Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-2 (2020) 8268–8275

# Orchestration vs. Choreography Functional Association for Future Automation Systems

Andreas Stutz*; Alexander Fay**; Mike Barth***; Mathias Maurmaier****

*Siemens AG, Digital Industries – Technology and Innovations, Karlsruhe, Germany (andreas.stutz@siemens.com).
**Helmut-Schmidt-University Hamburg / University of the German Federal Forces, Germany (alexander.fay@hsu-hh.de)
*** University of applied Science Pforzheim, Germany, (mike.barth@hs-pforzheim.de)
****Siemens AG, Digital Industries – Technology and Innovations, Karlsruhe, Germany (mathias.maurmaier@siemens.com).

Abstract: Production flexibility, engineering efficiency and faster time-to-market are customer needs in order to survive on the market. Highly flexible system architectures are the key to fulfill these needs. Today's procedural flexibility, like in the IEC 61512 standard, is not sufficient. Extending flexibility is a must, whereas central orchestration systems reach their limits. Besides orchestration there is a second association method, the choreography. The focus of this contribution is the question of how choreography can be applied in the automation context. Starting point of this contribution is a terminology foundation in the domain of micro-service automation systems followed by a characterization of orchestration and choreography in the automation context. In this paper, we show that both methods complement each other ideally. Central orchestration is used to coordinate decentralized choreographies – complexity reduction combined with high flexibility.

*Keywords:* service-oriented automation architecture, decentralized automation, service orchestration, service choreography, service composition, service aggregation, service association

## 1. INTRODUCTION

Production flexibility and engineering efficiency enables faster time-to-market. One promising approach to reach these goals in industrial automation is to compose production plants from intelligent process equipment assemblies with decentralized, pre-engineered functionality. Also, the Platform Industry 4.0 Research Advisory Board highlighted the demand for highly flexible system architectures (acatech 2019). In this report, methods to increase the degree of flexibility within system architectures have been explicitly mentioned as research gap.

Today's production systems are based on a central orchestration, as standardized in (IEC61512 1997). It only provides procedural flexibility, which is not sufficient for highly flexible productions. Extending the procedural flexibility also into the direction of open- and closed-loop control leads to a so-called micro-service architecture. In micro-service architectures, the association by choreography is favored over the orchestration, primarily due to the lack of a centralized middleware layer. While orchestration uses a central coordination, choreographies are designed based on the collaboration principle. In industrial automation, concepts to apply service choreographies are still missing. The long-term goal of this work is to do research and development for service choreography within industrial automation systems. The results of the underlying study of this paper are divided into two publications due to their scope. The present paper represents a more comprehensive but rather theoretical consideration of service associations in the context of industrial automation. The second paper (Stutz et al. 2020) complements this paper with a more practical approach to service association with a choreographic approach.

## 2. MOTIVATION

To develop a solution for automation choreographies, first a base terminology and characteristics of both methods is explained and correlated to the automation domain. In the analyzed literature, the terms aggregation, composition, orchestration, choreography, are often not clearly defined and used as synonyms. Regarding to this, this paper proposes a terminology foundation in the domain of service-oriented automation systems and characterization of orchestration and choreography in the automation context. For that, it starts, with an introduction to the methodology in section 3. In section 4, the clarification of the service term is given, which will be further used in this work. The classification of functional associations will be discussed in Section 5. Section 6 point out the characteristics of orchestration as well as choreography and its comparative analysis. The learnings are conclusively described within Section 7. Section 8 transfers the learnings into the context of automation with several simplified examples. Finally, in section 9 a short summary is given as well as the next steps are named.

## 3. METHODOLOGY

Service-oriented automation architectures provide a means to enhance the flexibility of automation systems. Until today, multiple research projects worked on the applicability of service-oriented principles (see table 1).

A total of 50 publications have been examined from 8 publicly funded research projects, as well as from privately funded works. About 75% of the contributions were in the

10.1016/j.ifacol.2020.12.1961

field of orchestration and about 25% in the field of choreography. While the majority of the literature originates from the field of automation, the basic literature for orchestration and choreography comes from the web service domain. In the following, six characteristics of orchestration and choreography are worked out and compared. First, both association methods are examined independently of each other, based on the literature. In a second step, these characteristics are categorized and compared. Finally, the findings from the comparison are interpreted and their significance for automation and control technology is derived. In the two sub-chapters in which the association methods are described, the literature used for this purpose is individually referenced.

