

## Architecture and knowledge modelling for self-organized reconfiguration management of cyber-physical production systems

Timo Müller, Simon Kamm, Andreas Löcklin, Dustin White, Marius Mellinger, Nasser Jazdi & Michael Weyrich

To cite this article: Timo Müller, Simon Kamm, Andreas Löcklin, Dustin White, Marius Mellinger, Nasser Jazdi & Michael Weyrich (2022): Architecture and knowledge modelling for self-organized reconfiguration management of cyber-physical production systems, International Journal of Computer Integrated Manufacturing, DOI: [10.1080/0951192X.2022.2121425](https://doi.org/10.1080/0951192X.2022.2121425)

To link to this article: <https://doi.org/10.1080/0951192X.2022.2121425>



© 2022 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 16 Sep 2022.



Submit your article to this journal [↗](#)



Article views: 504




View related articles [↗](#)



View Crossmark data [↗](#)

# Architecture and knowledge modelling for self-organized reconfiguration management of cyber-physical production systems

Timo Müller , Simon Kamm, Andreas Löcklin, Dustin White, Marius Mellinger, Nasser Jazdi and Michael Weyrich

Institute of Industrial Automation and Software Engineering, University of Stuttgart, Stuttgart, Germany

## ABSTRACT

The demand for reconfigurations of production systems is increasing, driven by shorter innovation and product life cycles and economic volatility. Another trend in the domain of industrial automation is the emergence of cyber-physical production systems, which offer promising potentials, for example, self-organization capabilities. A suitable cyber-physical production system architecture that incorporates knowledge modelling and management concerns, plus a reconfiguration management methodology, is crucial for realizing self-organized reconfiguration management. In this paper, first reference architectures, architectural patterns, and basic principles, as well as knowledge modelling and management approaches, are discussed in general. Afterwards, these are examined concerning the reconfiguration management use case focusing on UML/XML-based and ontology-based approaches. A novel approach comprising a multi-agent system and the MAPE-K concept for reconfiguration management is presented. In addition, the approach contains a service-oriented architecture for a deterministic plant control within a layered architecture. The knowledge modelling is realized through a UML information model, which can be integrated into the system utilizing XML files. Furthermore, the provided tool support is described. It enables a user to describe system components in an effort-reduced manner and conform to the schema defined by the information model and its restrictions via a GUI.

