Towards OPC UA as portable SOA Middleware between Control Software and External Added Value Applications

Martin Melik-Merkumians[1], Thomas Baier[2], Michael Steinegger[1],
Wilfried Lepuschitz[1], Ingo Hegny[1], Alois Zoitl[1]

[1]Vienna University of Technology – Automation and Control Institute
Gusshausstrasse 27-29/E376, AT-1040 Vienna, Austria

[2]MicroSys Electronics GmbH
Mühlweg 1, DE-82054 Sauerlach, Germany
E-mail: melik-merkumians@acin.tuwien.ac.at

Abstract

Automation systems (ASs) based on Service-Oriented Architecture (SOA) are becoming more wide-spread. Due to their inherent property of separating the hardware-specific implementation from the logical work- or control-flow, SOA-based application programming is regarded as the next logical step in AS programming. Several successful attempts for implementing SOA-based AS have been made, but always with non-industrial, non-standard software like Devices Profile for Web Services. Based on the requirements of typical industrial SOA-based AS application, we will show that the industry standard data acquisition software OPC Unified Architecture is capable to serve as a SOA-based middleware.

1. Introduction

Customer demands are changing from mass-production to mass-customization and ever shorter product cycles. Therefore flexible, modular, inter-operable and robust automation systems (ASs) are becoming essential. Vertical integration of ASs with Enterprise Resource Planning (ERP), Manufacturing Execution Systems (MES), Supervisory Control and Data Acquisition (SCADA), and other added-value applications (AVAs) (e.g., diagnostic systems, batch track & trace) is an enabler for such flexible ASs. Common approaches to vertical integration usually separate the AS into several layers. The resulting structure is usually called the automation pyramid and is shown in Fig. 1(a). Although this is a feasible approach, this usually means that all requests (e.g., invocation of functions, data acquisition) have to be transferred from one layer to another, each time passing through (usually) another proprietary interface with all the associated (and sometimes error-prone) data translations. Service-Oriented Architecture (SOA)-based approaches provide a convenient way to integrate ERP, MES, SCADA, and other AVAs directly with the control level, all through one common interface as depicted in Fig. 1(b).

Originating from the field of Information Technology, SOA approaches are utilized to establish a common interface between several application layers via standardized (web-)services. Each service is defined by its inputs and outputs, and the service contract. A service contract specifies first its pre-conditions, which have to be fulfilled, second the functionality of the service, and third the guaranteed post-conditions of the service, also known as Hoare triple [4]. Szyperski [20] suggested that execution time and space requirements of services shall also be part of service contracts, since changing execution time and space could break the functionality of subsequent services and by thus violating their contracts. As long as the service contract is not violated, the exact service implementation can be hidden from the service-user [1].

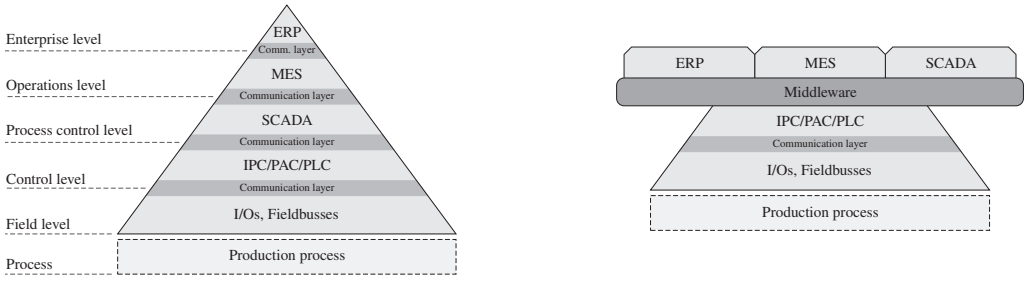
The EU projects SIRENA [1] and SOCRADES[1] proposed, implemented, and validated the feasibility of SOA-based AS approaches. Bohn et al. [1] evaluated several base technologies for a control level SOA interface, based on the following factors: plug and playability, device support, programming language independence, network media independence, large scalability, security, and high market acceptance. Based on these factors Devices Profile for Web Services (DPWS) was chosen as base technology for the SIRENA framework. But whereas SIRENA and SOCRADES used DPWS as its base technology, we analyze the eligibility of OPC Unified Architecture (OPC UA) as a base technology for SOA-based ASs.

