Implementation Guidelines for Closed Loop Control Algorithms on PLCs

Reinhard Hametner and Georg Schitter
Automation and Control Institute,
Vienna University of Technology
Gusshausstrasse 27-29/E376
1040 Vienna, Austria
hametner@acin.tuwien.ac.at

Andreas Voigt
VOIGT+WIPP Engineers GmbH
Märzstrasse 120
1150 Vienna, Austria

Alois Zoitl
fortiss GmbH
Guerickestrasse 25
80805 München, Germany

***Abstract*—The integration of closed loop control algorithms into industrial control systems represents an important topic in the automation domain. Timing constraints have to be taken into account, often leading to a compromise between real-time capabilities of the control and accuracy of the calculated values. Due to its event-based execution model, IEC 61499 bears advantages concerning the closed loop control application implementation. This provides flexibility and allows the implementation of various and more sophisticated feedback algorithms. The work described in this paper is concerned with these different implementation aspects. A detailed analysis is presented concerning their timing characteristics and solutions are proposed for the actual realization of these design variants by applying IEC 61499 function blocks.**

***Index Terms*—Automatic control systems development, Industrial automation systems, IEC 61499.**

I. Introduction

Embedded devices play an important role in the field of industrial automation systems. There is the need for using flexible and powerful Programmable Logic Controllers (PLCs) to cope with the increasing complexity of nowadays requirements from the automation industry [1].

Control algorithms for process industrial machinery like district heating plants or production systems for food are mainly implemented on embedded devices like PLCs. Computational and numerical extensive calculations for control algorithms are needed to optimize the industrial performance of many types of processes such as power plants, paper mills, and dryers.

To improve industrial process and resource efficiency it is important to control the plant by the use of feed forward and feedback control systems. Closed loop control allows for the stabilization of unstable processes, which are common in the process industry. Furthermore it can be designed for robustness, in order to handle disturbances and uncertainties of the process parameters, which may vary during operation or be even unknown [2]. Most used controller algorithms in industry are Proportional-Integral-Derivative (PID) controller algorithms. "PID control is by far the dominating control structure in industrial practice." stated by [3]. PID controller are most used to control Single-Input-Single-Output (SISO) plant systems, because the parametrization of PID controllers is easy to handle.

Current industrial automation systems handle closed loop control execution implicitly in cyclic execution model with a fixed time delay [4]. However, the fixed time delay is not suitable for all kind of control systems and therefore different implementation approaches have been investigated [5]. In this paper we focus on topology design guidelines for closed loop control strategies for new industrial control environments that give developers more control of the execution sequences (e.g., IEC 61499 [6]). The modeling for discrete real-time controllers is developed by applying the IEC 61499 standard in order to receive a platform-independent solution.

Section II of this paper present the state of the art of closed loop control implementations and event execution aspects for IEC 61499. Section III describes the derived research questions. Section IV shows the concept of processing industrial closed loop control applications, which is implemented, and results are presented in Section V. Finally, Section VI concludes the paper.

II. Related Work on Industrial Closed Loop Control Applications

IEC 61499 [6] has been developed as an extension to IEC 61131-3 especially as a methodology for modeling distributed industrial process, measurement, and control systems to obtain a vendor independent system architecture. As IEC 61131-3 the standard IEC 61499 defines concepts and models to encapsulate the control software in Function Blocks (FB), which can be assembled and newly distributed to several devices. The major difference between IEC 61131-3 and IEC 61499 is the execution model. IEC 61131-3 has an implicit sequential execution model with a cyclic activation of execution tasks. This has drawbacks for distributed applications where the cyclic execution model makes it hard to synchronize distributed applications and gives little control over the execution order of the function blocks in a control application.

