Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 72 (2018) 1088–1093

www.elsevier.com/locate/procedia

51st CIRP Conference on Manufacturing Systems

State-based control of process services within modular process plants

Henry Bloch[a,*], Stephan Hensel[b], Mario Hoernicke[c], Katharina Stark[c], Anna Menschner (neé Hahn)[b], Alexander Fay[a], Leon Urbas[b], Torsten Knohl[d], Jens Bernshausen[e]

[a]Institute of Automation Technology, Helmut-Schmidt-University, Holstenhofweg 85, 22043 Hamburg, Germany
[b]Chair of Process Control Systems and Process Systems Engineering Group, Technische Universität Dresden, 01062 Dresden, Germany
[c]ABB Corporate Research Opertions Management, Wallstadter Straße 59, 68526
[d]Bayer AG, 51386 Leverkusen, Germany
[e]INVITE GmbH, 51386 Leverkusen, Germany

* Corresponding author. Tel.: +49 40 65412816; fax: +49 40 65412004. *E-mail address:* henry.bloch@hsu-hh.de

Abstract

The modularization of process plants is an approach to cope with the changing requirements of the process industry, such as highly volatile markets and shorter product lifecycles. Each module offers services as encapsulated process functions to the superior control system for the orchestration. These services are controlled in a state-based way. The encapsulation of these services in a reasonable way is a difficult task, because aspects of process and automation engineering must be considered. This contribution introduces an approach for the encapsulation of process functions in services from an automation point of view. Furthermore, the operation modes of modules, services and field devices are discussed. Additionally, possible solutions for the orchestration based on recipe models are introduced. The developed concepts have been tested within a simulation environment.

Peer-review under responsibility of the scientific committee of the 51st CIRP Conference on Manufacturing Systems.

Keywords: Modular process automation; Microservices; Orchestration; State-based control, Operating modes, Service-oriented architectures

1. Introduction

The modularization of process plants is seen as a feasible approach to face current challenges in the global process industries such as shorter product lifecycles and high-volatile markets. These challenges result in the need of a flexible production system and a shorter time-to-market [1]. Projects such as the *F3-project*, demonstrated already the applicability of modularization approaches in the field of process technology and the resulting advantages [2]. To achieve a fully modularized plant, the automation system beside the modular process equipment, needs to be modularized. A first approach for the modularization of the automation system was shown in the project *DIMA, decentral intelligence of modular process plants* [3].

Different modules which offer process functions are combined to a modular process plant. Thereby, each module runs its services on an own controller and offers the services via an OPC UA server to a superior control system (SCS). In the SCS the services are orchestrated to achieve the required production process [4]. The presented approaches [2, 3] did not face the orchestration of services of modules so far.

This contribution presents the workflow of the encapsulation of services from an automation perspective and the state-based control of services. Additionally the authors present an approach of the orchestration of services. Furthermore, a first approach of operating modes and the changes between the operating modes of the services are presented.

The contribution is structured as follows: In Section 2 an overview of the general setup of modular process plants as well as the approach of the encapsulation of services is presented. In Section 3, the state-based control of services as well as the orchestration of services is presented, followed by the approach of operating modes of modular process plants in Section 4. The presented approaches are implemented in a simulation

Peer-review under responsibility of the scientific committee of the 51st CIRP Conference on Manufacturing Systems.
10.1016/j.procir.2018.03.037

environment and tested in a case study, which is presented in Section 5. The contribution is concluded in Section 6.

