An Engineering Method for Batch Process Automation using a Component Oriented Design based on IEC 61499

Wilfried Lepuschitz and Alois Zoitl
Automation and Control Institute, Vienna University of Technology
Gusshausstrasse 27-29, 1040 Wien, Austria
{lepuschitz, zoitl}@acin.tuwien.ac.at

Abstract

Flexible production systems represent key commodities for industrial companies in order to successfully persist on the global market. Batch processes and modifiable recipes already provide a certain flexibility concerning the design of production processes. However, the employed automation systems demonstrate deficits concerning reconfiguration and reusability.

IEC 61499 introduces new paradigms to the automation domain by offering generic architectures suitable for reconfigurable Distributed Control Systems.

This paper introduces an engineering method for process automation using a component based design and applied industrial standards to show the feasibility and benefits of designing a Distributed Control System based on the standard IEC 61499.

1. Introduction

Industrial companies are nowadays confronted with a difficult challenge on the global market. In order to succeed, they have to fabricate high-quality products based on consumer demands with short manufacturing periods at low costs. The industry therefore requires prospective technologies that offer support to cope with this challenge [1]. Multipurpose facilities providing flexibility are demanded to react quickly on changing marked demands [2].

Regarding the domain of process industries, the introduction of batch processes and modifiable recipes already brought certain flexibility at least from a process-oriented kind of view. Nevertheless the applied control systems are still based on classical automation concepts which do not necessarily bear resemblance to the physical and functional structure of the target system. These structural differences impede easy modifications concerning the system's functionality. However, a flexible production requires flexible automation systems that offer the possibility to reconfigure software components as well as hardware components [3].

Dutzler [4] introduces a Modular Distributed Control Architecture which follows the paradigm of object-oriented programming by encapsulating functionality and information into distinct entities. Combining these entities delivers functionality on a higher level. The system can be extended with further entities to provide additional functionality.

Newly adopted paradigms require the introduction or application of "new" standards. The IEC 61499 standard offers a generic architecture and guidelines for the use of software components, respectively Function Blocks (FB), in Distributed Control Systems (DCS) [5].

However, even though IEC 61499 is in general regarded as a possible successor to the standard IEC 61131, industrial enterprises hesitate to actually apply automation systems based on this promising standard. Resentments towards the implementation of IEC 61499 to the domain of process industry cannot be denied as was shown in a summer workshop with industrial and academic personnel [6]. Missing guidelines of how to actually use the benefits of this standard are considered to be a reason for the reluctance to employ IEC 61499 in Process Automation.

Evidently the necessity for defined guidelines for applying a DCS based on IEC 61499 in Process Automation exists.

This paper introduces a component oriented approach for process automation and an intuitive engineering method for the development of applications based on the introduced approach. According to the shown engineering method, a recipe based on the standard IEC 61512 [7] is converted into an IEC 61499 application.

2. Industrial Standards and Recent Developments in Process Automation

Even though new promising standards as IEC 61499 need time to be widely accepted, the role of industrial standards in general has undoubtedly been a valuable one. Numerous companies rely on the knowledge embodied in existing standards as they represent commonly agreed reference points and specifications. Companies that embrace those standards, or are even able to create *de facto* standards, can realize competitive advantages in comparison to their business rivals that lag behind [8]. It therefore appears reasonable to employ technological standards as key basic elements for introducing new

1-4244-1506-3/08/$25.00 ©2008 IEEE

concepts to any technological domain. This approach might raise their acceptance since companies tend to be reserved towards new concepts as a result of possibly high switching costs.

The following sections briefly introduce the utilized industrial standards. More detailed information about concepts and usage of these standards is presented in [9], [10] and [11].

2.1. IEC 61512 – Batch Control

The standard IEC 61512-1 *Batch Control – Part 1: Models and Terminology* provides reference models for batch control that can be used as a structural basis for the design of a DCS for process automation. Processes, physical equipment, control software and recipes are organized as hierarchical structures and relations between these structural models are defined.