OPC UA (IEC 62541) also meets the requirements imposed by [1] and is an industry (driven) standard which already is implemented in many controllers. Key points of OPC UA are

Standardization: is driven by an industry group of experts, and therefore is also implemented for many AS;

[1]SOCRADES - Service-oriented cross-layer infrastructure for distributed smart embedded devices (life span of the project: 2006 – 2009)

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640



(a) **traditional layered automation pyramid** (b) **SOA-based approach**

Figure 1. The automation pyramid: (a) traditional levels [16, p. 25] and (b) SOA-based approach.

Portability: OPC UA runs on the AS. The memory issues (as criticized in [2]) are overcome by the highly modularized approach. If the device cannot provide OPC UA services as required, a proxy device (PC or similar) can be used which still makes OPC UA at least equivalent to all other practical solutions on the market;

Efficiency: eliminates the need for proprietary communication drivers/protocols;

Security: secures the communication channel with state of the art mechanisms (e.g., SSL);

Extensibility: can be adapted to the needs of a vertical market through profiles (e.g., for ERP, IEC 61131); and

Compatibility: integrates previous OPC technology with a thin wrapper of OPC UA.

We will discuss the requirements of SOA-based automation applications by considering a few real-world examples. After a state of the art analysis of current OPC UA profiles for typical automation applications in IEC 61131 [5] or IEC 61499 [6] we will discuss the additions to existing profiles to enable the use of OPC UA for SOA-based automation applications.

2. Requirement analysis

In order to evaluate the suitability of and the requirements for OPC UA of SOA-based AS, we will elicitate the requirements of typical automation applications of SOA-systems. For that we will isolate the services, data, and messages required for operation. The chosen applications are the orchestration of ISA-88 based batch processing systems, distributed power generation and transmission grid control via IEC 61850-based systems, and diagnostic systems for AS.

2.1. ISA-88 based batch-process systems

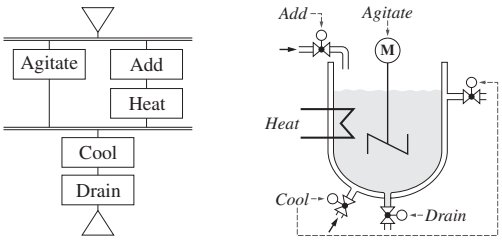
In the domain of process industries, the introduction of batch processes and modifiable recipes brought a certain grade of flexibility for producing different products in one process plant. Standardization efforts in this domain led to the specification of requirements for batch process control in the form of the technical standard ISA-88 [11], respectively IEC 61512 [7]. The concepts and definitions described in this standard are generally used as a basis for designing ASs in the batch process domain. In this context, ISA-88 provides a set of hierarchical reference models concerning processes, physical equipment as well as procedural control and describes their relations. Thereby, process functionality is achieved by mapping elements of the procedural control model onto those of the physical model. This is in principle done during the execution of recipes which define the general strategy for performing a production process.

For the production of a batch, i.e. a certain amount of a specific product, a control recipe specifies the sequence of activities in the form of procedural steps, which have to be provided by the physical components of the process plant. The phase logic interface forwards commands from the recipe phase to the referenced equipment phase to cause state transitions and to initiate the required production steps. Status messages from the equipment phase like completion are transmitted to the recipe phase.

In this context, the overall control structure of ISA-88 based production systems is a SOA-based system. A PC-based batch server is the service-user which orchestrates the production process via the services offered by the process equipment. The process equipment entities, usually controlled by industrial controllers such as Programmable Logic Controllers (PLCs), Programmable Automation Controllers (PACs), or Industrial PCs (IPCs), provide the necessary services to the batch-process system. Equipment phases are usually implemented by IEC 61131 or IEC 61499 applications or in general purpose programming languages, like C or C++.

Figure 2 shows the relationship between the recipe phases and the utilized equipment services. The services themselves are represented as equipment phases, and the service calls are represented by the recipe phases in [7]. The presented reactor is capable of adding process resources (*Add* service), heating (*Heat* service), cooling (*Cool* service), mixing (*Agitate* service), and draining (*Drain* service). Each piece of process equipment offers

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications," *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640



(a) **Recipe phases** (b) **Equipment phases**

Figure 2. ISA-88 recipe phase to equipment phase mapping.

its specific services based on its function (e.g., reactor, dehumidifiers, oxidizer) to the batch-process coordination system. The ISA-88 standard includes the exemplary services *dry*, *evaporate solvents*, *sterilize*, etc., each of them representing another service in a process plant.