Table 1: List of research projects

| **Projects** | **Publications of this Project** |
|---|---|
| SIRENA | ITEA 2005; Jammes et.al. 2005 |
| SOCRADES | SOCRADES 2009; Cannata et.al. 2008; Colombo et.al. 2010; Mendes et.al. 2008, 2012 |
| IMC-AE-SOP | IMC-AESOP 2013; Colombo et.al. 2012, 2014 |
| SkillPro | SkillPro 2019, Pfrommer et.al. 2014, Pfrommer et.al. 2019 |
| ProSEco | ProSEco 2013, Brito et.al. 2017, ProSEcoD1004 2014 |
| BaSys4.0 | BaSys40 2019, Malakuti et.al. 2018, Terzimehic et.al. 2017 |
| ENPRO2-ORCA | ORCA 2019, VDI2658 2019 |
| DEVEKOS | DEVEKOS 2019 |

## 4. SERVICE TERMINOLOGY

An accepted definition of web service is provided by (Sheng et.al. 2014, Richards 2015). Already four years earlier (Mendes et al. 2008, 2012) introduced a similar definition in the web-service based automation community. The community of service-oriented manufacturing systems, (Dorofeev et al. 2018, 2019, Pfrommer et al. 2019, FA721 2019) interpret a service as an executable skill. The representatives of the service-oriented process control community, (VDI2658 2019, Bloch et al. 2018) interpret services as an executable process function of a process equipment assembly. They all have in common the need to provide a predefined behavior and the fact that this functionality is accessible via standardized interfaces. Based on these interpretations the following definition of a service is proposed:

*(A) A service provides dedicated functionality in the form of procedures which results in a real-world effect in the underlying production process by implementation of the input-process-output pattern.*

The authors of (Sheng et.al 2014, Mendes et al. 2008, 2012, Dorofeev et al. 2019a) distinguish services into *atomic* and *composite*. *Atomic services*, also called *elementary services*, do not relay on another service to fulfil their functionality. The *composite service* is a combination of other composites and/or atomics. As mentioned in (FA721 2019) a skill represents an entity with the potential to achieve a measurable effect within the physical world.

The German industry standard (VDI2658 2019) defines interfaces for accessing process functions implemented in the process equipment assemblies. Sensors, actors and logic are hidden behind the interface.

The understanding of skills in the manufacturing domain (Dorofeev et al. 2019b) focuses on the executable implementation, whereas the (VDI2658 2019) focuses on the capsulation realized by standardized interfaces in the process industry. Both aim to achieve a real-world effect. From a functional perspective a skills and process functions are the same.

A small exemplary application of how "a measurable effect […] in the real world", can be understood from an automation perspective (FA721 2019) is shown in Figure 1. It shows a schematic closed-loop control, that can be provided as a service. A closed-loop control is characterized by a logic which controls the actor that influences the process. A sensor measures the process values and returns it to the logic. From a functional point of view, a service is able to realize a measurable real-world effect if this service contains an acting and sensing element as well as a logic in between. The described relationship is called "input-process-output-pattern".

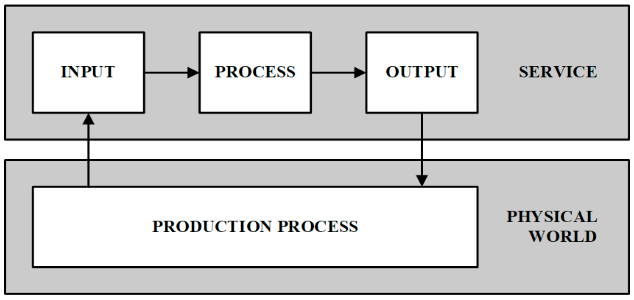

Figure 1: Schematic model of the input-process-output-pattern for describing the measurable real-world effect in automation services.

If a service encapsulates all the building blocks of the pattern, the measurable real-world effect can be achieved without any other associated service. This fact results in the following proposed definition regarding the independence of a service to other services regarding the encapsulated functionality:

*(B) Independence is given, if the input-process-output pattern is fully realized by a given service entity.*

Based on definition (A) and the independence criteria for services (B) the following term is proposed and used in this paper:

*(C) An elementary service is a non-associated service, which can be used independently and thus has a measurable real-world effect on its own.*

Within the cooperation between Siemens and Merck a new kind of service was developed – an analytical service. This service only provides a measurement functionality. Considering this idea, where a functionality is introduced which does not achieve a real-world effect related to the mentioned explanations, the definitions from (A) and (B) need to be extended in the following way:

(D) *A sub-elementary service is a non-associated service, which cannot be used independently and thus has not a measurable real-world effect on its own.*

With these definitions (A-D), the terms *composition* and *aggregation* are specified in a consistent way.