2. Modular process plants

2.1. General setup of modular process plants

The general structure of modular process plants is shown in Fig. 1. Beside modules, a backbone is part of the modular plant. The backbone provides energy, the plant network and the superior control system [5]. All modules are connected to the backbone by an information connection and a power connection and to other modules by process connections, e.g. piping. Each module is equipped with its own controller which runs an OPC UA server and controls the field devices of the module. This concept of modules including their own controller and communication server has been defined by the NAMUR (User Association of Automation Technology in Process Industries, www.namur.net). The process functions of the module are offered to the SCS as services. The modules of the plant can be of different module vendors, as the functionality of the module is described in a vendor-independent model, the Module Type Package (MTP) according to VDI/VDE/NAMUR 2658 [6]. One part of the MTP describes the services which are offered by the module. In these services the automation of the module's field devices and the interaction of the field devices is encapsulated. Thereby the process function can be started by the SCS without the need of the control of single field devices. Services are the core element of modular process plants, because they represent a new layer within the automation hierarchy. In Fig. 2 the communication setup of modular process plants is shown. The SCS as part of the backbone contains the human machine interface of the modular plant and the orchestration unit. The orchestration unit communicates with the services which control the field devices within the modules. A direct influence from the SCS to the field devices during the operating mode *Auto* (see Section 4) is not possible, which is the biggest difference to classical process plants. The indirect addressing of field devices by the means of services, ensures the integrity of the module and protects the know-how of the module vendor. The suitability of service-oriented approaches for modular process plants has been shown by the authors already in [7, 8].

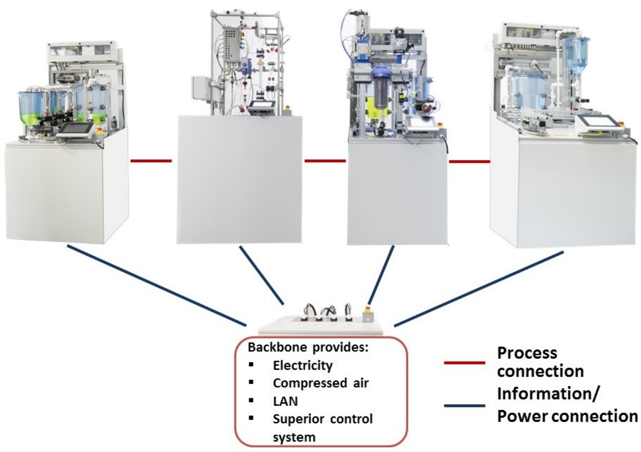

Fig. 1 General setup of modular process plants

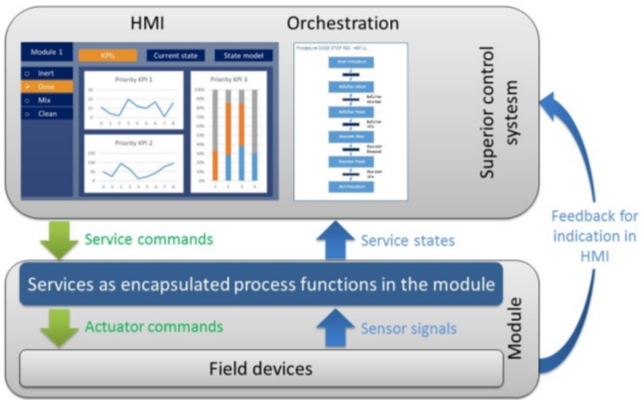

Fig. 2 Communication setup of modular process plants

2.2. Encapsulation of process functions as services

The engineering of modular process plants is divided into two different phases: i) the module engineering and ii) the integration engineering. In the first phase the module vendor designs and builds the module, in the second phase the plant owner combines different modules to a modular process plant. The interface between the two phases is the MTP, as a vendor independent description of the module [6].

In the first phase of the engineering, the module vendor has to design the services of the module. Therefore, basic process functions in reasonable combinations should be encapsulated as a service. Therefore different aspects need to be taken into account, such as process design aspects, device aspects and automation aspects. As it is hard to define which way of encapsulation and combination is reasonable, this contribution faces the encapsulation from an automation point of view. This encapsulation workflow covers five aspects, which are explained in the following subsections.

2.2.1. Choice of the granularity of services

First, the granularity of the service is determined by the process function of the service, e.g. *dosing*. If the process function of the module is a combination of fine granular functions, the functionalities of the smaller services must be determined. Therefore, the granularity of the services can be set from coarse grained until fine granular. For example, a *mixing*-service could be a combination of the finer granular services *filling, stirring* and *emptying*. The authors suggest to follow the design of batch-phases in discontinuous process plants for the choice of granularity of services [9]. This does not imply the approach is only suitable for discontinuous processes. For continuous production, the services need to be allocated to the production phases, such as s*tartup, continuous production* and *shutdown*.