In order to overcome this limitations events where introduced to IEC 61499. Such events control the activation of the application parts in a control application. Furthermore events are intended to actively control the execution sequence of the control application components (i.e., FBs) [7]. Although this intention is stated, up to now no investigation has been undertaken on developing rules and guidelines for the sequencing

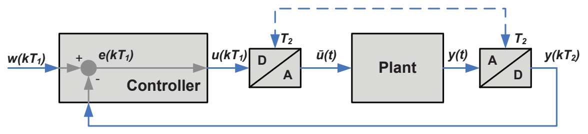

Fig. 1. Overview of the closed-loop control system, consist of the plant which needs to be controlled and the controller (e.g., PID controller). Sampling rates are T_1 and T_2.

of application parts in general and also not for closed loop control applications. Particularly, this work targets the latter domain. The structure of such closed loop control system is presented in Fig. 1.

A special FB library for closed loop control algorithms applied on the IEC 61499 standard is presented in [8]–[10]. They use for the processing of the events an cyclic execution process for the closed loop control application. Lastra et al. [11] demonstrates how an event-based system like IEC 61499 can be adopted for a scan-based embedded controller.

A transformation process from a developed closed loop control system application in MATLAB/Simulink into the IEC 61499 applications is presented by Yang and Vyatkin [12]. For this transformation all used behavior models have to be available in both Simulink and as FBs. Only the mapping is done by the transformation process.

The results of such available concepts from above are a basis for using closed loop control systems in industrial dynamic systems, but there is the need for specifying the processing order of the control application in detail to enable real-time control on PLCs. Therefore an appropriate execution handling concept for IEC 61499 applications has to be developed to apply these results in industrial control applications.

III. Research Questions

RQ 1: How can a framework for implementing closed loop control systems be developed considering real-time processing on PLCs? Closed loop control algorithms are an important controller type for industrial automation applications. However, current implementations are done in an ad-hoc way without any methodological approach for a target application optimized real-time execution behavior.

RQ 2: How can closed loop control systems be realized by industrial event-based execution methods? Closed loop control implementations demand strong constraints regarding timing, execution sequences, and interaction with other system parts. The event triggered mechanism introduced in IEC 61499 provides a valuable means for supporting and specifying such constraints in a natural way. In order to make use of these mechanisms an analysis on the implementation requirements of closed loop control and resulting structural design patterns need to be done.

IV. Event Concept for Implementing Closed Loop Control Systems

A. *Requirements for Industrial Closed Loop Controllers*

The general requirements of process control solutions on embedded platforms (e.g., PLCs) or customized controller boards (e.g., FPGA) are lean run-time environments and availability of pre-engineered controller Function Block libraries with advanced functionality. The common basic PID controller blocks do not allow design of more than one control strategy level (i.e., one manipulated variable). Having only static set-points and constraints is not sufficient to realize controller networks in performance optimized automatic operation software.

To achieve the desired quality of process control, the following functionality is needed. Model based calculation of PID parameters from plant characteristics (delay, dominant time constant, gain, and required closed loop dynamics). Further, inputs and modes for multiple external reference inputs to realize more complex process control strategies are useful. Internal model control and feed forward disturbance rejection to achieve high control performance even with large time delay would be practical. Tuning support for different types of controllers and plant types like internal model controller design and availability of specialized control algorithms for time delay systems (e.g., model based predictive controllers, or Kalman filters for noisy processes) would increase the desired quality of control applications.

Additional support of dynamically changing manipulated variable (MV) constraints with anti-windup as well as split-range or multi-controller networks are useful to give control engineers powerful tools to create various control strategies. Required controller set-point operation modes are:

- Manual (MAN) - The MV is set to a constant value, either by the operator or by another software part
- Auto (AUTO) - The controller uses the internal set-point to control the process variable (PV) towards this set-point
- External 1 (EXT1) - The controller uses the first external set-point
- External 2 (EXT2) - The controller uses the second external set-point
- Track (TRACK) - The controller output (MV) follows an external signal

General features, are needed to design sophisticated controller strategies on embedded systems to realize advanced process control for industrial processes:

- On/Off switch of the derivative-part on set-point, the actual value is always used for calculating the derivative-part
- Configurable rate limiters for all inputs
- Split-Range controller parameter switch: if controller output sequentially influences more than one actuator the parameter setting of the controller can changed (Switchable Controller Parameter, Gain Scheduling)
- Anti-Windup functionality for the controller when the manipulated variable (output) is in saturation, therefore the integrator-part of the controller will also be saturated
- Override control needs status inputs of the controller block: the controller needs the information when it is active and when another override controller takes the lead

The proposed requirements should consider to develop an applicable advanced industrial controller application. The main property of cooperative advanced process controllers is the mutual interaction of each other which has to be handled.

B. Timing Consideration of the Plant and Controller Interaction

In order to determine the appropriate execution sequence for the closed loop control application the first step is to investigate the requirements of closed control theory for discrete control applications. In theory the ideal case is that at time point *kT*. The measure process, value calculation, and write the output values to the plant are done at the same time. Fig. 2.a shows this ideal case. However this ideal case is not realizable, because of the time required for the analog-digital and digital-analog conversion and the execution time needed for the computation of the controller.

For a first investigation of the execution time for the control algorithm we can take a general linear digital controller with the order of 'nr' based on Åström and Wittenmark [5]:

$$F_R(z) = \frac{u(z)}{e(z)} = \frac{a_0 + a_1 z^{-1} + ... + a_{nr} z^{-nr}}{1 + b_1 z^{-1} + ... + b_{nr} z^{-nr}} \qquad (1)$$

From this the finite difference equation for the discrete calculation is:

$$\begin{aligned} u(k) &= a_0 e(k) + a_1 e(k-1) + ... + a_{nr} e(k-nr) \\ &\quad -b_1 u(k-1) - ... - b_{nr} u(k-nr) \end{aligned} \qquad (2)$$

A more compact vector notation can be obtained through

$$\begin{aligned} \underline{v}_u(k) &= [e(k) \ldots e(k-nr) \vdots \\ &\quad -u(k-1) \ldots - u(k-nr)]^T \end{aligned} \qquad (3)$$

$$\underline{p} = [a_0, a_1 \ldots a_{nr} \vdots b_1 \ldots b_{nr}] \qquad (4)$$

to

$$u(k) = \underline{p} \cdot \underline{v}_u(k). \qquad (5)$$

As explained the output of u(k) is not possible at the time point $t = kT$ because of the AD/DA conversion and the computation time needed for Equation (5). Depending on the order of the control algorithm and processing power of the used control device the execution of Equation (5) requires the time T_u. Therefore the earliest possible output $u(k)$ is available at $t = kT + T_u$. In addition to the delay time T_u the other execution taking place in the device (e.g., operating system functions) may additional introduce delay. This delay is typically varying between the different cycles and results in output jitter, see Fig. 2.b [5].

However the not-synchronized output generation (i.e., not in line with the sampling) and the jitter do not allow an exact evaluation of the closed loop control in the z-domain, which is the basis for the stability analysis. In order to overcome these limitations and reduce or avoid the output jitter is to introduce an extended but defined output delay of exactly one sample step [5]. This means that at the time instant kT the measure procedure $e(kT)$ and the output $u(kT-1)$ will take place, see Fig. 2.c [5]. This one sample unit delay is

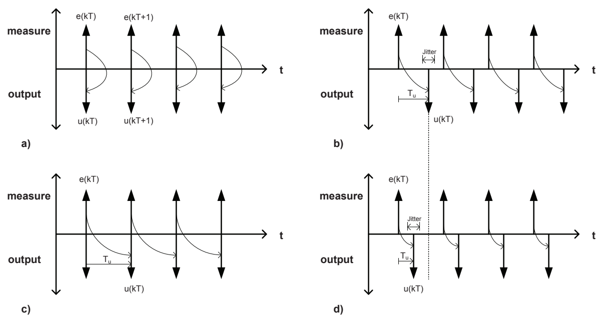

Fig. 2. Time Diagram: a) ideal case with measuring and output point at the same time; b) measuring and output as soon as possible, c) measuring and output at the next sample step, delay time T_u equals sample time; d) optimized case, measuring and incremental calculation with fast output

in many cases less critical than the disturbances introduces by the indefinite output generation of the as-fast-as-possible approach before. Furthermore the one sample unit delay can be considered during in the closed loop controller design.