## 5. SERVICE ASSOCIATIONS

This section provides a classification of *composition* and *aggregation* of automation services motivated by the understanding from the domain of software engineering. (Sheng et al 2014) introduced the term of a *composite service*. (Colombo et al. 2010) use *composing*, *aggregating* and *orchestrating* in an equal manner. The *composition* and *aggregation* abilities are named in (Colombo et al. 2014) as one of the biggest achievements of this research. Throughout these papers, the terms composition and aggregation are not used in a consistent manner. The baseline for the interpretation of these terms is given by the object-oriented programming paradigm. In (Rumbaugh et al. 1999) compositions and aggregations are introduced as specialized associations between two or more software components. The difference lies in the degree of interdependence between the associated components. An aggregation expresses that the component TWO, which is related to component ONE, can exist independently. The composition expresses that the related component TWO depends on the existence of component ONE. This understanding is now applied to the domain of service-oriented automation systems. The service composition expresses that the related service TWO cannot be used independently without the composing service ONE.
The idea of service associations is to combine existing functionalities to a more sophisticated one. Using the interpretation of aggregation and composition as named by (Rumbaugh et al. 1999) as well as the definitions 4.A-D the terminologies service association, service aggregation and service composition can be differentiated. A service, which is following the definition of 4.D does not provide an independent functionality and must be associated with another service, in a way, that the independence criteria (4.B) is fulfilled. With regard to independence, this kind of association fulfills the characteristic of a composition based on the fact that the sub-elementary service cannot be executed independently without the associated service. Following this understanding, the following definitions are proposed:

(A) *A service association is given if the associated services yield a new functionality, which can be further used in an independent manner.*

(B) *A service composition is given, if the association follows the composition principle and combines services, whereby at minimum one of these services is a sub-elementary service.*

(C) *A service aggregation is given, if the association follows the aggregation principle and combines elementary services or other associated services.*

These associations of services can be realized based on an orchestration or a choreography method. Both approaches will be characterized in the next sections.

## 6. ANALYSIS OF THE ASSOCIATION METHODS

### 6.1 EVALUATION CATEGORIES

The evaluation of the characteristics and the later comparison are separated into the following six categories.

**Accessibility** focus on the way how a service functionality can be used by other services or a higher layer system. In the category **operation** it is analyzed how the different aspects of the function to be realized are designed. **Deployment** characterizes the ability to be executed in a central or decentral manner. In the category **hierarchy**, the ability to encapsulate other functions is evaluated. **Interaction** evaluates the communication patterns to be used. The last category is the ability to hide or reduce **complexity**.

### 6.2 ORCHESTRATION METHOD

**Accessibility**

(Sheng et al 2014, Richards 2015) define service orchestration as a single coordination task under control of a single entity. (Spinelli et al. 2018) describe orchestration as the coordination of systems consisting of hierarchically complex systems. (Lau et al. 2015) use the term composition by coordination, which is a synonym for orchestration. Based on these findings, the following characteristic of orchestration is pointed out:

(A) *Orchestration defines an executable functionality from the perspective and under the control of a single accessible orchestrating entity.*

**Operation**

(Sheng et al 2014) define the control flow as well as the transaction management and exception handling as part of a web service. Following the explanations of (FA721 2019, IEC61512 1997), an automated function contains a coordination control, procedural control and a basic control depending on the functionality. This results in the next characteristic:

(B) *In applications, there are several functional layers, which have to be realized not only the pure sequential execution, but also other necessary system functions.*

**Deployment**

(Pfrommer et al. 2019) present an MES-based orchestration approach, where the MES handles different orders and executes them by orchestration of fine-grained services. (Klose et al. 2019) specify the requirement to provide a centrally located system to define sequences and how to execute them. (ProSEcoD1004 2014) uses common web-service orchestration engines. The mentioned contributions represent central orchestration mechanisms. Based on these findings the following characteristic is pointed out:

(C) *Orchestration can be realized in a centralized manner by placing the orchestrating entity in and execute them by the overlaying control system.*