## ARTICLE HISTORY

Received 8 February 2022  
Accepted 31 August 2022

## KEYWORDS

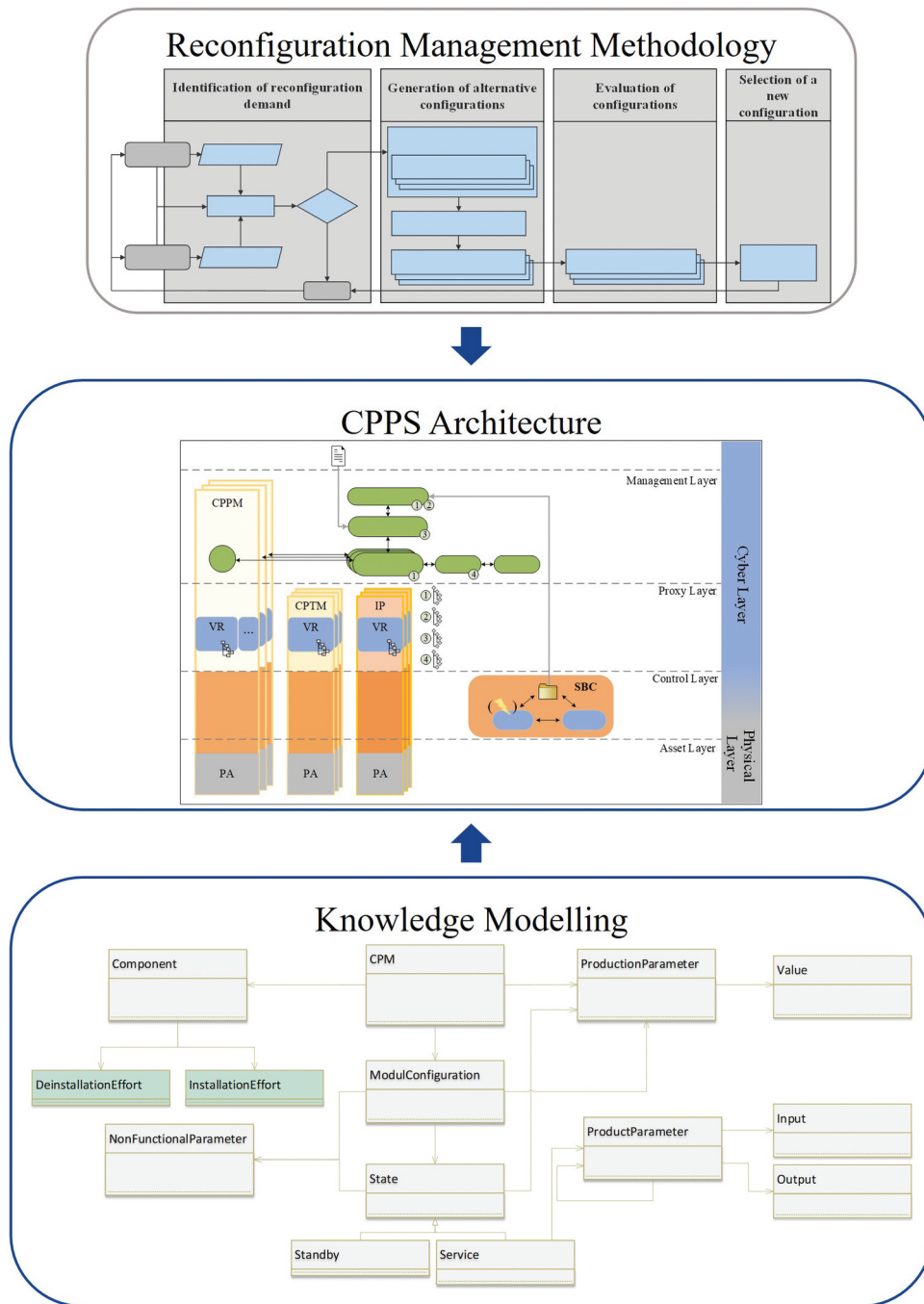
Cyber-physical production systems; architectures; knowledge modelling; reconfiguration management; ontologies

## 1. Introduction

The future of industrial automation will be shaped by the concept of cyber-physical systems (CPSs), which are physical systems with their own intelligence and cyber abilities, and will feature a high degree of intelligence (Wan et al. 2018; Grochowski et al. 2020; Vogel-Heuser et al. 2020). This is due to the promising potentials which CPSs offer for the production domain. Some of these are self-configuration or self-organization capabilities, which, e.g. lead to more cost-effective and efficient production. Furthermore, increasing demand to customize products (Zhang et al. 2016), shorter innovation and product life cycles (Köcher et al. 2020; Järvenpää, Siltala, and Lanz 2016) result in frequently changing production requirements. Therefore, objectives for production systems are becoming ever more unpredictable during the design phase of these systems. Consequently, adaptations of production systems by means of reconfigurations (i.e. adaptations during the

operational phase) have to be carried out frequently. Production systems composed of multiple CPSs are also referred to as Cyber-Physical Production Systems (CPPSs). Hence, the reconfiguration of CPPSs has become an active field of research (Hengstebeck, Barthelmey, and Deuse 2018; Balzereit and Niggemann 2020; Engelsberger and Greiner 2018) that tackles the challenges of frequent changes during the operation of future production systems.