The process model segments a process into the elements: process, process stage, process operation and process action. The physical model abstracts the physical organization of a batch manufacturing company into seven levels whereof only the four lower levels – process cell, unit, equipment module and control module – are considered as technically relevant. The procedural control software is segmented into the elements: procedure, unit procedure, operation and phase. Elements of the procedural control model are mapped onto those of the physical model to create so-called equipment entities (Figure 1). These equipment entities are labeled according to the physical model and provide process functionality based on the elements of the process model. Furthermore IEC 61512 distinguishes four types of recipes which are organized according to either the process model or the procedural control model.

Batch control systems currently employed on the market are based on the structural models of this standard, respectively its counterpart ISA S88.01 *Batch Control, Part 1: Models and Terminology*. This allows a certain grade of comparability and interoperability between these products [2]. Concerning any process-oriented activities, the presented structural guidelines of IEC 61512 are taken into account by the concepts introduced in this paper.

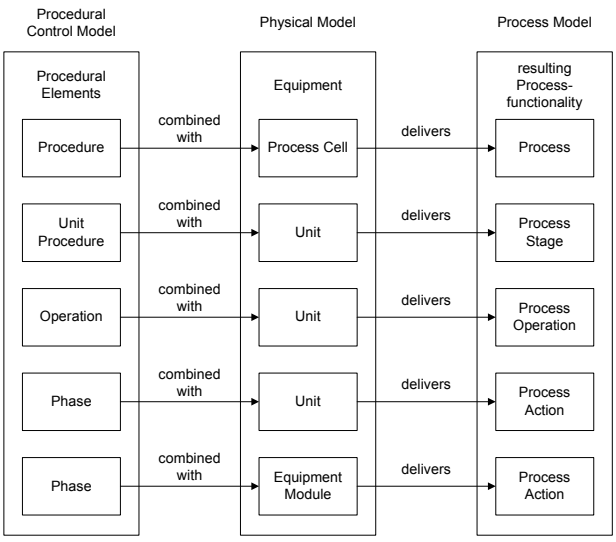

Figure 1. Structural models of the standard IEC 61512 and their relations [7]

2.2. IEC 61804

The standard IEC 61804 *Function blocks (FB) for process control* [12] specifies the requirements of distributed process control systems based on function blocks. It is applied in several process industries such as the petrochemical, pharmaceutical, or food and beverage domains.

Annex A of this standard describes a method for the development of control architectures. This method states that the requirements for function blocks used in a control system are key elements for the design activities. These requirements can be obtained in four steps:

- Process Flow Diagram (PFD),
- Extended Piping and Instrumentation Diagram (Extended P&ID),
- Control Hierarchy Diagram (CHD) and
- Functional Requirement Diagram (FRD).

Parts of this method are used and moreover modified according to the amended structural concepts of IEC 61512.

2.3. Recent Approaches for IEC 61499 in Process Automation

During the recent years, IEC 61499 has received an ongoing attention by (mainly academic) researchers regarding its implementation in discrete processes. However, the domain of process automation with batch and continuous processes has been treated more like a "red-headed stepchild" and only a few results concerning the applicability of IEC 61499 within this domain exist. Likewise to the approach presented in this paper, the following concepts are based on the structural models of IEC 61512.

Peltola *et al.* [13] convert a Procedure Function Chart's (PFC) hierarchical structure into a hierarchy of IEC 61499 composite function blocks. Each PFC element, regardless of its position in the hierarchy, is transformed into one function block and encapsulates the function blocks of lower-level PFC elements. The event connections between these function blocks represent the PFC element's sequential control flow. Certain event handling function blocks ensure a correct synchronization in the case of simultaneous activities and further function blocks realize other PFC elements such as time or conditional delays. The structural resemblance of this concept to the PFC's structure facilitates an intuitive design of an IEC 61499 application. A generic basic function block model is introduced that represents the

Procedural State Machine as it is described in the standard IEC 61512.

Thramboulidis *et al.* [14] provide a method for the transformation of PFC elements to function block networks by utilizing use cases and UML constructs. Each use case represents a unit of functionality that describes the behavior of the system to achieve a certain goal. These use cases can then be employed for the configuration and reconfiguration of the batch system. The next step towards obtaining a function block network is to convert the use cases into Object Interaction Diagrams (OIDs) based on UML models. These are afterwards automatically transformed into function block networks by using Corfu FBDK.

The described concepts introduce methods for the development of IEC 61499 applications. However, specific guidelines about the low-level software elements required to actually access the functionality of the physical equipment (sensors and actuators) are not presented in these concepts.