Depending on the service, which can be seen as some kind of function call, zero or more service specific process parameters must be provided. For example according to [7] the *Heat* service can be invoked with only the final temperature as its parameter. If this service is used the process resources will be heated unregulated to the specified final temperature. Another possibility to invoke the *Heat* service would be to specify a heating profile with the heating rate, holding temperature, and holding time as its parameters. In [7], the ambiguity of those two different forms of heating services is solved by assigning different names to these services (e.g., *Heating* and *HeatProfile*).

Additionally ISA-88 introduces a set of operational basic states such as *Running* or *Stopping* for the equipment phases in the form of a state transition diagram (see Fig. 3). Depending on the momentary state, the equipment phase logic executes according control code for performing process-related functionality. For instance during the state *Running* the control code for the phase's normal operation is executed (e.g. achieving a specific temperature in the reactor) while for instance in the state *Stopping* code for a safe process stop is executed. So depending on the current equipment state certain services shall not be executed, even if the controlling system (e.g., SCADA, MES wants to invoke the service.

2.2. Power transmission and distribution systems

Equipment used in electrical power transmission and distribution systems (e.g., switches, bus-bars, transformers, meters) have long life-cycles, which lead to usually heterogeneous equipment in power systems.

As a result each time a new type of equipment is introduced additional interfaces have to be implemented in the control systems to attach a new component to the legacy system. The IEC 61850 tries to solve this problem, by introducing a common data and communication model. It is planned to classify all equipment components in electrical power generation, transmission and distribution, as well as to create a suitable data and communication model for all components. Security issues (e.g., operator authentication) are not directly covered by IEC 61850. Nonetheless security is a major issue in power transmission and generation systems. Security issues and concepts are taken from IEC 62351 [9].

By its ability to enable the interoperability of heterogeneous plants and plant equipment IEC 61850 is also recognized as an important building block for smart grids [13,15]. Typical tasks for smart grids are the automatic optimization of the grid with respect to fluctuations due to unpredictable loads and generation, changing market prices, consumer priority, and so on. Even more, the electrical equipment in smart grids is owned and operated by multiple market participants (e.g., energy supply companies, companies with high energy demands). Typical services for the operation of smart grids include *switching* (e.g., generator to grid), *changing set-points* (e.g., frequency for inverter, requested power for generators), *metering* (e.g., consumption/delivery to grid), *measuring* (e.g., current voltage, current, power), and *condition monitoring and failure detection*. Based on the requirements and characteristics of (smart) grids, SOA has been recognized as appropriate approach to orchestrate and operate (smart) grids [3, 19]. Minimal services, standardized interfaces and data models for the various classes are already covered in several normative documents. For distributed energy resources IEC 61850-7-420 [10] is the relevant standard document.

In the information model of IEC 61850 electrical equipment (e.g., photovoltaic (PV) cells, fuel cells, line breakers) is represented as Logical Device (LD). Logical Nodes (LNs) serve as building blocks for the LD [10]. LNs represent basic functions (e.g., measuring functions, switching functions), specialized functions (e.g., inverter, rectifier, battery charger), but also data relevant for engineering and operation (e.g., characteristics of PV modules). Fig-

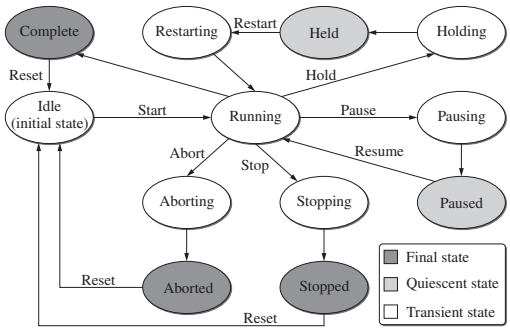


Figure 3. ISA-88 Equipment phase state machine [7, p. 31].