In certain applications it would be beneficial if we could get as near as possible to the ideal case presented in Fig. 2.a by reducing the execution time T_u. This can at the one hand be achieved with faster control devices or by splitting the execution time allowing to perform part of the calculations at uncritical time instants [5]. At first we can extract from Equation (3) and (4)

$$\begin{aligned} \underline{v}_u^*(k) &= [e(k-1) \ldots e(k-nr) \vdots \\ &\quad -u(k-1) \ldots - u(k-nr)]^T \end{aligned} \qquad (6)$$

$$\underline{p}^* = [a_1 \ldots a_{nr} \vdots b_1 \ldots b_{nr}]. \qquad (7)$$

The use of Equations (3) to (5) delivers

$$u(k) = \underline{p}^* \cdot \underline{v}_u^*(k) + e(k) a_0. \qquad (8)$$

The expression $\underline{p}^* \cdot \underline{v}_u^*(k)$ can be calculated at the time range $(k-1)T \leq t \leq kT$. Thus the calculation of $u(k)$ according to Equation (8) can be performed with this pre-calculated term and a simple term. This needs substantially less calculating time than the calculation corresponding to Equation (5). The resulting execution sequence is shown in Fig. 2.d.

C. IEC 61499 Structural Design for Closed Loop Control

With the investigation on the execution requirements of the closed loop control system we can now derive resulting general event flow design patterns for implementing discrete closed loop control systems in IEC 61499. Fig. 3 shows different event handling configurations of the closed loop control system. This system consists of an `E_CYCLE` FB, the `Controller` FB, and the `Plant` FB. The `E_CYCLE` FB is responsible to generate a cyclic event with a time period `DT` defining the sampling rate for the discrete closed loop control. The `Plant` FB is in charge for encapsulating and handling the interaction with the controlled plant. That means it has to perform the measurement and data pre-processing of the sensor values and the setting of the controller outputs to the actuators.

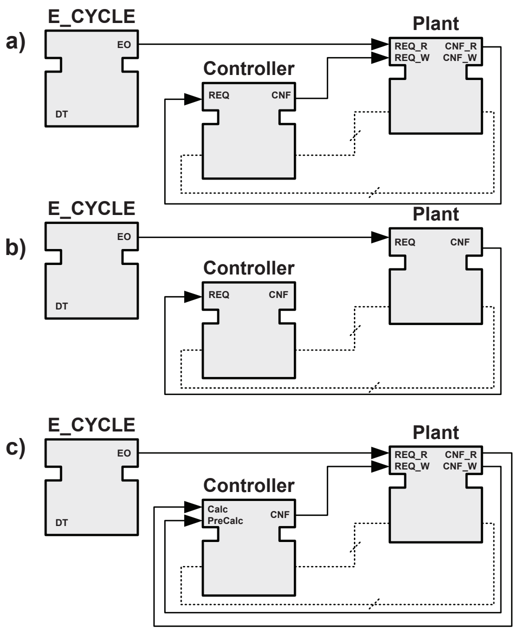

Fig. 3. Event handling of the closed loop control design; a) output directly after calculation, b) output with delay of one sample step, c) optimized output with incremental pre-calculation in each cycle for fast output

The first event design pattern presented in Fig. 3.a is the write-as-fast-as-possible approach. This approach is based on Fig. 2.b. The `E_CYCLE` FB triggers in each cycle the plant at the `REQ_R` to read the discrete parameter sensor values of the plant. This is acknowledged with the event `CNF_R`, which will trigger the execution of the Controller. After the Controller has finished its calculations it triggers the `REQ_W` event, informing the plant that new controller values are available for the actuators.

