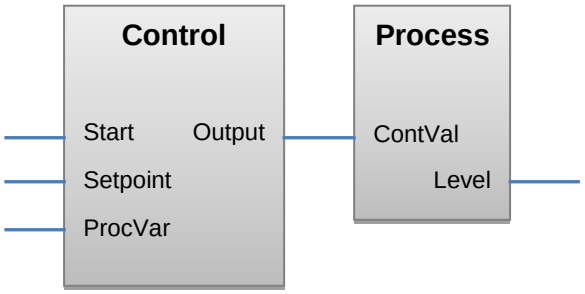

Figure 5. FBD network

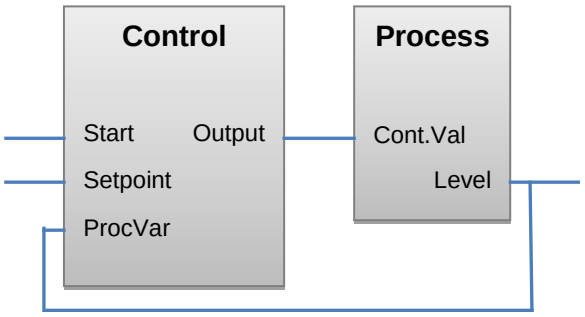

Figure 6. FBD network with feedback

In a more complex function block network where feedback signals are included and the execution order is hard to determine (Figure 6), additional mechanisms are provided by several IEC 61131-3 programming software, although these supplementary functions are not included in the scope of IEC 61131-3. As a consequence, the execution of FBD-networks is often defined inconsistent of the standard which strongly limits portability across different control systems.

IEC 61131-3 does have a system of passing events between function blocks but it does not fully meet the demands of flexibility in a complex function block network [3].

The execution control stated by the standard IEC 61499 uses events to control the execution of the event-driven function blocks (Figure 4) in the network. When designing the function block network, the engineer can through the events clearly define the execution order even in complex networks.

Another difference between IEC 61131-3 and IEC 61499 applications are how variables are used and the exchange of data between different resources. In 61131-3 both local and global variables are used where the global variables can be both read and written by any program in the resource. Global variables are problematic to define and also to synchronise in the whole network. It might be difficult to distinguish in which part of the system they are used and updated. In IEC 61499, no variables, neither local nor global, exist outside the functions blocks. Another method in IEC 61131 for exchange of data besides variables is to use communication function blocks. In a network of some few PLCs, it is possible to describe the behaviour of the system. Nevertheless, both global variables and communication function blocks fail to handle complex distributed systems with several devices, not only PLCs, with various functionalities in the system. The SIFB described by IEC 61499 and the communication interfaces handled by resources and devices are structured to handle complex distributed systems. The engineer designing the system does not need to configure the communication within the system but only decide to which devices that the communication should be directed.

Lewis [3] summarizes the insufficiencies of IEC 61131-3 adapted in a distributed system as:

- An application is not distributable over multiple resources.
- The execution order in a complex function block network is not always possible to clearly define.
- When assigning tasks to programs and function blocks the standard does not provide necessary flexibility.
- The execution model for function blocks provided by IEC 61131-3 can not be distributed across several resources.

5 CASE STUDY

A production cell including both manufacturing and assembly operations has been chosen for the case

study. As illustrated in Figure 7, the production equipment in the cell includes a 3-axis table top CNC machine and a 3-axis gantry robot. Other important devices in the cell are a PLC with I/Os, a PC and a human machine interface (HMI).

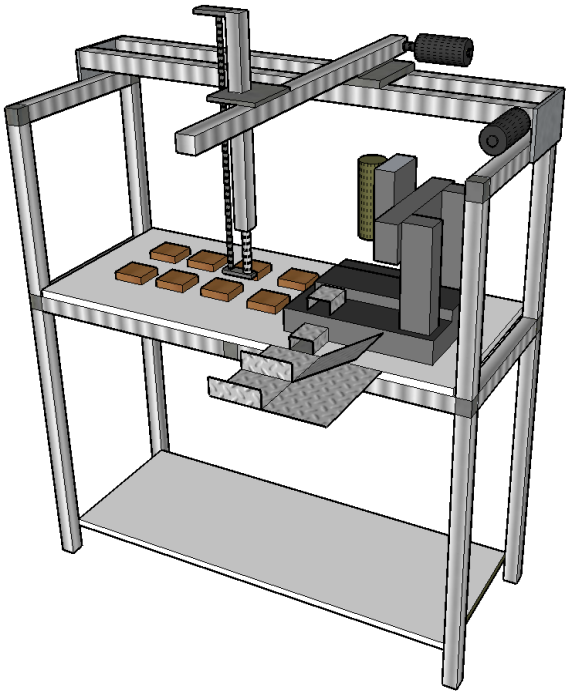

Figure 7. Production cell used in the case study

The robot gets raw material from the magazine and places it in the CNC machine. When the machining process is finished, the robot can assemble parts onto the machined products. After the assembly process, the finished product is moved by the robot to the output storage. The aim of the case study is to show that the whole cell with all its functionalities can be modelled and executed using IEC 61499 function blocks.