In order to comprehensively address the reconfiguration of CPPSs in the discrete manufacturing domain, a basic concept to enhance CPPSs with self-organized reconfiguration management was introduced in (Müller et al. 2020). The applied methodology has further evolved and is presented in (Müller et al. 2021b) and (Müller et al. 2021a) in detail. However, to enable self-organized reconfiguration management, an appropriate approach for knowledge modelling and management is crucial. Furthermore, to reach the goal of self-organized reconfiguration management, a suitable CPPS



**Figure 1.** Contribution to the overall concept of self-organized reconfiguration management for cyber-physical production systems.

architecture, which incorporates the knowledge modelling and management concerns as well as the methodology, has to be defined as depicted in Figure 1. Additionally, the CPPS architecture must also include the control of the production system.

With respect to the aforementioned modelling and architecture aspects for self-organized reconfiguration management, the following requirements have to be met:

- Appropriate representation of the reconfiguration management steps within the cyber-physical production system.
- Automated execution of the reconfiguration management.
- Exploiting the potentials of cyber-physical production systems.
- The generation and evaluation of alternative configurations should consider reconfiguration

as well as production efforts by means of non-functional criteria (here: time, cost, and energy).

Based on the above, the research objectives of the contribution are as follows:

- (1) Deriving a suitable cyber-physical production systems architecture, which incorporates knowledge modelling and management concerns, as well as a reconfiguration management methodology.
- (2) Deriving a suitable modelling approach to enable the aforementioned.

This can be achieved either by relying entirely on specific approaches in the state of the art, relying entirely on a novel approach, or by combining existing approaches which can be additionally enriched with new aspects. Therefore, another objective is to:

- (3) Examine and discuss the state of the art with respect to (1) and (2) while further providing additional value for the community by means of guidance on a general level.

The remainder of this paper is structured as follows. Section 2 is dedicated to the discussion of CPPS architectures as well as knowledge modelling and management approaches. In Section 3, a generalizable CPPS architecture that supports self-organized reconfiguration management and incorporates the knowledge modelling and management is proposed. Subsequently, Section 4 provides an overview of the implementation and focuses primarily on the provided tool support. Section 5 describes the conducted evaluation. Finally, a conclusion and an outlook regarding future work are given in Section 6.

## 2. Related work

The related work is subdivided into Section 2.1 concerning basic principles as well as architectures and patterns for CPPS and Section 2.2 covering knowledge modelling and management aspects.

### 2.1. Architectures and patterns for CPPS

Since the emergence of CPPS is an evolutionary development, there are some established basic principles

that should be incorporated for the realization of these systems and their architecture as well. Therefore, principles like ‘(de-)composition’ (divide and conquer), which encompasses the ‘separation of concerns’, that in turn, together with the principle of ‘information hiding’ (including the utilization of well-defined interfaces), implements the principle of modularity, are fundamental (Gharbi et al. 2017). The principle of modularity is especially, but not exclusively, important in the context of reconfiguration as it increases the reconfigurability of a system. To that end, some techniques for a good design should be considered, which are known as the SOLID principles, i.e. the **S**ingle responsibility, **O**pen-closed, **L**iskov substitution, **I**nterface segregation, and the **D**ependency inversion principle first introduced by (Martin 2000). Based on the above, a general advice for software architects in most cases is to aim for a loose coupling and a high (functional) cohesion of components.

To cope with the increasing complexity of systems, the concept of layers has been established, which enables abstraction and separation of partial aspects of a software, such as data processing or user interaction.

Some approaches, which are recently especially popular in the context of CPS, are the multi-agent systems (MAS), utilized to provide distributed intelligence, and the paradigm of service-oriented architectures (SOAs), enabling the idea of Plug & Produce (Leitão et al. 2016; Schmidt, Müller, and Weyrich 2020). Furthermore, MAPE (or MAPE-K) incorporates the essential functions of architecture-based self-adaptation, i.e. **M**onitor, **A**nalyze, **P**lan, **E**xecute, and **K**nowledge components by means of a reference model, in order to deal with uncertainties during run time (Musil et al. 2017) and was introduced by IBM (Kephart and Chess 2003). An underlying idea of this concept is the mentioned ‘separation of concerns’.