As batch production plants are required to possess the capability to execute different control recipes, the control system needs to be reconfigurable. However, it seems that the described concepts do not support dynamic reconfiguration without downtime as new function blocks are created with every change of the control recipe, respectively the PFC. In order to be executed, these function blocks need to be compiled, inhibiting therefore zero downtime.

3. The Engineering Method

The standard IEC 61512 introduces a general approach for the development of a batch production system. This approach is based on a dual iterative design method to determine the suitable equipment objects as well as the fitting procedural elements (Figure 2). Considerations of one determination activity influence the other one. Process-oriented considerations regarding the material treatment mark the main criterion for the determination of the necessary procedural elements that characterize the functionality of the corresponding equipment objects. The chosen procedural elements are then used to convert recipes into an executable application. However, equipment-oriented considerations concerning the provided functionality of equipment objects influence the range of possible procedural elements that can be developed and chosen from to convert the recipes. These equipment objects compose the production line which represents the physical environment for the execution of the application to produce a batch. To ensure flexibility for the creation of new recipes, the procedural elements should be developed in a manner to provide any available functionality of the equipment objects not regarding any already used recipes. According to IEC 61512 the development and maintenance of these procedural elements is a continuous activity to constantly improve the applied process technology.

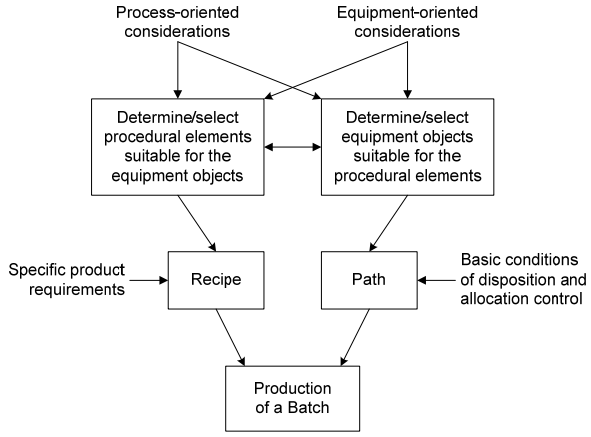

Figure 2. The engineering approach presented in the standard IEC 61512 [7]

Based on this general approach, we describe an intuitive engineering method using a laboratory process plant as the target system which will be introduced in the following section.

3.1. The Laboratory Process Plant

The FESTO compact unit acts as the target system for the presented control architecture. In general this laboratory process plant is used for training and educational tasks to communicate industry-related competences. It is composed of various industrial components which provide training possibilities concerning open-loop as well as closed-loop control technologies.

The engineering method presented in this paper is described on the basis of the FESTO compact unit to facilitate a better understanding. Figure 3 shows the laboratory process plant.

Five controllers from different vendors are used to control sensors as well as actors of the FESTO compact unit and to host the software components of the Distributed Control Architecture.

3.2. Amendments to the Structural Models and Concepts of IEC 61512

To achieve a consistent design of the Distributed Control Architecture, some amendments are made to the models and concepts introduced in the standard IEC 61512.

IEC 61512 defines the correlations between the procedural control model, the physical model and the process model according to Figure 1. Evidently, elements of different levels of the procedural control model and the physical model are combined to form elements of the process model. Inconsistencies occur especially regarding the unit level of the physical model as it is combined with three different elements of the procedural control model to form also three different process model ele-