The second design pattern, presented in Fig. 3.b, has a one sample time delay, as shown in Fig. 2.c. It can be achieved by eliminating the triggering of the plant after the controller has finished. In this case when the `Plant` FB is triggered by the `E_CYCLE` FB it will not only read the sensor values but also set the currently available controller values for the actuators. By triggering the `Controller` FB after the `Plant` FB the controller will calculate new controller outputs with the most recent inputs and because of the missing event link to the plant they will be used at the next time the plant is triggered. Furthermore with the single event for triggering the input reading and output writing of the plant the jitter is greatly reduced and therefore this implementation is very near the ideal theoretical one used for designing the closed loop controller.

The final event design pattern represents the optimized case shown in Fig. 2.d, where spare time in the cycle is used for doing preparation calculations for the next cycle. Similar to the as-fast-as-possible design pattern of Fig. 3.a first the sensor reading is triggered followed by the control algorithm and an immediate output writing (i.e., the event sequence `REQ_R`, `CNF_R`, `Calc` and `REQ_W`, see Fig. 3.c). However, then the confirmation of the writing process is used (`CNF_W`), for triggering the pre-calculations of the controller. This design pattern is not a pure event design pattern as it requires also a special designed controller implementation that offers the possibility that the two calculation steps can be triggered separately.

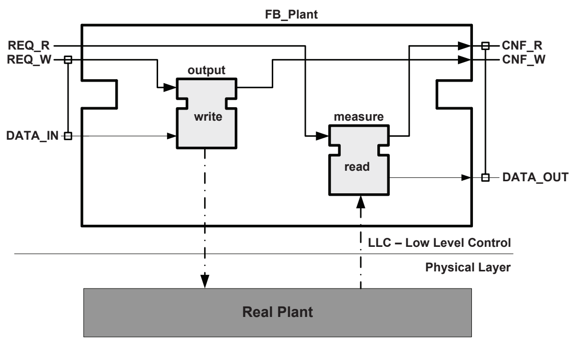

Fig. 4. Plant interface with separate event inputs for triggering a FB to read parameters and FB for write parameters to the real plant

D. Interface Design of the Plant

For interacting with the controlled plant we introduced in the previous section a generic `Plant` FB. This FB encapsulates all actions required for delivering sensor data and handling actuator data. With it we achieve a separation of application parts into hardware independent and hardware dependent parts [13]. As already stated the correct and optimal event handling of the plant enables an accurate calculation of the closed loop control system. For this we propose a general design pattern for the plant interface that allows the application in any of the controller event design patterns shown in Fig. 3.

Fig. 4 shows the event and data configuration of the `FB_Plant` interface design as well as the internal structure. The plant can be split into a low level control (LLC) layer and a physical layer. The LLC layer consists of the controller, measurement systems, and output units to control the physical layer, which contains industrial sensors and actuators. The interface from the LLC layer to the physical layer is realized with the `write` and `read` FBs which are contained in the `FB_Plant`. These FBs contains the communication and interaction means with the I/O interface of the hardware. According to IEC 61499 this is achieved with so called Service Interface FBs (SIFBs).

The `FB_Plant` interface has two separate event inputs with corresponding event outputs. One set (`REQ_R CNF_R`) for performing the sensor read operation and the second set for performing the actuator write operation (`REQ_W CNF_W`). The `WITH` qualifier, which is specified in the IEC 61499, is used to specify an association between data input or data output variables and an associated input event or output event. This feature can be used for the specification of the sampling of data at the FB's interface (i.e., into the FB or out of the FB). If the event `REQ_R` is triggered, an output event `CNF_R` occurs and therefore the data output (i.e., measured sensor value) is available for further processing. Analogue, if the event `REQ_W` is triggered, the input data of the `FB_Plant` (i.e., controller