(Jammes et al. 2005) present a composed service which uses an orchestration method to associate sub-elementary services from sub-ordinate controllers in a third controller. (Mendes et al. 2008, 2012) use a high-level controller for orchestration, which interprets petri-nets. The listed contributions unify the commonality of a decentralized deployed execution of the orchestration. The demonstrated examples point out, that this kind of orchestration requires a special pre-design of the method. Based on these findings, the following characteristic can be formulated:

*(D) Orchestration can be realized in a decentralized manner, by placing the orchestrating entity in and execute them by the module-own integrated controller.*

**Interaction**
(Sheng et.al 2014) state that the interaction between the orchestrating entity and the orchestrated services must be realized in a request-reply pattern to ensure the coordination of underlying services. Thus, the following characterization concerning interaction is derived:

*(E) For the orchestration of services, a request-reply interaction is required.*

**Hierarchy**
(Terzimehic et al. 2017) demonstrated an orchestrated function in IEC61499 which can be made accessible via OPC UA for an external orchestration. Following this contribution and the two proposed definitions (6.C and 6.D), it is possible to provide hierarchical associated services. If the highest level of service association is achieved, no further associated service can be provided. This fact results in an additional definition proposed in this paper:

*(F) An orchestration entity can be made accessible as a new independent service, except on the highest level of orchestration.*

**Complexity**
(Dorofeev et al. 2019b) present skill-based control architecture approaches consisting of multiple hierarchical layers realized by orchestration. According to the authors, the majority of field entities should be hidden to reduce complexity. This results in the last proposed characteristic of orchestration:

*(G) Reduction of complexity is realized by hiding the underlying services behind the orchestrating service.*

## 6.3 CHOREOGRAPHY METHOD

**Accessibility**
(Sheng et al. 2014) introduce choreography as a description of observable behavior, by defining the interaction between each of the involved services. In (Richards 2015) choreographies are described as the interaction between defined processes. (Lau et al. 2015) introduce composition by interaction as collaborative method. Based on these contributions, the following definition is proposed:

*(A) A choreography describes the observable behavior determined by the interaction of services accessible via a single or two different endpoints.*

**Operation**
(Seeger et al. 2018, 2019) introduce a recipe concept with offerings and signal connections of and between services. Based on this contribution and (Sheng et al 2014, Peltz 2003), choreographies require interactions on different levels. This leads to the following characteristic:

*(B) The behavioral description of a choreography has to cover not only the pure collaborative functionality, but also other relevant system behaviors.*

**Deployment**
(Mendes et al. 2008) demonstrate a choreography of two smart mechatronic components by exchange of boolean signals between the controllers. In (Chen et al. 2017), the authors built a smart city application based on several services that are associated via choreography in a central runtime environment. These publications point out that a choreography can be deployed in a decentralized or centralized manner. This results in the following delimitations:

*(C) A choreography can be executed centrally, within an uniform runtime environment.*

*(D) A choreography can be executed in a decentralized manner, distributed over several, potentially heterogenous runtime environments.*

**Interaction**
In the domain of web-services (Peltz 2003) describes that choreography is defined as the message exchange between the different workflows. (Sheng et al. 2014) name this interaction mechanism “send/receive pattern”. Within (Mendes et al. 2008) messages are sent as triggers between petri nets. (Seeger et al. 2018, 2019) apply a message-based mechanism to communicate variables between the different logic nodes. (Starke et al. 2013) use event-based interaction, too. Besides the send/receive pattern, (Cordes et al. 2020) introduced a monitor/act interaction pattern. All of these contributions lead to the following property:

*(E) A choreographed service interacts via send-receive or monitor-act interactions.*

**Hierarchy**
According to (Peltz 2003, Sheng et al. 2014) a choreography is a fully transparent association method, which results in the following characteristic:

*(F) Choreographies result in a transparent, non-hierarchical functional association.*

**Complexity**
Because of missing hierarchy abilities, choreographies become more complex the higher the number of involved services and required interactions is. These learnings result in the following proposition:

*(G) Choreographies realize a transparent association, whereby the complexity of interaction rules increases with the number of relevant services and interactions.*

## 6.4. COMPARATIVE ANALYSIS

In chapter 6.2 and 6.3 the two association methods, orchestration and choreography, have been characterized. In this chapter, the different characteristic findings are compared.