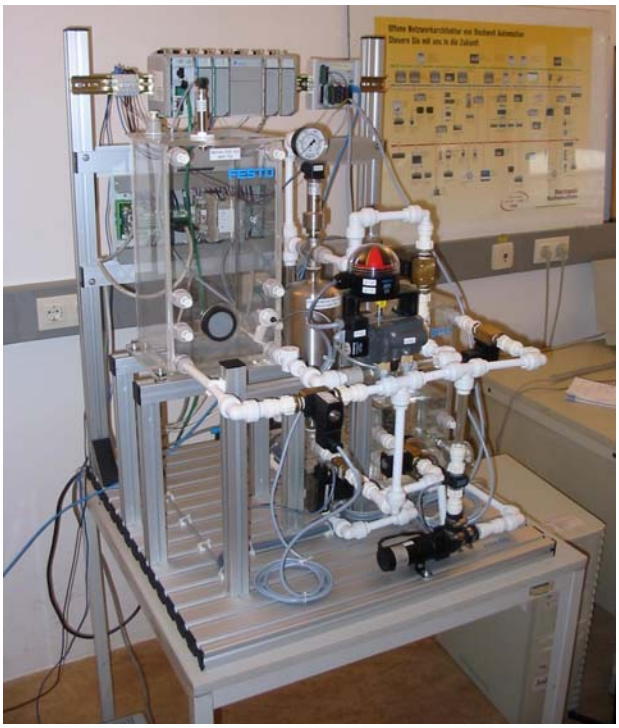

Figure 3. The laboratory process plant

ments. The approach of this paper only correlates corresponding elements of the same levels.

It is explicitly stated in this standard that the structural models may be diminished or extended as long as the consistency of each model remains assured.

Due to the physical limitations of the used equipment it appears to be reasonable to diminish the procedural control model by removing the element unit procedure and as a consequence to diminish the process level by removing the element process stage.

The definition of a phase of the procedural control model claims that it can be segmented into steps and transitions. This further aggregation shall be emphasized by introducing the step as the lowest-level procedural element. Regarding steps as the elementary components that can be combined to form higher-level elements facilitates the design process concerning the development of execution sequences. These steps are mapped onto control modules to provide functionality on a very low level. Therefore it is necessary to extend the process model by introducing the process step as the lowest element to maintain consistency throughout the structural models.

The resulting consistent structural concept (Figure 4) shall facilitate intuitive design activities concerning the Distributed Control Architecture.

3.3. Identification of the Structural Elements

The suggested engineering method utilizes the Extended P&ID as well as the CHD to identify and structure the elements of the process model and the physical

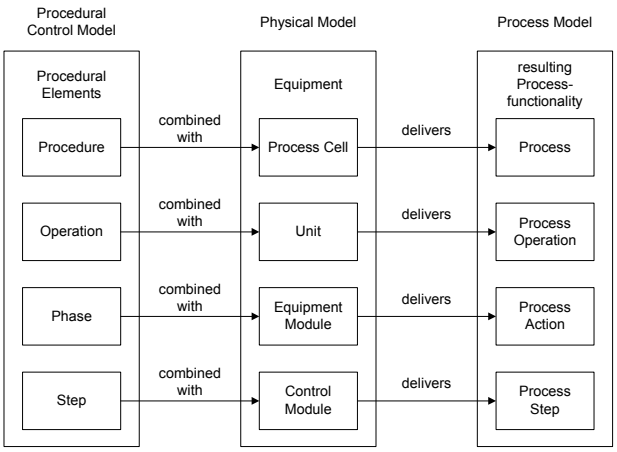

Figure 4. Amended structural models

model. These are afterwards used to develop the procedural elements necessary to access the physical equipment's functionality.

The Extended P&ID represents a graphical schematic view of a part of a process and displays the used equipment and correlated Control Functions. These Control Functions cover various tasks such as determining sensor values, controlling actuators and handling open-loop as well as closed-loop functions. The identification of Control Functions on different hierarchical levels leads to the determination of the process elements as well as the physical elements of the process cell. Applying a bottom-up approach delivers the newly introduced process steps and their corresponding control modules that provide the functionality. The aggregation of process steps leads to the identification of process actions that rely on equipment modules. Combining process actions provides the functionality of process operations that rely on units. Finally the aggregation of the identified process operations delivers the process itself that uses the complete process cell. The next step treats the creation of the framework describing the process elements after their identification.

Figure 5 shows a CHD which represents a hierarchical view of all identified process elements which are deemed necessary for a demonstration application. The CHD clearly displays the dependencies between the elements of the different levels. Once all required process elements are identified, corresponding procedural elements that can be executed on the physical equipment are created according to the component based architecture described in the following section.