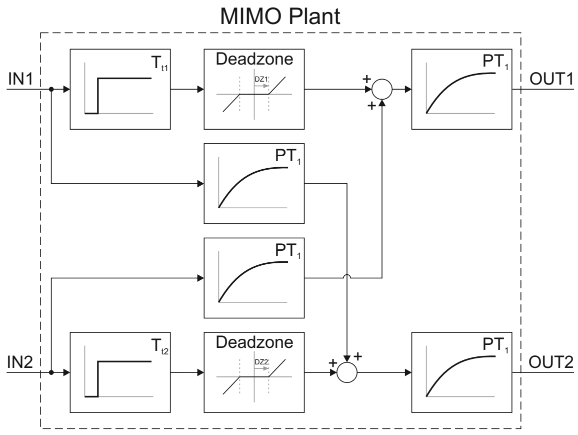

Fig. 5. Overview of the used Plant structure represents an incinerator for energy production

output) will be written and sent by the `write` FB to the real plant and confirmed by the output event `CNF_W`.

V. Example Implementation

A. Integration of the Plant

For the example implementation we use the model of incinerator as it is used in the energy production domain. The plant represents a weakly coupled first order process with input time delays, actuators with dead zone (DZ) behavior, and PT1 elements (Fig. 5). The plant consists of two inputs, IN1 is the fuel supply and IN2 is the combustion air supply. These two parameters are used to control the temperature of the boiler.

Both time delay blocks, represents the time duration of the fuel (T_{t1}) / air (T_{t2}) supply through the pipe into the burner. Further, both inputs are characterized by a DZ element representing a valve position controller DZ for the fuel and air supply flow through the valve. This is used to prevent the fuel supply valve from permanent movements to minimize the wear. Typical DZ parameters are approximately 1-2 percent of the valve position.

The two PT1 elements, before the output, represents the burner boiler with a thermal inertia and nonlinear heating actuators. The main process dynamics are modeled by LTI (Linear Time Invariant) first order lag elements for the heat capacity or mass (upper right PT1 block) and the gas mixing process in the burner (lower right PT1 block). The weak cross-coupling of the process is modeled by first order lag elements which influences the dynamic behavior of the burner.

Output 1 (OUT1) is the burner temperature and Output 2 (OUT2) is the oxygen excess concentration in the flue gas. Both plant inputs have effects on both outputs. That means, the fuel supply increases the burner output temperature and decreases the oxygen excess concentration of the flue gas as well as the combustion air increases oxygen content in the flue gas and decreases the burner temperature.

A simulation model is implemented with the behavior of the presented plant in 4DIAC by IEC 61499 FBs. This plant which has to be controlled is designed to validate and proof the following features of the closed loop control strategy commissioning environment:

- Time delay to show that a large number of states can be handled.
- Online change of time delay value have to be checked. Therefore a dynamic allocation of memory is needed.
- Hard non-linearity of the dead zone actuators.
- Cross coupled input output relations of the process.

Fig. 6 shows the used FB interface design of the plant (`MIMOPlant_CS`) which communicates with the simulated plant.

B. Implementation of the Industrial PIDT1 Controller

A special PIDT1-Controller is developed to be able for processing control systems in a wide industrial application field. Therefore the requirements from Section IV-A are considered and applied for the development of the industrial applicable PIDT1 controller. Common straight forward implementations of the classical PID-Controller are not applicable for industrial applications, because there is the need to have some additional features like set-point mode switches.