The communication between the PC and the PLC is via an Ethernet. Communication based on RS 232 is used between the PLC and the CNC machine. The SIFB managing the RS 232 communication receives controller commands from the previous function blocks and directly controls the CNC machine, through byte-strings, without translation to G-code. Moreover, the responses from the CNC are distributed through the SIFB back to the function block network in the PLC. The control of the robot is built in a similar way. The outputs from the function blocks are directly controlling the axes of the robot.

For control of both the gantry robot and the CNC, a function block network including machining feature function blocks (MFFB) are used. In an MFFB, all needed intelligence to control the specific device is embedded. When for example milling a pocket, the only inputs needed are the coordinates together with the size and depth of the pocket. How to mill the pocket, what approaching direction to use etc., are calculated by the embedded algorithms in the MFFB. The paths used for moving both the robot and the CNC axes are calculated by the embedded algorithms in the function blocks.

The operator inputs data to the production cell through a function block based HMI. There are three available manufacturing alternatives. The two major ones are to produce holes or pockets, deciding their sizes and positions. In the pockets, the gantry robot are able to assemble parts if the operator so decides. The third manufacturing alternative is the use of the CNC as an engraver and “mill” text onto the products. Using the HMI the operator also decides how many products to produce. The states of the process are displayed on the HMI throughout whole production cycle.

6 CONCLUSIONS

The result of the test case shows that a production cell programmed using IEC 61499 function blocks meets the requirements of interoperability, reconfigurability and portability. No vendor specific program code (G-code) is needed to control the equipment in the production cell. The PLC used in the test case is a Beckhoff CX1010, the CNC machine a KOSY 3 and the used software nxtControl.

This research is founded by the Swedish Knowledge Foundation within the research project Wise-ShopFloor. The authors would also like to acknowledge the contributions of José Enrique Palomeque Soto and Raúl Díaz Ríos who have worked together with us during the case study. Our thanks also go to the staffs at nxtControl GmbH for their valuable support during the project.

REFERENCES

[1] IEC 61131-3 (2003) *International standard of Programmable Controllers – Part 3. Programming Languages Ed 2.0*, International Electrotechnical Commission.

[2] IEC 61499-1 (2005) *International standard of function blocks—Part 1: Architecture*, International Electrotechnical Commission.

[3] Lewis, L. (2001) *Modelling control systems using IEC 61499,* The Institution of Electrical Engineers ISBN 0852967969.

[4] Thramboulidis, K.. (2005) *IEC 61499 in factory automation,* Proceedings of the International Conference on Industrial Electronics, Technology and Automation, CISSE-IETA 05.

[5] Strömman, M., Sierla, S. and Koskinen, K. (2005) *Control Software Reuse Strategies with IEC 61499*, 10th IEEE International Conference on Emerging Technologies and Factory Automation, Catania, Italy.

[6] Thramboulidis. K. (2009) *The Function Block Model in Embedded Control and Automation From IEC 61131 to IEC 61499,* WSEAS Transactions on Computers, **Vol.8, No.9**.

[7] Wang, L., Brennan, R.W., Balasubramanian, S. and Norrie, D.H. (2001) *Realizing holonic control with function blocks,* Integrated Computer-Aided Engineering, **Vol.8, No.1**, pp. 81-93.

[8] Schwab, C., Tangermann, M. and Ferrarini, L. (2005) *Web-based methodology for engineering and maintenance of distributed control systems: the TORERO approach,* The Third IEEE International Conference on Industrial Informatics.

[9] Thramboulidis, K. and Tranoris, C. (2001) *An architecture for the development of function block-oriented engineering support systems,* IEEE International Conference on Computational Intelligence in Robotics and Automation.

[10] Brennan, R.W., Zhang, X., Xu, Y. and Norrie, D.H. (2002) *A reconfigurable concurrent function block model and its implementation in real-time Java,* Integrated Computer-Aided Engineering, **Vol.9**, pp. 263-279.

[11] Norbert, V. and Krämer, B.J. (2001) *Automated verification of function block-based industrial control systems,* Science of Computer Programming, Elsevier, **Vol.652**, pp. 1-13.