4. Component Based Architecture

The standard IEC 61512 provides hierarchical structural models concerning the organization of a production facility. In fact, the physical model is applied in the industry as the technical equipment used in a facility is

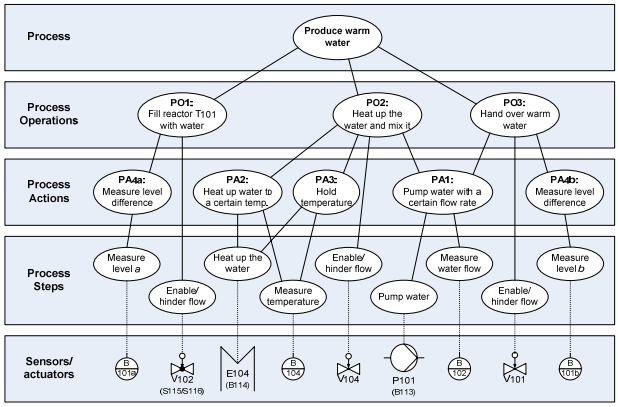

Figure 5. Control Hierarchy Diagram

indeed structured in a compositional manner. Machines are aggregated into machine groups and these are further aggregated to compose a complete production facility. Machines themselves can be segmented as well into sensors and actuators. This approach emphasizes the possibility to reuse components for different applications. Furthermore components are replaceable as long as their substitutes provide the required functionality and meet the necessary specifications.

This component based approach can be mapped as well to the architecture of the control system. The resemblance to the physical structure of the target equipment facilitates the development of a control system and allows very intuitive engineering efforts.

4.1. Concept of a Component

Sünder *et al.* [15] introduce a hierarchical control system composed of devices with a standardized architecture. Using a consistent structure for all devices regardless of their position within the control system shall facilitate an easier use of this concept. This applies for vendors of components respectively automation engineers that develop those components as well as end-users respectively process engineers that focus on the development of recipes. These devices are referred to as Automation Components (AC) and each comprises three parts [16]:

- Mechatronic components: sensors, actuators and electric circuits;
- An embedded control device: computing device with interfaces to the mechatronic components and to the network;
- Software components: data and control logic to carry out automation functions.

The structure of the software components can further be segmented into the following elements [3]:

- Logic: provides the basic control functionality of the component by combining inputs, outputs and inner device states.
- Diagnostics: performs diagnostic functions such as life-time diagnostic or basic error detection.
- Human Machine Interface (HMI): is used for interactions between operators and the device and is composed of two parts. One part represents the AC within an external visualization device (e.g. a handheld) while the other one is included in the universal component interface to provide a communication interface for the external part.
- Universal Component Interface (UCI): The UCI represents an interface for the communication of the component with the external HMI as well as higher-level components. Requests to the component are received by the UCI and sent to the logic or diagnostics part. Any responses of these parts are forwarded by the UCI to the HMI or higher-level components.

4.2. Hierarchical Structure

Dutzler [4] distinguishes between two types of ACs. One AC type interacts directly with the physical processes and can be referred to as a basic AC respectively intelligent sensor/actuator unit. The controller of a basic AC is able to execute complex tasks such as diagnostic functions independently of its superior control. Logic and diagnostics part of a basic AC directly access the physical equipment. For instance combining a ball valve with two position sensors delivers a basic AC. The ball valve itself is controlled by the Logic and provides the functionality to enable or hinder a water flow through a pipe. Whenever a superior control demands a change of the valve's state, the diagnostics part of the valve's software performs a position check by gathering data from the position sensors to verify the valve's position. An error event is created in case the valve does not reach the required position after a given amount of time. Basic diagnostic functions are therefore performed directly by these basic ACs.

Aggregated ACs represent the other component type and provide more abstract functionalities by encapsulating lower-level ACs and their functionality (Figure 6). Aggregated ACs communicate with the UCI of their encapsulated ACs and provide a UCI for any higher-level ACs to be accessed. Furthermore they possess additional functionality distributed within their logic and diagnostics. All executable and any coordination algorithms concerning lower-level ACs are part of the logic. The general information management as well as error and exception handling is performed by the diagnostics part. Figure 4 shows an aggregated AC composed of two basic ACs. For instance this could be a flow control comprising a pump and a flow sensor. A suitable control algorithm executed in the logic creates the control value for the pump according to a required reference flow value received by the UCI from a higher-level AC and the data gathered by the diagnostics part from the basic AC controlling the flow sensor. The created control value is then forwarded to the basic AC controlling the pump. A process cell that contains pumps, various valves