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640

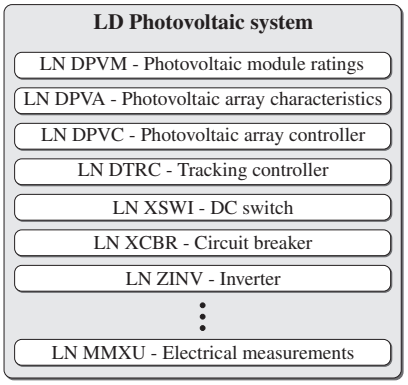


Figure 4. Logical device (LD) for photovoltaic system according to IEC 61850. Typical functions are represented as logical nodes (LN).

ure 4 provides an example for a PV system. Additional LNs may be added, if the corresponding functionality is provided by the equipment (e.g., solar plant). Each type of LN consists and encapsulates mandatory, optional, and conditional data objects. Data objects in LNs are grouped into measured values, controls, metered values, status information, settings, and those without category [10].

Even though IEC 61850 is more related to communication and data models, it also concerned with equipment services. IEC 61850-7-2 [8] specifies the common model for services, service communication and invocation, general service types (e.g., association services *Associate*, *Abort*, *Release*), and common equipment services (e.g., *GetServerDirectory*, *GetLogicalNodeDirectory*, *GetAllDataValues*).

2.3. Diagnostic systems

Safety-critical processes as in the chemical or pharmaceutical industry and requirements for ever shorter plant down-times requires process monitoring, fault detection and diagnostic systems. Industrial process monitoring and diagnostic systems can be differentiated into offline and online applications.

Offline diagnostic applications are mainly based on the evaluation of historical data sets and therefore require a service *getHistorical*, which requests historical data from a desired node set. The required recorded data can be exchanged non-cyclical and without time constraints with the automation systems.

In contrast, online model- or signal-based methods are applied for process monitoring, automatic protection or supervision with fault diagnosis [12]. In the case of online diagnosis and process monitoring, services for requesting measured values, control values and other information like state information or event logs of the automation system are required. Thus, these services should be invokable with desired node sets or create subscriptions for the node set with small cycle times in order to allow accurate and real-time process diagnosis. Beside the request of sensory or other data, the online diagnostic system should be able to suggest actions to the operational personnel or even initiate appropriate counteractions in case of automatic protection. Furthermore, the online diagnostic system should be able to trigger alarms or alerts with alarm number, priority and alarm text within the process control system.

Since not all process data are required at every cycle, or diagnosis could be applied only for sub-parts of a plant, a possible required service could be the definition of sub-plant-specific views on node subsets in the OPC UA address space. In this case, the diagnostic system transmits the desired node identifiers of the subset and receives the values or requests the subscription registration for the specific monitored items. This service should initiate the creation of a view node referencing the desired node subset, creating subscriptions or an event-notifier if desired and returning the subscription reference. However, this service is not yet available [14, p. 135].

Additionally, the diagnostic system should be able to initiate test processes. For example, a pressure test for a vessel could be a possible test case, consisting of isolating the vessel (close all valves), increasing and monitoring the pressure in the vessel.

2.4. Resulting requirements

Based on this investigation and the common requirements on industrial AS we derived the following requirements a service-oriented middleware for industrial automation has to fulfill:

- Providing access to process/equipment data or specific sets of process/equipment data;
- Integration in field and control level equipment;
- Easy integration in SCADA, MES, ERP, etc.;
- Hierarchical structuring of automation equipment;
- Reliable real-time communication capabilities suitable for industrial use;
- Immunity due to equipment faults and faulty communication;
- Able to represent automation equipment states;
- Invoke workflows and programs in automation equipment.

OPC UA fulfills all above-mentioned requirements, and has already been used successfully in several other projects [2, 3, 18, 21, 22].

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications," *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640

3. Current and future OPC UA profiles

OPC UA has been designed for providing unified communication and interaction means for SOA and is therefore well suited for implementing different views on AS. One of the views may be designed for SOA-based applications, whereas others may be data driven (e.g. for Human Machine Interface (HMI) systems) or provide an automation standard view (like one for IEC 61131). We will first provide a short overview on OPC UA and then examine the IEC 61131 profile for OPC UA developed as a joint effort of the OPC Foundation and PLCopen [17]. As there is no profile for IEC 61499 available yet, we will discuss how to implement one in the spirit of the IEC 61131 profile.