The *Reference Architectural Model Industrie 4.0* (RAMI 4.0) (Hankel and Rexroth 2015) was set up to combine the crucial elements of the Industry 4.0 approach. In order to do so, it depicts a three-dimensional layer model consisting of the three axis: ‘Hierarchy Levels’, ‘Life Cycle & Value Stream’, and ‘Layers’. The *Layers* axis is further subdivided into the six layers: ‘Business’, ‘Functional’, ‘Information’, ‘Communication’, ‘Integration’, and ‘Asset’. As RAMI 4.0 incorporates different user perspectives, it provides a common understanding for standards and

use cases (Hankel and Rexroth 2015). However, RAMI 4.0 does not claim to depict ‘the’ architecture per se, but merely the rough framework for CPPS in the context of Industry 4.0. RAMI 4.0 assumes the use of Industry 4.0 components, i.e. components with communication capabilities that are managed as digital entities, e.g. by means of an Asset Administration Shell (AAS).

After an analysis of technology stacks concerning the application of adaptation mechanisms within the literature, the authors of (Musil et al. 2017) derived a general multi-layer architecture of CPS. It comprises six separated layers, which are denoted as: ‘Physical Layer’, ‘Proxy Layer’, ‘Communication Layer’, ‘Service and Middleware Layer’, ‘Application Layer’, and as an optional extension, the ‘Social Layer’. They observed the trend of combining different adaptation mechanisms ‘that interact and coordinate across multiple layers of the technology stack’ (Musil et al. 2017). Prominent examples of these combinations are ‘smart elements & MAPE’, ‘MAS & smart elements’ or ‘MAS & MAPE’. Whereby *smart elements* are representing physical components which are capable of self-adaptation.

The work of (Hennecke and Ruskowski 2018) aimed to consolidate the results of some previous research efforts in order to derive a reference architecture for flexible CPPS with Plug & Produce capabilities (PERFoRM architecture). Some of the incorporated work originates from IDEAS (Onori et al. 2012), SOCRADES (Colombo and Karnouskos 2009), GRACE (Leitão et al. 2015), and IMC-AESOP (Colombo, Bangemann, and Karnouskos 2014). The PERFoRM architecture (Leitão et al. 2016) consists of three layers and utilizes the SOA paradigm. According to DIN SPEC 16,593–1, which is concerned with SOAs in the context of RAMI 4.0, SOAs are one of the technological cornerstones of the I4.0 vision.

The SOA-based manufacturing middleware depicts the core element of the PERFoRM architecture to enable a seamless vertical and horizontal integration of various components (e.g. tools or production components). This architecture is one of many examples utilizing a middleware, an approach that covers the communication and information layer proposed in RAMI 4.0 (Hennecke and Ruskowski 2018). In addition to the middleware, key features of the PERFoRM architecture are standardized interfaces as well as technological adapters – together realizing a wrapper

functionality (e.g. for the integration of legacy components) – to overcome heterogeneity issues by realizing a ‘common language’ within the system.

The aforementioned describes only some prominent examples of reference architectures and patterns, whereby more approaches can be found in the literature, such as the 5C architecture, which depicts a guideline for the implementation of CPS (Lee, Bagheri, and Kao 2015). Furthermore, prominent reference architectures for the Internet of Things are the Industrial Internet Reference Architecture (IIRA) and the Internet of Things – Architecture (IoT-A), which are discussed in (Weyrich and Ebert 2015). Another category of architectural approaches is recently emerging in the context of Industry 5.0 and addresses human-centered (or human-centric) cyber-physical production systems (H-CPPSs) (Löcklin et al. 2021), such as proposed in (Sichao, Wang, and Wang 2022) or (Yuqian et al. 2022). However, instead of focussing on reconfiguration issues, they aim to achieve a better integration of human workers within CPPSs, in which machines and humans work closely together.