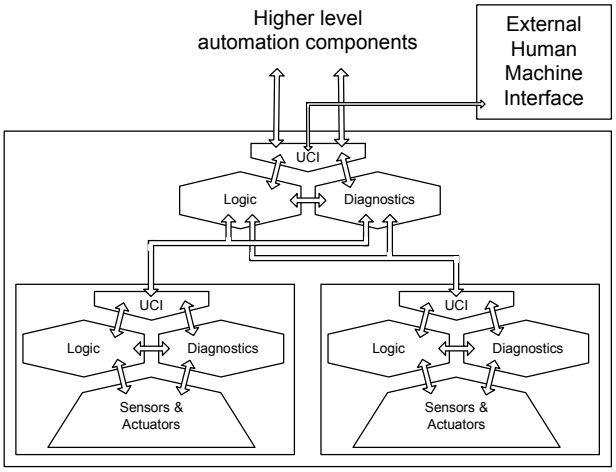

Figure 6. Architecture of an aggregated AC containing two basic ACs [3]

and sensors represents an aggregated AC on a larger scale but in fact this term can even be applied to a whole factory.

4.3. Embedding the ACs in IEC 61512

Basic ACs represent equipment entities on the control module level. They provide functionality on the process step level handling an actuator, a sensor or a combined unit with basic diagnostic functions.

Aggregated ACs represent equipment entities on the equipment module level or above. Therefore they realize more abstract functionalities on the process action level or above.

A component libary has to be created which contains procedural elements for all levels ranging from the topmost element, the procedure, down to the steps. The components provided in this library are used to convert a control recipe into an application. Reusability and flexibility are key attributes concerning the ACs. In order to enhance the reusability of software elements, they should be designed as hardware-independent as possible. For instance a software component for handling a valve should be capable of controlling different types of valves. Moreover these software elements need to be executed on different types of controllers.

The resulting application consists of a single hierarchical structure composed of ACs providing the complete control of the process cell.

5. Realization with IEC 61499

The component library contains all necessary procedural elements to compose an executable application. Apart from a few exceptions on the step level, all procedural elements comprise a logic and a diagnostics part as well as a UCI. To emphasize more reusability, the procedural elements themselves are created as hardware-independent as possible, nevertheless they need to be mapped onto physical entities in order to deliver complete ACs that provide the required functionality. Hence, for the definition of procedural elements that offer more specific functionalities on a higher level, the actual physical structure of the process cell has to be taken into account.

Each of the software elements (logic, diagnostics and UCI) is represented by a composite function block. As stated in the previous chapter, the applied component based architecture implies the aggregation of components to form higher-level elements. Concerning the software elements of the ACs, the aggregation is realized only on a functional level. The UCI of a lower-level component needs to be accessed by both logic and diagnostics part of one or even several higher-level components and can therefore not be integrated directly into a single higher-level element. The higher-level component's logic and diagnostics parts comprise solely certain service interface function blocks (SIFB) to access the lower-level UCIs. This design features the possibility to exchange components without the need to adapt the software of their superior components as long as the corresponding SIFB to access the changed component doesn't need to be modified or substituted. For example a magnetic valve can be replaced by a ball valve which offers the same functionality. Both types of valves can be accessed by one type of SIFB therefore the higher-level component is not affected by a change of the valve type. The SIFBs that need to be implemented in the software elements for each component can be determined according to the CHD created earlier.

5.1. Converting a Control Recipe into an Executable Application

Converting a control recipe into an application is achieved by determining the required elements of the component library according to the procedural elements of the recipe. These required elements are sequenced to form an application which is executed on a process cell to produce a batch. The conversion can be done on any level of the hierarchical structures.

Performing the conversion on the lowest level requires a control recipe which includes every single step that has to be carried out. Only procedural elements on the step level need to be provided which eases the creation of the component library. However, the process engineer responsible for creating the control recipe would be required to determine the exact sequence of steps to achieve the demanded functionality of the recipe.

If the conversion would be done on the highest level, a control recipe would only contain the request for one specific procedure. In this case, procedural elements of all levels have to be provided in regard of all certain possibilities to create recipes – which is basically impossible.