3.1. Overview

OPC UA aimed to replace the various OPC[2] standards by an open architecture. OPC was based on "standard" technology by Microsoft (OLE[3], COM[4]). In practice, OPC was bound to Windows, requiring a Windows computer connected with the AS acting as the OPC server. Similar to its predecessor OPC, OPC UA is still is a client-server protocol. Concerning portability, the major difference is that the server is now using a portable communication stack which can be (and often is) implemented directly on the AS.

The functionality of the previous OPC standards, like OPC Data Access (OPC DA), OPC Historical Data Access (OPC HDA), OPC Alarms & Events (OPC A&E) is aggregated into a single standard, which is the new OPC Unified Architecture. To protect legacy technology, there an existing OPC server can be used in the new OPC UA technology via a wrapper.

The contents of an OPC UA server are always model-based. It supports the concept of data models which can be defined to match an application's needs. A data model can be published as a so-called profile, providing a standardized interface to clients.

Typically a profile is specifically designed for vertical integration (e.g., MES, IEC 61131). Profiles extend the means of communication OPC UA defines (how to connect and talk to the server) by providing information on how the data is organized and can be accessed (see below for a short discussion on the IEC 61131 profile).

In addition to models and profiles, OPC UA adds the concept of state machines (e.g., for recipes as specified in ISA-88) and programs. Programs are (named) functionality which can be run by invoking the program through OPC UA. This is a Remote Procedure Call (RPC) mechanism or to be more precise, a mechanism for object remoting as provided by systems like CORBA, or COM and this is the foundation for SOA based applications.

[2]OLE for Process Control (OPC) was the initial name of the industrial standard, today OPC is no abbreviation anymore.
[3]Object Linking and Embedding (OLE)
[4]Component Object Model (COM)

3.2. IEC 61131

The IEC 61131 profile for OPC UA has been developed in a joint venture of the OPC Foundation and PLCopen. Experts from industry and academia have designed a model (and profile) to map the IEC 61131 application model to OPC UA. Various implementations of this model have been presented on major trade fairs since 2010. Obviously the main advantage of this profile as shown in the applications is that the HMI (SCADA) application can be developed independently from the PLC application.

The final goals of the IEC 61131 profile are:

- providing a portable data model PLC vendors implement and HMI vendors can rely on,
- providing means for standardized PLC-to-PLC communication, extending the rather basic means defined in IEC 61131-5, and
- exposing IEC 61131 functionality to OPC UA as programs (not necessarily IEC 61131-3 programs).

At the time of writing of this article the first goal already has been achieved and has been implemented by PLC and SCADA vendors. The second goal, PLC-to-PLC communication is currently an in-work item. The third goal have been postponed as focus is set on the OPC (UA) "killer application" data access (formerly OPC DA).

Although the IEC 61131 profile for OPC UA is widely accepted by the industry, it does not cover all requirements identified in this investigation. A service oriented middleware needs the ability to invoke services in the PLCs. While OPC UA considers such features they have currently not been used with IEC 61131-3.

Summarizing the IEC 61131 profile for OPC UA is well suited for HMI interaction but currently the profile does not expose the functionality required as an infrastructure for a SOA based application.

3.3. IEC 61499

IEC 61499 has been defined as reference model for adaptive reconfigurable control systems. In order to support these properties two major changes compared to previous PLC programming methods have been introduced: First a direct support for distributed control systems allowing to localize changes and secondly an event driven execution paradigm giving the control engineer more control over execution orders and application part synchronization. These are also key properties for our targeted service oriented middleware. However, currently we are not aware of any investigations utilizing OPC UA together with IEC 61499. For this a profile similar to the IEC 61131 profile for OPC UA would be needed, especially for unified HMI interaction. Such a profile could even represent the same features as the IEC 61131 profile for OPC UA and therefore allow the same interaction mechanisms independent of the target system (e.g., IEC 61131 or IEC 61499). However this is not the main scope of this investigation, we are more interested in the

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640

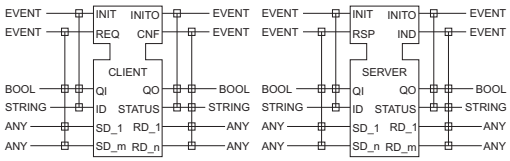


Figure 5. Generic client/server service interface function blocks for bidirectional data transfer according to IEC 61499-1 Annex E.

capabilities of representing and implementing services in IEC 61499.