**Accessibility** (6.2.A, 6.3.A) - An orchestrated service is accessible via a single entity, whereas a choreography can have different start and end entities. An entity provides the communication-related endpoint to access the associated functionality. If choreography is designed with one single start and end entity, there is no accessibility difference between both methods.

**Operation** (6.2.B, 6.3.B) - Considering (IEC61512 1997), the realizing logic can be differentiated into coordination control, procedural control, regulatory control for open- and closed loops, exception handling as well as interlocking. Both association methods allow the execution of logic over all levels to fulfill the associated service function. In this category there is no difference between both methods.

**Deployment** (6.2.C/D, 6.3.C/D) - The deployment can be realized in a central or decentral manner. A central orchestration system is more flexible compared to the decentralized approach, because it can be easily changed and redeployed (Peltz, 2013). A decentralized orchestration must be pre-engineered within the modular unit, because the software of the unit shall not be changed after testing. The centrally executed choreography results in longer delay times due to its used delegates (Chen et al. 2017). The aspect of longer delay times compared to orchestration is also valid for all kind of orchestrations. Decentralized choreography, however, provides the possibility to realize shorter delay times due to its direct interaction.

**Interaction** (6.2.E, 6.3.E) – In an orchestration, the interaction works in a request-reply behavior from the perspective of the orchestrating entity. This results in minimum two interactions. For choreographies, there are two different interaction variants. These variants are differentiated by the service, which triggers the next action – the previous service or the service itself. The previous one follows an external control and requires two interactions, monitoring and activation of the next service. The self-activated service activates itself depending on other associated services, by monitoring them. This results in a self-activated interaction pattern which is preferably used for choreographies.

**Hierarchy** (6.2.F, 6.3.F) – Hierarchical structures are only established in the orchestration method.

**Complexity** (6.2.G, 6.3.G) – Orchestration provides the ability to hide the complexity of underlaying services, whereas in choreography all involved services are transparently shown. With increasing complexity of associated functionalities, the engineering complexity for choreographies increases over-proportionally. In an orchestrated system, this complexity is encapsulated and hidden.

## 7. CONCLUSION

This paper provides a detailed differentiation of the term ‘service’ (4.A-C). With elementary services, sub-elementary services and associated services, an extended definition of a service and its constructs has been proposed. The analysis of service associations emphasized the differentiation of composition and aggregation (5.A-C).

Choreography is addressed in a minority of scientific projects. In the domain of service-oriented automation systems, 75% of the analysed papers use orchestration for service associations. This is most likely due to the fact, that the realization of orchestration is more familiar to existing development tools within the automation domain. There is no method described in literature, that can be used to design automation choreographies.

The characteristics of orchestration and choreography (6.2.A-G, 6.3.A-G) and their comparison (6.4) pointed out that there exists no best-practice method for realizing highly flexible automation systems. Choreographies evolve their strengths within smaller and less-complex associations as well as in a decentralized approach to achieve shorter delay times. Due to missing hierarchy and hiding abilities, choreography is not advised for larger associations, whereas orchestration should be preferably used only in procedural associations, due to longer delay times.

## 8. FUNCTIONAL ASSOCIATION IN AUTOMATION

Based on 4.A-D, an automated functionality comprising a sensor, an actor and logic, which affects the process can be interpreted as an elementary service from the automation perspective. A sub-elementary service contains only parts of the input-process-output pattern.

Following 5.A-C, a simple media transfer, as the combination of a closed-loop flow control service and an inlet valve service, is a service composition, because the inlet valve service does not fulfill the independence-criteria due to the missing flow sensor. A mixing function associated of a stirring service and a tempering service, results in an aggregation, because stirring and tempering service are designed independently.

An orchestration within the automation can be visualized like the service association in figure 2/3. Considering the example of the mixing services, as a combination of stirring and tempering, the mixing service provides a new interface to be accessible from the overlaying system (6.2.A/F). The implementation of the mixing service realizes the hiding mechanism (6.2.G) and the executable logic, for procedural control, regulatory control and interlocking logic (6.2.B). The interaction is directed from the coordinator to the services (6.2.E). The mixing service can now be executed as a decentral implementation (6.2.D) or in the orchestration layer (6.2.C).