Additionally, a service should run mostly independent from other services. Interdependencies between services, so-called service relations, should be avoided whenever possible. Service-relations have two different sources: a) Services which occupy the same physical equipment within a module and b) necessary ordered process steps. The source a) is referred to as shared equipment. To avoid shared equipment the services should be designed is such a way, that each equipment is assigned to only one specific service. The source b) is unavoidable in the process industry. To obtain the required product, a defined order of process steps has to be executed. This order is mainly defined by the orchestration of the services, see section 3. Beside the orchestration the modules itself define specific orders of services, e.g. a rendering inert has to be completed first, before any other service can be started. This process related type of service relations cannot be avoided. The reduction of service-relations follows also the concept of Microservices, which suitability for modular process plants is shown in [8]. Whenever a service relation is determined by the module vendor, the degree of freedom within the orchestration of the modular process plants decreases. All service relations must be taken into account in the orchestration of services, otherwise the liability for the module's safe functionality of the module vendor might expire.

2.2.2. Choice of the service-type and mandatory service states

After the functionality of the service has been set, the module vendor has to decide if the service should be a *self-completing service* or *continuous service.* These two service-types differ in the way of completion of the determined functionality of the service. A *self-completing service* ends if a set parameter or process value is reached, whereby a *continuous service* ends when the service receives a command to stop or abort. For a better understanding the state model for services according to IEC 61512 ed. 2 [10] has to be taken into account, see Fig. 3. Within the joint working group 2.3.1 of NAMUR and ZVEI (German Electrical and Electronic Manufacturer's Association, www.zvei.org) this state model was specified for services in modular process plants.

Each service follows the state model and within each state a program is executed which controls the field devices, which are needed for the execution of the functionality of the service, further details in section 3.2. The state model consists of the mandatory states: *Idle, Running, Aborting* and *Aborted*. These states represent an initial state, the state of the execution of the main function of the service and two states to abort the services. The state *Complete* is only part of *self-completing services*. This state represents that the service reached the defined parameter, e.g. process time, or a process value, e.g. temperature.

2.2.3. Choice of optional service states

Loops for *Pausing, Holding* and *Stopping* can be added to the mandatory state model for services which are more complex. Whenever loops are part of the state model, the whole loop must be implemented to ensure the compatibility with the SCS. A loop consists of a transient state, which are all states with an–*ing*-ending, and the following waiting state, which can be *Paused, Held*, *Stopped* and *Aborted* [6]. The loop to hold the service is an exception, as it is a combination of two transient states, *Holding* and *Unholding,* and the waiting state *Held*.

2.2.4. Determination of service parameters

Additionally, the needed service parameters must be determined by the module vendor. All parameters which can be adjusted during the runtime of the plant or should be indicated in the human machine interface (HMI) need to be described in the MTP. For each service state which uses service parameters, information regarding the utilized parameters, such as the range, the unit, and a default value, must be described in the MTP. The default value will be used by the service state whenever the operator does not set a value for the parameter.

2.2.5. Assurance of safe states

To ensure that the modules can always be set into a safe state by the SCS, the module has to be designed fail-safe. This requirement could result in an over-instrumentation within the modules, because in- and output-valves are needed in the main process connections. Furthermore, the services should be