Elements, which are not defined within the standard IEC 61499, are integrated by the Service-Interface Function Block (SIFB) concept. With SIFBs external functionality like I/O access or communication can be used within IEC 61499 applications. However, only the interface of these Function Blocks (FBs) can be described with the means of IEC 61499 as the internals and its implementation are device specific. Apart from the generic definition, IEC 61499 defines two general kinds of execution behaviors for SIFB: the application triggered and the resource triggered behavior. The first means that the IEC 61499 is in charge for activating the service which is encapsulated by the SIFB, while for the latter the resource (e.g., communication system) will be the activating element triggering execution within the IEC 61499 application. The resource triggered execution behavior is exactly the behavior required for the remote service invocation of the targeted middleware as it allows remote systems to remotely trigger execution within an IEC 61499 application.

IEC 61499 has been designed as model for distributed control systems. Therefore it also considers communication in its models. In Annex E of IEC 61499-1 two generic communication models are presented utilizing the SIFB concept. The first is the unidirectional interaction model with the Publisher and the Subscriber FB and the second one is the bidirectional interaction model with the Client FB and the Server FB (cf. Fig. 5). In our investigation the first model is of interest when the higher levels just like to trigger the control device without the need of a feedback. However most of the time at least an acknowledgment informing on the status of the request is required.

As services involve bidirectional data exchange and services are offered by the device to the network the second model utilizing the server FB is the best fit for representing services from IEC 61499 to OPC UA. This allows to explicitly model services provided by the control application through adding them to the control application. FB parameters can be used to specify the service name and any required properties. The data in- and outputs used in the server block define the data used for the service. Hereby the data outputs of the FB are the data provided by the higher level to be used as input for the service execution and the data inputs of the FB are the results of the service execution. For interaction with OPC UA it is of interest how this data is represented in OPC UA. IEC 61499 has adopted the data type definition from IEC 61131-3 and as there is already a definition for the OPC UA representation of IEC 61131-3 data types in the IEC 61131 OPC UA profile we will utilize the same definitions for our service data representation. The `IND` output event of the resource triggered FB is the activation trigger for the requested service and the `RSP` input event of the FB is the notification back to the invoker on the finishing of the service execution.

Compared to the possibilities of IEC 61131 we identified several advantages for implementing services with IEC 61499. At first, this approach directly represents services within the control application. This makes them better understandable and directly visible for the control engineer. Also OPC UA can directly activate the services without the need of setting and resetting any guarding flags. Finally the event triggered execution of IEC 61499 allows an easy synchronization of service activation, execution, and finishing.

4. IEC 61499 SOA example

This section will provide a short example of SOA-based process equipment programming in IEC 61499, based on the process reactor in Sec. 2.1 (see Fig. 2). The reactor provides four services to the AS: *Add*, *Heat*, *Cool*, *Agitate*, and *Drain*. As proposed in Sec. 3.3 each service will be represented as an SIFB.

Figure 6 shows the SIFB for the *HeatProfile*, as described in Sec. 2.1. Behold that the outputs on the *HeatProfile*-SIFB are the needed parameters for the service. They will be fed into the inputs of the hardware-specific implementation of the service (see Fig. 8). If the service is invoked via OPC UA the `REQ` event output will fire an event, thereby triggering the service implementation. The data-input `Service_Result` and the corresponding event-input `CNF` are used to provide feedback to the service-user.

Considering Fig. 8 it can be noticed that the service-representing SIFBs are only connected to their corresponding hardware-specific implementation FBs. As we apply the SOA-based approach we separate the hardware-specific parts of the automation application from the application's workflow.

Last but not least we also compare the hierarchical structure of the IEC 61499 application to the OPC UA address space model. Lets consider that the process reac-