[12] Dubinin, V. and Vyatkin, V. (2006) *Towards a formal sematic model of IEC 61499 function blocks,* The 4th IEEE International Conference on Industrial Informatics.

[13] Vyatkin, V. and Dubinin, V. (2007) *Alternatives for execution semantics of IEC 61499,* The 5th IEEE International Conference on Industrial Informatics.

[14] Sünder, C., Rofner, H., Vyatkin, V. and Favre-Bulle, B. (2007) *Formal description of IEC 61499 control logic with real-time constraints,* The 5th IEEE International Conference on Industrial Informatics.

[15] Pang, C. and Vyatkin, V. (2008) *Automatic model generation of IEC 61499 function block using Net Condition/Event Systems,* The 6th IEEE International Conference on Industrial Informatics.

[16] Pang, C. and Vyatkin, V. (2010) *IEC61499 function block implementation of intelligent mechatronic component,* The 8th IEEE International Conference on Industrial Informatics.

[17] Vyatkin, V. and Dubinin, V. (2010) *Refactoring of Execution Control Charts in basic function blocks of the IEC 61499 standard. IEEE Transactions on Industrial Informatics*, **Vol.6, No.2**, pp. 155-165.

[18] Gerber. C., Hanisch. H-M. and Ebbinghaus. S. (2008) *From IEC 61131 to IEC 61499 for Distributed Systems: A Case Study,* EURASIP Journal on Embedded Systems**, Vol. 2008.**

[19] Wenger. M., Zoitl. A., Sünder. C. and Steininger. H. (2009) *Transformation of IEC 61131-3 to IEC 61499 based on a model driven development approach*, 7th International Conference on Industrial Informatics, Cardiff, Wales.

[20] Dai. W. and Vyatkin. V. (2009) *A Case Study on Migration from IEC 61131 PLC to IEC 61499 Function Block Control*, 7th International Conference on Industrial Informatics, Cardiff, Wales.

[21] Dai, W. and Vyatkin, V. (2010) *On migration from PLCs to IEC 61499: adressing the data handeling issues,* The 8th IEEE International Conference on Industrial Informatics.

[22] Vyatkin, V. and Chouinard, J. (2008) *On comparisons of the IsaGRAF implementation of IEC 61499 with FDBK and other implementations,* The 6th IEEE International Conference on Industrial Informatics.

[23] Yang, C. and Vyatkin, V. (2010) *Model transformation between MATLAB Simulink and function blocks,* The 8th IEEE International Conference on Industrial Informatics.

[24] Ebenhofer, G., Rooker, M. and Falsig, S. (2011) *Generic and reconfigurable IEC 61499 function blocks for advanced platform independent engineering,* The 9th IEEE International Conference on Industrial Informatics.

[25] Minhat, M., Xu, X. and Vyatkin V. (2009) *STEPNCMillUoA: a CNC system based on SETP-NC and function block architecture,* International Journal Mechatronics and Manufacturing Systems*,* **Vol.2, No.1/2**, pp. 3-19.

[26] Minhat, M., Vyatkin, V., Xu, X., Wong, S. and Al-Bayaa, Z. (2009*) A novel open CNC architecture based on STEP-NC data model and IEC 61499 function blocks,* Robotics and Computer-Integrated Manufacturing*,* **Vol. 25**, pp. 560-569.

[27] Wang, H., Xu, X. and Tedford, J.D. (2007) *An adaptable CNC system based on STEP-NC and function blocks,* International Journal of Production Research*,* **Vol.45, No.17**, pp. 3809-3829.

[28] Huang, X. (2010) *Enhancing STEP-NC compliant CNC controller for distributed and reconfigurable environment in production line,* International Conference on Computer, Mechatronics, Control and Electronic Engineering, pp.106-109.

[29] Wang, L., Feng, H-Y., Song, C. and Jin, W. (2007) *Function block design to enable adaptive job shop operations,* Proceedings of ASME 2007 International Design Engineering Technical Conference, IDETC2007/DAC-34260.

[30] Wang, L., Hao, Q. and Shen, W. (2007) *A novel function block based integration approach to process planning and scheduling with execution control,* International Journal of Manufacturing Technology and Management, **Vol.11, No.2**, pp. 228-250.

[31] Doukas, G. S., Thramboulidis, K. and Koveos, Y. C. (2006) *Using function block model for robotic arm motion control,* Proceedings of the 14th Mediterranenan Conference on Control and Automation.

[32] Vyatkin, V. (2011) *IEC 61499 as enabler of distributed and intelligent automation: State of the art review.* IEEE Transactions on Industrial Informatics*,* **Vol.7, No.4**, pp. 768-781.