The PIDT1 controller is designed as IEC 61499 composite FB which includes several encapsulated functions. First of all, the common PID controller functionality is implemented. Then a *limit* functionality is implemented which supports an upper and lower value limitation of the controller output. Furthermore an additional *dead zone* functionality is implemented. This feature is needed to level the manipulated variable, so that the controller needs not to work for small deviations. To get the possibility to add an additional *offset* value for the manipulated variable a feed-forward parameter can be parametrized. In some cases a *negative gain* output value is needed. Therefore the input parameter `NGAIN=true` can be set. One of the most used feature is the *set-point mode switching*. The mode switch will start when the `MODE` parameter is changed. At the moment six set-point values can be chosen, i.e., `SW_IN`, `EXT1`, `EXT2`, `MAN_IN`, `EXTY`, `TRACK`. Especially, for some of set-points additional time values can be set, e.g., `T_SW=1s`, `T_EXT1=3s`, `T_EXT2=0.5s`. These time values are used for the smooth crossover by a ramp functions, when switching between the setpoints. This is need to avoid set-point values step changes which can result in unstable process behavior.

C. Resultant Overall Control System

The implementation of the closed loop control algorithm is done in IEC 61499 using the open source project called 4DIAC [14]. We used two controller instances (i.e., two PIDT1-Controller) to control the MIMO plant behavior. Each PIDT1-Controller for each plant input. We implemented the three proposed event concepts which have been identified in the last section, see Fig. 3.

Fig. 6 shows the IEC 61499 implementation of the event handling approach which is presented in Fig. 3.b. The event-handling concept presented in Fig. 3.a as well as Fig. 3.b are applicable for the process industry like district heating plants. Therefore, such control systems consists of a large response

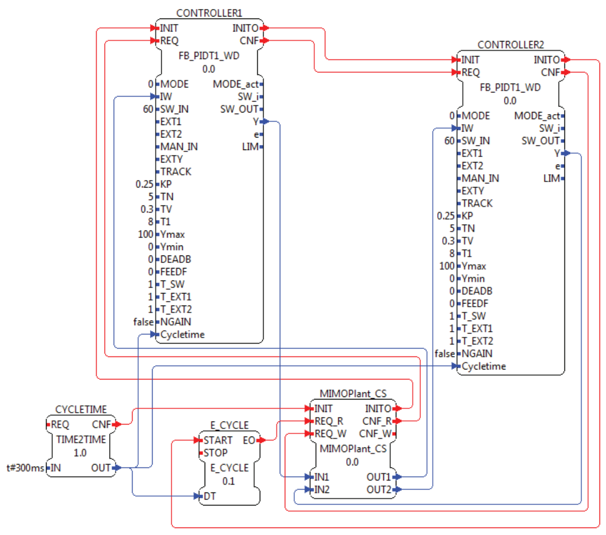
Fig. 6. Implementation of the Closed Loop Control System in 4DIAC

time and the pre-calculation concept presented in Fig. 3.c is not necessary. The concept pictured in Fig. 3.c is applicable for systems which needs a short response time like motion control applications [15].

The controllers are triggered by an `E_CYCLE` FB with a cycle time of $t_{controller} = 300ms$ whereas the plant is triggered by a different internal cycle time of $t_{plant} = 100ms$. The triggering of the closed loop control system is implemented by a hybrid execution processing approach (event-based and time-driven approach). Therefore the controller part and the plant part are sampled cyclically by a fixed time rate which we called time-driven approach with an overall event-based execution triggering of the closed loop control system, see Fig. 6. Furthermore, the internal processing of the controller and the internal processing of the simulated plant are handled by IEC 61499 events. This results in an ordered execution of the `Controller` FB and the `Plant` FB. There is the possibility to define the execution order of the internal behavior from the `Controller` and the `Plant` FB by events explicit.

VI. Conclusion and Future Work

This paper outlines how closed loop control systems can be used for industrial automation applications by using IEC 61499 FBs. The requirements applicable for industrial controller algorithm implemented on embedded devices like Programmable Logic Controllers have been shown. Based on our analysis and the resulting requirements of our industry partners we developed an advanced industrial PIDT1-Controller which is usable for the process industry like district heating plants. In order to determine the appropriate execution sequence for the closed loop control application by IEC 61499 events, three different processing concepts have been developed and implemented. Through the evaluation of the proposed event handling concepts after implementation, several fields of application could be identified. Hence, we used the resulting design patterns of the closed loop control for the process industry domain and for controlling motion control applications.