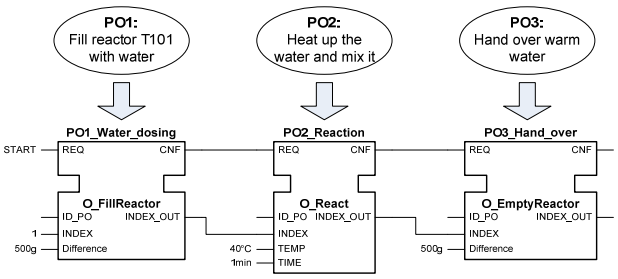

Figure 7. Process operations converted to an executable IEC 61499 application

A promising approach proves to be the conversion on either the operation or the phase level of the procedural control model. In these cases the component library needs to provide procedural elements up to the operation, respectively phase level. For the demonstration application, we decided to convert the control recipe on the operation level. Figure 7 shows the identified process operations and their corresponding IEC 61499 FBs. These FBs represent the service interface function blocks to access the UCIs of the automation components on the operation level.

Even though created extensively, a component library does not necessarily serve all possible recipes therefore considerations concerning new recipes may require new functionalities not yet integrated into the library. The component based architecture facilitates extensions since new ACs can easily be added to the library to enlarge the range of provided functionality.

6. Discussion of Results

The Distributed Control Architecture presented in this paper and its corresponding engineering method offer several gains for the process industry but nevertheless are faced with some problems as well.

The method used for the extraction of the functionality provided by the process cell is based on the standard IEC 61804 but modified in order to be compatible to the used structural models. This extraction approach combined with a component based architecture leads to a very intuitive method for the creation of software elements.

The actual implementation on the laboratory process plant reveals certain problems. The development of the software elements for the component library unveils a weakness of the employed component concept. The aim was to develop software elements that support two types of reusability. However, this aim is reached only partially.

Firstly, the intention was to develop software elements which are capable to handle more than just one type of specific sensor/actuator unit or grouping. However, software elements for higher-level components have to be designed specific with regard to the target equipment. On the contrary the created software elements on the step level can be used to handle several different sensor/actuator units. For instance both a ball valve and a magnetic valve can be controlled by the same type of software component. Phase level functionalities like a PID-controller are not bound to specific hardware at all and can therefore be created regardless of the target equipment.

Secondly, created software elements should be independent from any controller type they might be allocated to. However, software elements on the step level need to access the physical I/O ports of their host controller and therefore have to be designed specifically for this type of controller. On the contrary higher-level software components are not bound to specific controller types and can be executed regardless of the controller's hardware configuration.

7. Conclusion and Outlook

This paper explains the development of an IEC 61499 application for process automation. The approaches used as a basis for these development activities are not new but their combination delivers a very intuitive engineering method. Two parallel activities can be identified that influence each other.

On the one hand, considerations concerning the used equipment lead to the determination of the system structures that limit the possible range of applicable recipes. By identifying the functionalities on all levels, a component library with procedural elements based on ACs is created. Furthermore these elements are used to compose an application by converting the elements of the control recipe into corresponding IC 61499 FBs.

On the other hand, considerations concerning the process lead to the development of a control recipe which describes the methods that have to be applied in order to produce a demanded batch. However, the conversion of the control recipe into an application may reveal a lack of functionality. In this case, new elements have to be added to the component library to meet the requirements of the recipe. It may even be necessary to physically extend the process cell itself to provide the demanded functionality.

Combining both activities leads to the development of an IEC 61499 application providing the complete equipment control for the process cell (Figure 8). To finally produce the required batch, the application has to be executed on the corresponding process cell.

Though the applied Distributed Control Architecture provides a certain range of reusability, further research emphasis has to be laid on the concepts concerning the instance specific parameterization of the ACs. However, the integration of basic ACs could already be simplified by providing complete intelligent sensor/actuator units with software elements on the step level to handle the hardware elements.

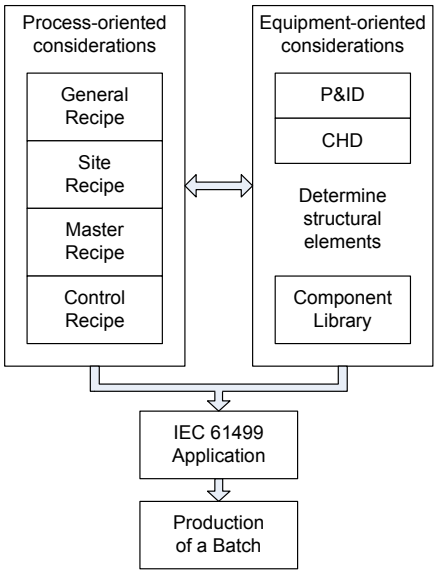

Figure 8. Structure of the engineering method

The concept of the ACs presented in this paper uses composite FBs to represent the three different software elements (logic, diagnostics part and UCI). Using subapplications instead might lead to a better structuring of these elements. However, this approach has yet to be investigated [17].