The method of choreography can be demonstrated with a media transfer service (see figure 4/5). The shown choreography will be accessible at one defined service (6.3.A). Each of the services gets a behavioral configuration (6.3.B), to fulfill procedural control, regulatory control and interlock logic. They contain information about their own monitor-act interactions (6.3.E) that must be performed to realize this association. Choreography-enabled services can be configured as decentral systems (6.3.D), non-choreography-enabled services can be enabled by connecting them to choreography proxies, which leads to a centrally located choreography (6.3.C). Independent of the central or decentral approach, all the services are equal and transparently visible (6.3.F/G).

Based on these considerations and it's transfer into the automation domain choreography should be preferably used to form decentralized service associations. These decentralized associations are used to close open endpoints for regulatory control (open- and closed control loops) and interlocking via direct module-to-module communication for shorter delay times. The choreographed associations can then be further used in a procedural only manner, similar to the standardized orchestrations systems can handle today. This combination of central orchestration and decentral choreography enhance the flexibility of the system and reuse existing standards of well-known production systems.

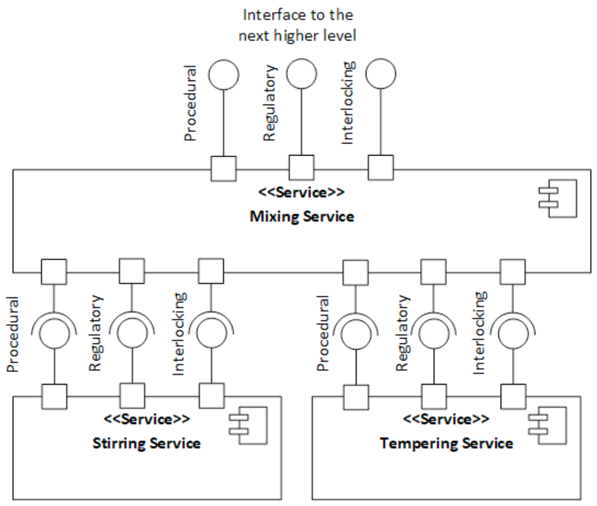

*Figure 2: Schematic component model of an orchestrated automation service*

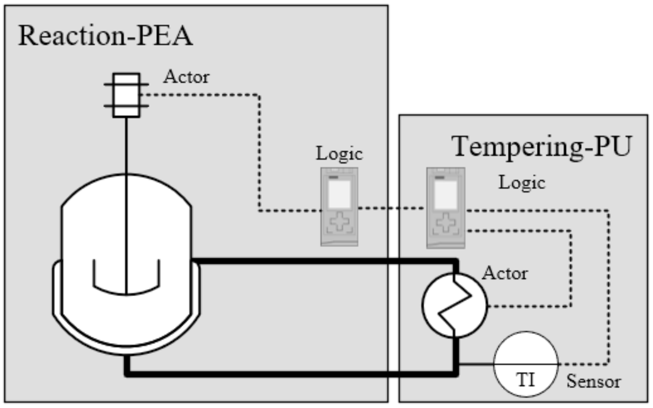

Figure 3: Technological Schema of the mixing example

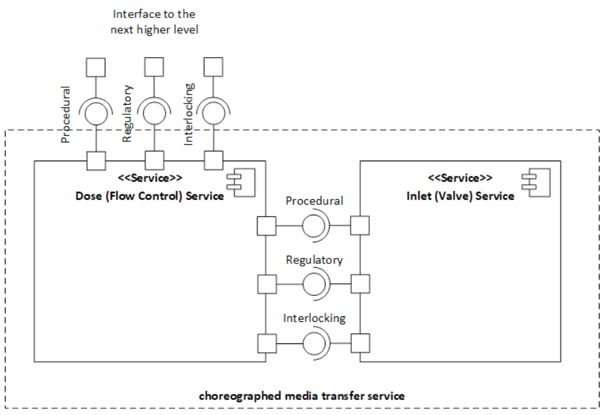

Figure 4: Schematic component model of a choreographed automation service

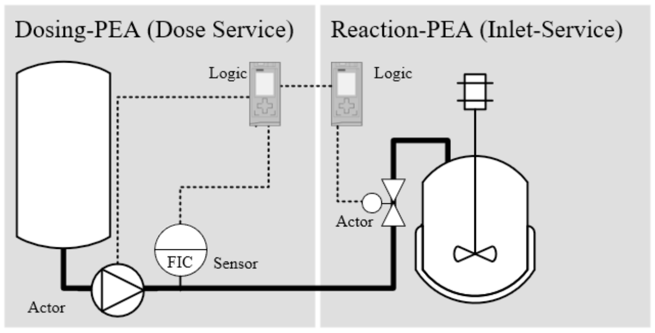

Figure 5: Technological Schema of the media transfer example

For a more detailed practical presentation of service associations by using a choreography methods it will be forwarded to the complementary contribution (Stutz et al. 2020).