Our future work on modeling event-based closed loop control systems will focus on using a model predictive controller (MPC) instead of the PIDT1-Controller. Therefore an industrial MPC running on embedded systems will be developed which will be used in event-based automation applications. Our current execution triggering implementation of the closed loop control algorithm applications uses a hybrid execution approach (i.e., time-driven and event-based approach). In future work the execution process will be switched from the hybrid approach to a fully event-based approach by using interrupt-driven methodologies.

Acknowledgment

The authors want to thank Richard Wipp and Lukas Haffner from VOIGT+WIPP Engineers GmbH for their fruitful discussions. This research has been funded by the Klima+Energie Fonds under grant agreement no. 829823, managed by the Österreichische Forschungsförderungsgesellschaft (FFG).

References

[1] G. Doukas and K. Thramboulidis, "A Real-Time-Linux-Based Framework for Model-Driven Engineering in Control and Automation," in *IEEE Transactions on Industrial Electronics*, 2011, pp. 914 – 924.

[2] G. Franklin, J. Powell, and M. Workman, *Digital control of dynamic systems*, ser. world student series. Addison-Wesley, 1998.

[3] K.-E. Årzén, "A simple event-based pid controller," in *IFAC World Congress 1999*, 1999.

[4] J. H. Sandee, W. P. M. H. Heemels, and P. P. J. Van Den Bosch, "Case studies in event-driven control," in *Proc. of the 10th Intl. Conf. on Hybrid Systems: Computation and Control*. Berlin: Springer, 2007, pp. 762–765.

[5] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems Theory and Design*, 2nd ed. Prentice-Hall, 1990.

[6] IEC 61499-1, *Function blocks – Part 1: Architecture*. Geneva: International Electrical Commission, 2005.

[7] R. Lewis, *Modelling control systems using IEC 61499 – Applying function blocks to distributed systems*. London, UK: The Institution of Electrical Engineers, 2001, iSBN 0-85296-796-9.

[8] F. Auinger, T. Strasser, and J. H. Christiensen, "Using IEC 61499 Function Blocks (FB) for Closed Loop Control Applications," in *International IMS Forum 2004, Villa Erba, Cernobbio, IT, 17-19 May 2004*, 2004.

[9] T. Strasser, F. Auinger, and A. Zoitl, "Development, implementation and use of an IEC 61499 function block library for embedded closed loop control," in *Industrial Informatics, 2004. INDIN '04. 2004 2nd IEEE International Conference on*, 2004, pp. 594–599.

[10] K. Oberhauser, "Development of a Measurement and Control Component Library for IEC 61499," Master's thesis, Vienna University of Technology, 2002.

[11] J. Lastra, A. Lobov, and L. Godinho, "Closed loop control using an IEC 61499 application generator for scan-based controllers," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, vol. 1, 2005, pp. 323–330.

[12] C. han Yang and V. Vyatkin, "Model Transformation between MATLAB Simulink and Function Blocks," in *Industrial Informatics (INDIN), 2010 8th IEEE International Conference on*, 2010, pp. 1130 – 1135.

[13] I. Hegny, T. Strasser, M. Melik-Merkumians, M. Wenger, and A. Zoitl, "Towards an Increased Reusability of Distributed Control Applications Modeled in IEC 61499," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 9 2012, pp. 1–8.

[14] A. Zoitl, T. Strasser, and A. Valentini, "Open source initiatives as basis for the establishment of new technologies in industrial automation: 4DIAC a case study," in *IEEE International Symposium on Industrial Electronics (ISIE)*, 7 2010, pp. 3817 –3819.

[15] A. Zoitl, *Real-Time Execution for IEC 61499*. Durham, North Carolina, USA: International Society of Automation, 2009.