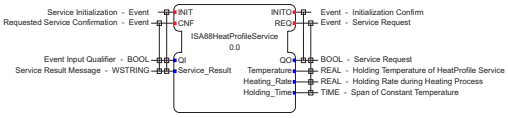
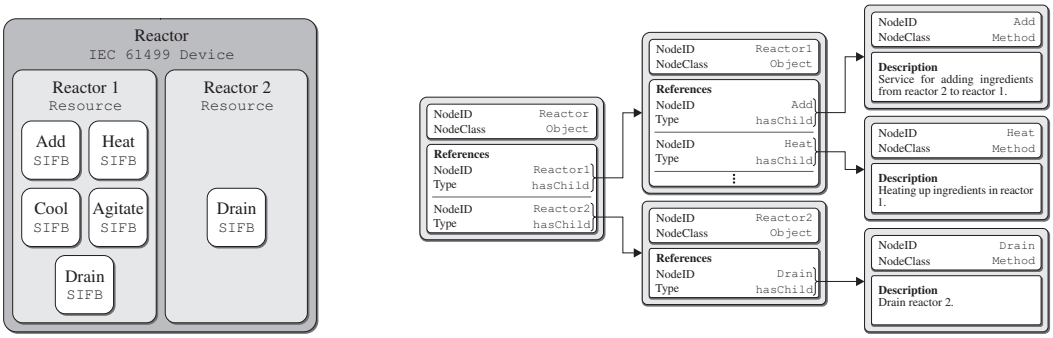


Figure 6. Definition of the ISA-88 HeatProfile service as IEC 61499 function block.

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640



(a) **IEC 61499 application model** (b) **OPC UA address space**

Figure 7. Comparison between IEC 61499 application model and OPC UA address space model.

tor as shown in Sec. 2.1 is part of a bigger multi-process-reactor unit, which consists of two equal process reactors but is controlled by the same hardware (e.g., PLC, PAC, IPC). In IEC 61499 we could model this by an IEC 61499 device, representing the multi-process-reactor unit, and two process reactor resources, each representing one of the reactors (see Fig. 7). In each of the resources (respectively each of the reactors) several services are offered and implemented as shown before.

Based on the IEC 61499 application model, we can automatically derive an OPC UA address space model. The IEC 61499 device becomes the root-node of our OPC UA-(sub-)-tree. Both of the two reactors (represented by the IEC 61499 resources) are connected to the root-node, showing that they are sub-parts of the multi-process-reactor unit. The services (SIFBs in the IEC 61499 resources) are also represented by OPC UA address space nodes, connected to the appropriate process equipment.

As we can see IEC 61499 is suitable for SOA-based automation application programming. Furthermore the IEC 61499 application model can be easily transformed into an OPC UA address space model.

5. Conclusion

SOA-based control is a rapidly developing field, spreading in all application fields of ASs. Due to their inherent property of separating the hardware-specific implementation from the logical workflow, SOA-based ASs are seen as the next step in AS application programming.

Figure 8. ISA-88 like reactor resource implementation in IEC 61499.

Several properties have to be taken into account for the correct selection of a SOA-based middleware, especially in industrial automation systems. Besides the functional requirements, for example the ability for service invocation, or data acquisition capabilities, non-functional economic requirements have to be taken into account. That means beside all the mandatory technical functions a SOA-based middleware has to provide, a SOA-based middleware for industrial automation has to be low-cost and highly available.

Based on the success of Classic OPC in the near future all automation control equipment (e.g., PLC, PAC, IPC) will be delivered with OPC UA already part of its standard software, regardless if it is controlled by an IEC 61131 or IEC 61499 system, or programmed by a general purpose programming language. Apart from the future high availability of OPC UA systems, OPC UA also covers all necessary functional requirements for SOA-based control. Another benefit of the high industrial acceptance of OPC UA is that several industrial user-groups will (and already have) develop OPC UA-profiles for several application fields and AS programming languages. This will further increase the acceptance of OPC UA, while further reducing the costs of OPC UA-based systems.

We also showed, based on an ISA-88-batch processing application, the suitability of IEC 61499 for SOA-based application engineering and how to translate the IEC 61499 application model automatically to an OPC UA address space model.

Finally based on our findings, it can be concluded that OPC UA is suited for implementing a SOA-based middleware in industrial control.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) project Holonic Integration of Cognition, Communication and Control for a Wood Patching Robot (Hol-I-Wood PR) under grant agreement No.

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640

284573.

This work has been supported by the Austrian Research Promotion Agency (FFG) project MASGrid (832037) under the Research Studios Austria program.

This work has been supported by the Austrian Research Promotion Agency (FFG) project PrOnto (829576) under the BRIDGE program.