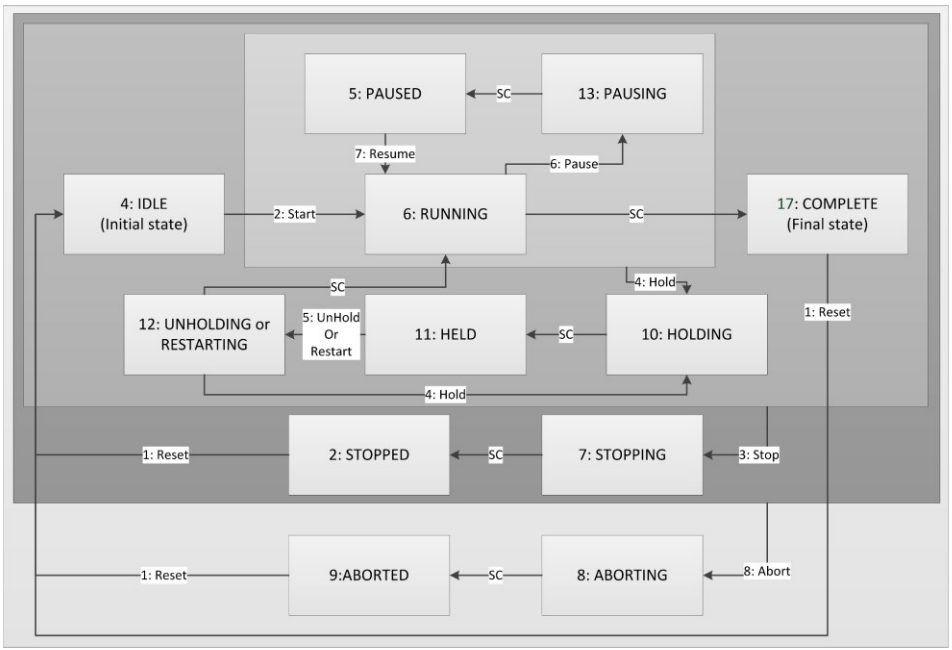

Fig. 3 State model of services according to IEC 61512 ed. 2 [10]

designed in such a manner, that each module can be set into a safe-state without any restrictions to other modules of the plant.

3. State-based control of services in modular process plants

3.1. Recipe-based control of classical process plants

In IEC 61512 [9] four different types of recipes are defined, which are common in the process industries: General recipe, site recipe, master recipe and control recipe. The goal of the usage of recipes in different layers of abstraction is the reuse of the recipes. By this approach, production processes can be executed on different technical equipment and at different plants. By the hierarchical structure of the recipes, the tracing of the root of the recipe is always ensured. The recipes give a description which educts are used, which requirements are set for the equipment and which chemical and physical shifts have to be executed to achieve the requested product. Therefore, the control recipe, as the most detailed recipe, refers to all actuators and sensors in a specific plant.

3.2. Recipes as basis for the orchestration of services

In modular process plants the control of the plant is based on the approach of control recipes. As the access to sensors and actuators in modular process plants is not allowed in general (see. Fig. 2), the orchestration has to base on the services of the modules. Fig. 4 gives an overview of the structure of the orchestration for modular process plants, here an extract of the orchestration for the NAMUR example case is presented; the case is presented more in detail in Section 5. The orchestration is structured in three different levels based on ISA 106 [11]: a) the steps of procedures; b) the procedures and c) the orchestration phases.

As stated in the section before, services are controlled by the means of a state model, see Fig. 3. Each state and each service command is numbered to ensure a unique identifier for each service state and service command. Beside the numbered transitions in the state model, the model contains *SC* (state change) transitions. These transitions are triggered by an event which is caused by the service itself. For example, after the state *Stopping* the transition to *Stopped* is a *SC*-transition; this transition is caused by the end of the program in the state *Stopping*.

The service transitions which are numbered are triggered by the orchestration unit in the SCS or by other services of the same module. Whenever a service requires another service to change to a specific state, e.g. combined services as more complex process function, a module-internal signal is sent, which triggers the transition. The service commands from the orchestration unit are set within the steps of the procedures. The service commands are set in the states of these steps (light-blue squares in Fig. 4) and the service states are queried at the transitions of these steps (dark-blue in Fig. 4). A sequence of steps is named procedure. Each procedure is allocated to a specific module of the plant as the resource of the procedure.

One abstraction layer above the steps of a single procedures the procedures are set into a specific order. Procedures can be executed in a synchronous or in a sequential manner, as phases within batch environments [9]. Fig. 4 gives an example how the procedures can be set in a synchronous and partly in a sequential order. Thereby, each horizontal line represents the beginning or the end of a synchronization of procedures, the part of synchronization is always between two horizontal lines. Furthermore, the procedures can be composed to orchestration phases. These phases represent the overall status of the production program. Typical phases for a continuous production process are: *Startup, continuous production* and *shutdown* (see subsection 2.2.1). These phases can be controlled, e.g. started, paused or aborted, in the same manner as recipes are controlled in classical process plants. The difference to modular process plants is the addressing of services instead of field devices.