## 9. SUMMARY AND NEXT STEPS

This paper presented a deeper research on services, their associations and how they can be realized with orchestration and choreography. As already shown, decentralized choreography and central orchestration provide an ideal combination of both methods. The next step is to examine the effects of decentralized choreography on a central orchestration system. Based on that, the resulting requirements for a state-of-the-art system architecture will be derived.

## REFERENCES

acatech (2019) Key Themes Industrie 4.0 – Research and development needs for successful implementation of Industrie 4.0, Research Counsil Plattform of the Plattform Industrie 4.0, München

BaSys4.0 Project Website, (2019), www.basys40.de, 13.10.2019

Bloch H.., Hensel S., Hoernicke M., Katharina S., Menschner A., Fay A., Urbas L., Knohl T., Bernhausen J., (2018), State-based control of process services within modular process plants, 51st CIRP Conference on Manufacturing Systems, 1088-1093

Brito G., Di Orio G., Barata J., (2017), Orchestrating loosely coupled and distributed components for product/process servitization , 15th International Conference on Industrial Informatics (INDIN) , 1199-1204, IEEE, Emden

Chen L., Englund C., (2017), Choreographing Services for Smart Cities: smart traffic demonstration, 85th Vehicular Technology Conference, IEEE, Sydney

Cannata A., Gerosa M., Taisch M., (2008), SOCRADES: A framework for developing intelligent systems in manufacturing, International Conference on Industrial Engineering and Engineering Management (IEEM), IEEE, Singapore

Colombo A.-W., Karnouskos S., Mendes J.-M., (2010), Factory of the Future – A Service-oriented System of modular, dynamic reconfigurable and collaborative Systems, Artificial Intelligence Techniques for Networked Manufacturing Enterprises Management, Springer, London

Colombo A.-W., Mendes J., Leitao M., Karnouskos S., (2012), Service-oriented SCADA and MES supporting Petri nets based orchestrated automation systems, 38th Annual Conference of IEEE Industrial Electronics, IEEE, Montreal

Colombo A.-W., Bangemann T., Karnouskos S., (2014), IMC-AESOP outcomes: Paving the way to collaborative manufacturing systems, 12th International Conference on Industrial Informatics (INDIN), IEEE, Porto Alegre

CORDES, Sophia et al. NAMUR-MTP for plug-&-operate in production-oriented logistics. atp magazin, v. 62, n. 1-2, p. 86-93, feb. 2020. ISSN 2364-3137

DEVEKOS Project Website, (2017), https://www.devekos.org/projekt/neue-maschinenarchitektur/, 13.10.2019

Dorofeev K., Zoitl A., (2018), Skill-based Engineering Approach using OPC UA Programs, 16th International Conference on Industrial Informatics (INDIN), 1098-1103, IEEE, Porto

Dorofeev K., Profanter S., Cabral J., Ferreira P., Zoitl A., (2019a), Agile Operational Behavior for the Control Level Devices in Plug&Produce Production Environments, 24th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Zaragoza

Dorofeev K.; Wenger M.; (2019b); Evaluating Skill-based Control Architecture for Flexible Automation Systems, 24th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Zaragoza

Gathmann, M.; (2018); Flexible thanks to modular automation; Online; (2019.11.06); Siemens AG; https://new.siemens.com/global/en/company/stories/industry/flexible-thanks-to-modular-production.html

IEC 61512-1, (1997), Batch Control – Models and Terminology, International Standard, IEC, 1997

IMC AESOP Project Website, (2013), https://cordis.europa.eu/project/rcn/95545/factsheet/en, 13.10.2019

Jammes F., Smit H., (2005), Service-oriented Paradigms in Industrial Automation, Transactions on Industrial Informatics, 62-70, IEEE

Klose, A. et.al., (2019), Orchestration Requirements for Modular Process Plants in Chemical and Pharmaceutical Industries; Chemical Engineering & Technology; Vol. 42; WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2282-2291

Lau K.-K., Di Cola S., (2015), (Reference) Architecture = Components + Composition (+ Variation Points), CobRA '15 - 1st International Workshop on Exploring Component-based Techniques for Constructing Reference Architectures, ACM Press, Montreal