Downtime-less reconfiguration of an automation system for a batch production plant represents a key ability to enable the execution of different control recipes on a single process cell. The approaches shown in this paper represent a possible path to advance towards this aim by integrating IEC 61499 to the domain of batch processes based on a concept of predefined components in a library that can be used to convert control recipes into executable applications.

References

[1] B. Favre-Bulle, and G. Zeichen, *Zukunft der Forschung in den Produktionswissenschaften*, Verein zur Förderung der Modernisierung der Produktionstechnologien in Österreich - VPTÖ, 2006.

[2] S. Kuikka, *A batch process management framework: Domain-specific, design pattern and software component based approach*, PhD thesis, Helsinki University of Technology, 1999.

[3] C. Sünder, A. Zoitl, and C. Dutzler, "Functional structure-based modelling of automation systems," in *Int. J. Manufacturing Research*, Vol. 1, Iss. 4, pp. 405-420, 2006.

[4] C. Dutzler, *A Modular Distributed Control Architecture for Industrial Automation Systems – Designed for Holonic Systems and Vertical Integration*, PhD thesis, Vienna University of Technology, 2003.

[5] IEC TC65, *IEC 61499: Function blocks for industrial-process measurement and control systems – Parts 1 to 4*, International Electrotechnical Commission (IEC), 2004-2005.

[6] M. Ströman, S. Sierla, J. Peltola, and K. Koskinen, "Professional designers' adaptations of IEC 61499 to their individual work practices," in *Proc. ETFA '06 IEEE Conference on Emerging Technologies and Factory Automation*, pp. 743-749, 2006.

[7] IEC TC65, *IEC 61512-1: Batch control – Part 1: Models and terminology*, International Electrotechnical Commission (IEC), 1997.

[8] K. Lyytinen, and J. L. King, "Standard Making: A Critical Research Frontier for Information Systems Research," in *MIS Quarterly*, Vol. 30, Special Issue on Standard Making, pp. 405-411, 2006.

[9] M. Barker, and J. Rawtani, *Practical Batch Process Management*. The Netherlands: Newnes, 2005.

[10] V. V. Vyatkin, *IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design*. USA: ISA, 2006.

[11] R. Lewis, *Modelling control systems using IEC 61499*. UK: The Institution of Electrical Engineers, 2001.

[12] IEC TC65, *IEC 61804-1: Function blocks (FB) for process control – Part 1: Overview of system aspects*, International Electrotechnical Commission (IEC), 2006.

[13] J. Peltola, J. H. Christensen, S. Sierla, and K. Koskinen, "A Migration Path to IEC 61499 for the Batch Process Industry," in *Proc. 5th IEEE International Conference on Industrial Informatics*, pp. 811-816, 2007.

[14] K. Thramboulidis, S. Sierla, N. Papakonstantinou, and K. Koskinen, "An IEC 61499 Based Approach for Distributed Batch Process Control," in *Proc. 5th IEEE International Conference on Industrial Informatics*, pp. 177-182, 2007.

[15] C. Sünder, A. Zoitl, M. Rainbauer, and B. Favre-Bulle, "Hierarchical Control Modelling Architecture for Modular Distributed Automation Systems," in *Proc. 4th IEEE International Conference on Industrial Informatics*, pp. 12-17, 2006.

[16] V. V. Vyatkin, "Intelligent mechatronic components: control system engineering using an open distributed architecture," in *Proc. ETFA '03 IEEE Conference on Emerging Technologies and Factory Automation*, pp. 277-284, 2003.

[17] C. Sünder, A. Zoitl, J. H. Christensen, M. Colla, and T. Strasser, "Execution Models for the IEC 61499 elements Composite Function Block and Subapplication," in *Proc. 5th IEEE Int. Conference on Industrial Informatics*, pp. 1169-1175, 2007.