For a better understanding of the communication between the orchestration and the services, Fig. 5 shows an interaction example of the SCS, a service and a field device. The SCS invokes a service by the command *Start.* This results in a state change of the service to the state *Running*, which is also communicated as feedback to the SCS. Within this service state the internal program of the service, executed on the module's controller, send the command to open *valve Y001*. The feedback of the field device results in a state change of the service, which is communicated to the SCS. Afterwards the service is *reset* in a second orchestration step. The orchestration steps could be in the same procedure, but do not have to be of

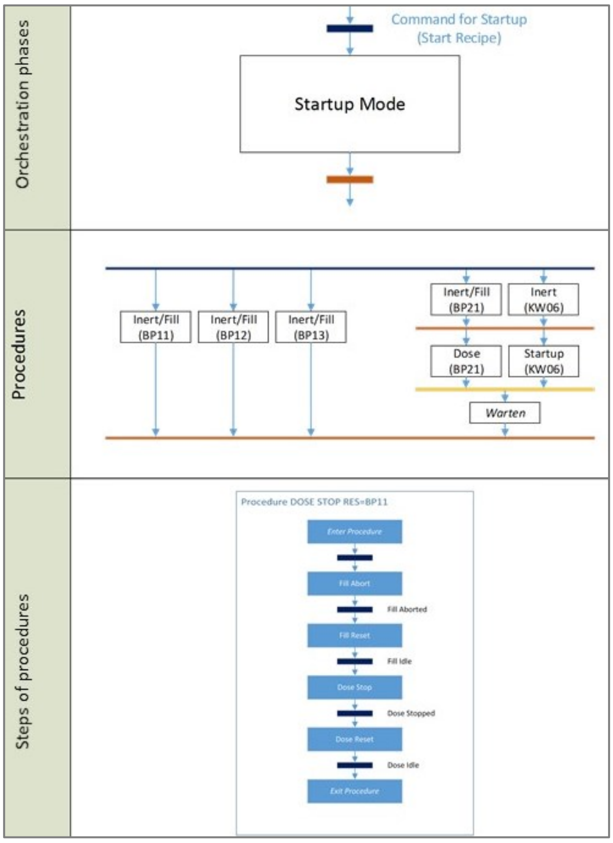

Fig. 4 Structure of the orchestration of modular process plants

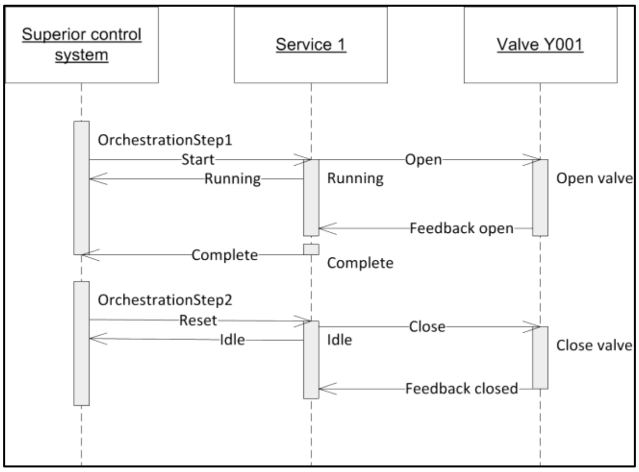

Fig. 5 Example of the communication between the SCS, services and a field device

the same procedure. Different procedures can invoke service state changes within the same orchestration phase.

4. Operating modes of services

As in classical process plants, also modular process plants can be operated in different operating modes. Here, not a module is set into an operating mode, but the services within the modules. Whenever a service is set to a different operating mode, the service sets the occupied field devices of this specific service also in the distinct operating mode.