Malakuti S., Bock J., Weser M., Venet P., Zimmermann P., Wiegand M., (2018) Challenges in Skill-based Engineering of Industrial Automation Systems, 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Turin

Mendes J., Leitao P., Colombo A.-W., Restivo F., (2008), Service-oriented process control using High-Level Petri Nets, 6th International Conference on Industrial Informatics (INDIN), IEEE, Daejeon

Mendes J., Leitao P., Colombo A.-W., Restivo F., (2012), High-Level Petri Nets for the process description and control in service-oriented manufacturing systems, International Journal of Production Research Vol. 50, 1650-1665, Taylor & Francis

ORCA Project Website, (2019), http://enpro-initiative.de/ENPRO+2_0/ORCA-p-275.html, 13.10.2019

Peltz C., (2003), Web Service Orchestration and Choreography, Computer 10/2013, 46-52, IEEE Computer Society

Pfrommer J., Stogl D., Kiril A., Schubert V., Hein B., (2014), Modelling and Orchestration of service-based manufacturing systems via skills, International Conference on Emerging Technology and Factory Automation (ETFA), IEEE, Barcelona

Pfrommer J., Stogl D., Kiril A., Escaida Navarro S., Hein B., Beyerer J., (2019), Plug & Produce by Modelling Skills and Service-oriented Orchestration of reconfigurable Manufacturing Systems, VDI-Bericht 2351, VDI Verlag, Düsseldorf

ProSEco Project Website, (2013), www.proseco-project.eu, 13.10.2019

ProSEco D100.4, (2014), Tecnalia Research and Innovation, ProSEco Consortium

Richards M., (2015), Microservices vs. Service-Oriented Architecture, O'Reilly Media, Sebastopol

Rumbaugh, J., Jacobson, I., & Booch, G. (1999). The Unified Modeling Language Reference Manual. http://www.temida.si/~bojan/IPIT_2014/literatura/UML_Reference_Manual.pdf. Accessed 11.11.2019

Seeger J., Arunrao R., Bröring A., (2018), Running distributed and dynamic IoT choreographies, Global Internet of Things Summig (GIoTS), IEEE, Bilbao

Seeger J., Deshmukh R., Sarafov V., Bröring A., (2019), Dynamic IoT Choreographies, Pervasive Computing, 19-27, IEEE, Kyoto

Sheng Q.Z., Qiao X., Vasilakos A.V., Athanasios V., Szabo C., Bourne S., Xu X., (2014), Web services composition: A decade's overview, Information Science 280, 218-238

SIRENA Project Website, (2005), https://itea3.org/project/sirena.html, 13.10.2019

SkillPro Project Website, (2019), https://www.iosb.fraunhofer.de/servlet/is/33500/, 13.10.2019

SOCRADES Project Website, (2009), http://www.socrades.net/, 13.10.2019

Spinelli S., Cataldo A., Pallucca G., Brusaferri A., (2018), A distributed control architecture for a reconfigurable manufacturing plant, Industrial Cyber-Physical Systems (ICPS), 673-678, IEEE, St. Petersburg

Starke G., Kunkel T., Hahn D., (2013), Flexible collaboration and control of heterogeneous mechatronic devices and systems by means of an event-driven, SOA-based automation concept, International Conference on Industrial Technology (ICIT), IEEE, Cape Town

Stutz A., Fay A., Barth M., Maurmaier M., (2020), Choreographies in Microservice-Based Automation Architectures – Next Level of Flexibility for Industrial Cyber-Physical Systems, 3rd International Conference on Industrial Cyber-Physical Systems, IEEE, Tampere, Finnland

Terzimehic T., Wenger M., Zoitl A., Bayha A., Becker K., Müller T., Schauerte H., (2017), Towards an indstry 4.0 compliant control software architecture using IEC 61499 & OPC UA, 22nd International Conference on Emerging Technologies and Factory Automation, IEEE, Cyprus

VDI/VDE GMA FA 5.16, (2019), Automation Engineering of Modular Systems in den Process Industry (Series of Technical Rules), Beuth Verlag, Berlin

VDI/VDE GMA FA 7.21, (2017), Industrie 4.0 Terms and Definition, https://www.vdi.de/ueber-uns/presse/publikationen/details/industrie-40-begriffeterms, 13.10.2019