References

[1] H. Bohn, A. Bobek, and F. Golatowski. SIRENA – Service infrastructure for real-time embedded networked devices: A service oriented framework for different domains. In *Proc. of the Int. Conf. on Networking, Int. Conf. on Systems and Int. Conf. on Mobile Communications and Learning Technologies*, pages 43–48. IEEE Computer Society, 2006.

[2] G. Cândido, F. Jammes, J. B. de Oliveira, and A. W. Colombo. SOA at device level in the industrial domain: Assessment of OPC UA and DPWS specifications. In *Proc. of the 8th IEEE Int. Conf. on Industrial Informatics*, pages 598–603, 2010.

[3] A. Claassen, S. Rohjans, and S. Lehnhoff. Application of the OPC UA for the smart grid. In *Proc. of the 2nd IEEE PES Int. Conf. and Exhibition on Innovative Smart Grid Technologies*, pages 1–8, 2011.

[4] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

[5] Int. Electrotechnical Commision. *IEC 61131 – Programmable controllers, Part 3: Programming Languages*, January 2003.

[6] Int. Electrotechnical Commision. *IEC 61499 – Function blocks, Part 1: Architecture*, January 2005.

[7] Int. Electrotechnical Commission. *IEC 61512 – Batch Control, Part 1: Models and Terminology*, 1997.

[8] Int. Electrotechnical Commission. *Technical Report IEC/TR 61850 – Communication networks and systems in substations – Part 7-2: Basic information and communication structure - Abstract communication service interface (ACSI)*, 2003.

[9] Int. Electrotechnical Commission. *Technical Specification 62351 – Power systems management and associated information exchange - Data and communications security, Part 1: Communication network and system security - Introduction to security issues*, 2007.

[10] Int. Electrotechnical Commission. *Technical Report IEC/TR 61850 – Communication networks and systems for power utility automation – Part 7-420: Basic communication structure – Distributed energy resources logical nodes*, 2009.

[11] Int. Society of Automation. *ISA-88 – Batch Control*, 1995.

[12] R. Isermann. Supervision, fault-detection and fault-diagnosis methods – An introduction. *Control Engineering Practice*, 5(5):639–652, 1997.

[13] M. Janssen, P. Crossley, and L. Yang. Bringing IEC 61850 and smart grid together. In *Proc. of the 2nd IEEE PES Int. Conf. and Exhibition on Innovative Smart Grid Technologies*, pages 1–5, 2011.

[14] W. Mahnke, S.-H. Leitner, and M. Damm. *OPC Unified Architecture*. Springer, 1st edition, 2009.

[15] S. Mohagheghi, J.-C. Tournier, J. Stoupis, L. Guise, T. Coste, C. A. Andersen, and J. Dall. Applications of IEC 61850 in distribution automation. In *IEEE PES Power Systems Conf. and Exposition*, pages 1–9, 2011.

[16] S. Y. Nof. *Springer Handbook of Automation*. Springer Publishing Company, Incorporated, 1st edition, 2009.

[17] PLCopen and OPC Foundation. *OPC UA Information Model for IEC 61131-3*, March 2010.

[18] S. Rohjans, M. Uslar, and H.-J. Appelrath. OPC UA and CIM: Semantics for the smart grid. In *Proc. of the IEEE PES Transmission and Distribution Conf. and Exposition*, pages 1–8, 2010.

[19] S. Sučić, A. Martinić, and D. Francesconi. Utilizing SOA-ready devices for virtual power plant control in semantic-enabled smart grid analyzing IEC 61850 and OPC UA integration methodology. In *Proc. of the 2nd IEEE Int. Conf. on Smart Grid Communications*, pages 43–48, 2011.

[20] C. Szyperski. *Component Software Beyond Object-Oriented Programming*. Addison-Wesley, 2nd edition, 2002.

[21] D. van der Linden, H. Mannaert, W. Kastner, V. Vanderputten, H. Peremans, and J. Verelst. An OPC UA interface for an evolvable ISA88 control module. In *Proc. of the 16th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, pages 1–9, 2011.

[22] J. Virta, I. Seilonen, A. Tuomi, and K. Koskinen. SOA-based integration for batch process management with OPC UA and ISA-88/95. In *Proc. of the 15th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, pages 1–8, 2010.

Pre-Print version of the article: Martin Melik-Merkumians et al., "Towards OPC UA as portable SOA middleware between control software and external added value applications,"*Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, 2012. DOI: 10.1109/etfa.2012.6489640