Following the concept of PackML according to [12] multiple state models, one for each operating mode, are allocated to the services. The mandatory operating modes for modular process plants are as follows:

a) *Auto*: The service commands are set in the orchestration unit of the SCS. The production follows the defined sequence of procedures of the orchestration unit.

b) *ManualCentral*: The service commands are set by the operator by the means of the central HMI of the plant. Additionally, the module vendor can define specific field devices in the MTP which can also be operated manually by the operator.

c) *ManualLocal*: The service commands and the commands for the within the MTP defined field devices are set locally at the module. The control could be operated via a local control panel, a mobile device or a key at the field devices itself.

More operating modes may be specified in the standardization process of the MTP by the working groups of NAMUR, ZVEI and VDI/VDE GMA (The Association of German Engineers/Association of Measurement and Automation Technology, www.vdi.de/gma) [6].

To change between the operating modes, a service has to follow a transition from one state model to another. Therefore, the module vendor has to define in which states of each operating mode a safe change to another operating mode can be executed. Therefore, the module vendor must take the fail-safety of the module (see subsection 2.2.5) into account.

To change between the state models in defined states, changes are only allowed between waiting states. In transient states the status of the field devices and the module in total is not distinct. If the operator invokes a change of the operating mode, while the service is in a transient state, the service rejects the change-request and informs the operator by a message about the rejection and the current state of the service. The design of the SCS defines, if the change-request expires or if the request stays retained until a request is possible.

5. Case study

Within the working groups of NAMUR and ZVEI an example case of a continuous operating modular process plant has been set up. The plant consists of six modules of three different types: four dosing modules (named BP), a mixing and reacting module (CM05) as well as a distillation module (KW06). Each BP module offers the services *Inert, Fill* and *Dose*. The CM05 module offers the services *Heat* and *Run* and the KW06 module offers the services *Inert, Start-up* and *Shutdown*.

The authors implemented this example case in a simulation environment. The setup of the simulation environment is shown in Fig. 6. The simulation environment is divided into two parts: In Matlab Simulink Stateflow the services and the orchestration unit are implemented; while the visualization is implemented in a platform independent web application based on Foundation for Apps (foundation.zurb.com) and NodeJS (www.nodejes.org). The communication between the two parts is implemented by an OPC UA server which is wrapped into a OPC DA server.

Within the example case, a description of sequences which are typically implemented in sequential function charts (SFC) following IEC 61131-3 [13] is given. These SFCs were taken as process functions and encapsulated into services. Thereby the presented aspects were considered. As the example case does not include physical modules, the aspect of over-instrumentation for the assurance of safe-states (see subsection 2.2.5) could not be validated. But the process and instrumentation diagrams of the modules did not show in- and output valves in all modules. The module CM05 would need at least two input valves.

All services of the example case have been implemented in Matlab as grouped state charts, which have the service command as input and service state as output of the chart. In the orchestration unit, a single orchestration is implemented so far. The orchestration is also implemented as a state chart, with the service states as input and the service commands as output of the chart.

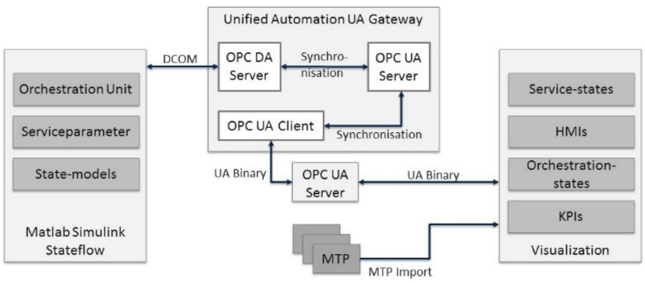

Fig. 6 Structure of the simulation environment [4]

Within the visualization, different information is provided to the operator, such as key performance indicators for each module. Additionally, a plant overview, the orchestration and the services are visualized. As this contribution focusses on the services, their visualization is introduced more in detail.

A screenshot of the service visualization is given in Fig. 7. All current available services of the selected module BP11 are listed on the left side of the visualization. Symbols represent the current state of the service, e.g. a square represents the state Stopped. On the right side, the state model of the selected service is shown. In Fig. 7 the service *Inert* is in the state *Running*, all possible commands which are allowed in this state are shown on the right side. Additionally, the next expected state transition, here self-complete to Complete; is shown in the bottom part. In the very top three different tabs are located which allow to change the view between the modules, the KPIs and the current state.

Through the visualization, the service models in Matlab can be controlled. Furthermore, the visualization can import MTPs for an automatic generation of the visualization. The authors described the automatic generation more in detail in [4].

The implementation of the NAMUR example case in the simulation environment shows the applicability of the state-based control of services for modular process plants.

Fig. 7 Visualization of services in the simulation environment

6. Conclusion

This contribution has presented an approach for the state-based control of services in modular process plants. Five aspects for the encapsulation of process functions as services have been introduced and described in detail: the choice of the service granularity, the choice of the service type, the determination of required service states, the determination of required service parameters and the assurance of safe states. Furthermore, an approach for the state-based control of services has been presented. Therefore, an approach for the structure of the orchestration in the superior control system has been introduced. Additionally, the interaction of the orchestration, services and field devices has been presented. Based on the orchestration the operation modes *Auto, ManualCentral,* and *ManualLocal* for modular process plants have been defined.

The applicability of the presented approaches has been shown in an example case which was implemented in a simulation environment. In summary, the contribution has presented approaches to fulfill a flexible automation system for modular process plants.

In future work, the authors will develop a service relation concept and innovative concepts for the visualization of modules and services in the central HMI of modular process plants. Beside these topics, all authors contribute actively to the standardization process of the Module Type Package in several working groups of NAMUR, ZVEI and VDI/VDE GMA.

References

[1] Bieringer, T.; Bramsiepe, C.; Brand, S.; Brodhagen, A.; Dreiser, C.; Fleischer-Trebes, C.; Kockmann, N.; Lier, S.; Schmalz, D.; Schwede, C.; S., A.; Stenger, F. Modular Plants - Flexible chemical production by modularization and standardization - status quo and future trends. ProcessNet Whitepaper, Frankfurt am Main, Dec. 2016.

[2] F3 Factory. [Online] Available: http://www.f3factory.com/scripts/pages/en/home.php.

[3] Holm, T.; Obst, M.; Fay, A.; Urbas, L.; Hempen, U.; Albers, T.; Kreft, S. Engineering method for the integration of modules into fast evolving production systems in the process industry. In 2015 IEEE International Conference on Automation Science and Engineering (CASE), Gothenburg, Sweden, 2015, pp. 1042–1047.

[4] Bloch, H.; Hensel, S.; Hoernicke, M.; Hahn, A.; Fay, A.; Urbas, L.; Wassilew, S.; Knohl, T.; Bernshausen, J.; Haller, A. Model-based Engineering of CPPS in the process industries. In 15th IEEE International Conference on Industrial Informatics, Emden, Germany, 2017.

[5] NE 148 Automation Requirements relating to Modularisation of Process Plants, 2013.

[6] VDI/VDE/NAMUR 2658-1 Automation engineering of modular systems in the process industry - General concept and interfaces, 2017, Greenprint.

[7] H. Bloch, A. Fay, and M. Hoernicke. Analysis of service-oriented architecture approaches suitable for modular process automation. In 21th IEEE Conference on Emerging Technologies and Factory Automation (ETFA): September 6 - 9, 2016, Berlin, 2016.

[8] Bloch, H.; Hoernicke, M.; Hensel, S.; Hahn, A.; Fay, A.; Urbas, L.; Knohl, T.; Bernshausen, J. A Microservice-Based Architecture Approach for the Automation of Modular Process Plants. In 22nd IEEE International Conference on Emerging Technology and Factory Automation, Limassol, Cyprus, 2017.

[9] IEC 61512-1 Batch Control - Part 1: Models and Terminology, 1997.

[10] IEC 61512-1 ed. 2 Batch Control - Part 1: Models and Terminology, unpublished draft.

[11] ISA-TR 106.00.01 Procedure Automation for Continuous Process Operations, 2013.

[12] ANSI/ISA-TR88.00.02-2015 Machine and Unit States: An implementation example of ANSI/ISA-88.00.01, 2015.

[13] IEC 61131-3 Programmable controllers - Part 3: Programming languages, 2013.