## Discussion

Whilst their value is undisputed, by their very nature, the aforementioned reference architectures and patterns remain at a rather abstract level and most likely cannot be directly applied without further effort with respect to the more specific use case.

Many recent architectures encompass SOAs or MASs to reach high flexibility and reconfigurability and to cope with heterogeneity (e.g. due to different component vendors), by realizing a loose coupling of distributed encapsulated functionality. It should be noted that MASs, which are realized strictly reactively, do not differ from SOAs in their external appearance (Schmidt, Müller, and Weyrich 2020). Additionally, a trend towards the utilization of middleware can be recognized within the literature.

Based on the above and depending on the use case under consideration and its requirements, the authors conclude that one of the following two options can be chosen: The first possibility is to concretize a single reference architecture or pattern, whereas the second is to develop an own architecture inspired by several existing reference architectures and patterns. The latter was chosen to obtain a CPPS



architecture that enables the self-organized reconfiguration management.

## 2.2. Knowledge modelling and management

In order to enable the realization of diverse methodologies within CPPSs, or CPSs in general, a knowledge modelling and management concept is a mandatory prerequisite. This concept cannot be considered completely detached from the systems architecture, as it has to be incorporated within the architectural approach. This means that the necessary knowledge must be held and provided to the system's distributed components in such a way that they can implement their respective functionality based on it. Here, the basis for vertical as well as horizontal integration across all layers and components of a production system is a uniform information model (Hoffmann et al. 2013; Schmied et al. 2020). This is required since CPPS (or CPS in general) consist of components that originate from diverse vendors, leading to *semantic heterogeneity* (Constantin et al. 2020). The term *semantic heterogeneity* denotes the existence of diverse ways to express different/equivalent concepts (Jirkovský et al. 2016). Another driver of heterogeneity is the previously mentioned increasing frequency of changes to production systems. Besides semantic heterogeneity, especially syntactic and structural heterogeneities have to be considered in this regard (Kamm, Jazdi, and Weyrich 2021).

Therefore, more recent approaches such as (Constantin et al. 2020) aim to achieve a system-wide *semantic interoperability*, i.e. a data and information exchange characterized by an unambiguous, shared meaning. To achieve this, the aforementioned utilization of a common information model is a prominent approach incorporated in diverse reference architectures (Reinhard, Jacoby, and Žarko 2016). However, there are several ways to achieve and maintain this common information model, especially once a system moved beyond the design phase.

In addition, information models can either be tailored for their specific use case or can be standardized by a group of stakeholders due to reusability and efficiency reasons (Schmied et al. 2020; Köcher et al. 2020). A prominent example of standardized information models are 'companion specifications' employed in OPC UA (Ref. <https://opcfoundation.org/about/opc-technologies/opc-ua>

/). An approach for a general machine model for CNC machine-tools based on OPC UA is presented in (Mourtzis, Milas, and Athinaios 2018), whilst a common OPC UA model for CPSs is proposed in (Beregi et al. 2021). Another provider of a growing number of standardized information models is the AAS community. Here, information and data are captured in a standardized way using so-called submodels, which can be imagined as digital forms and which are greatly enhanced by the inclusion of standards such as ECLASS to ensure interoperability. The definition of further submodels is the subject of current work in various standardization committees. Standardized submodels can be the basis and starting point for creating automation solutions. Besides, the concept of AAS is not in contradiction to the concept of MAS. In fact, the MAS concept can be used to implement an AAS (Vogel-Heuser et al. 2019). A more detailed view on the AAS is given in (Arm et al. 2021), which further covers an automated design and integration of the AAS for I4.0 components.

There are two prominent disciplines covering approaches to address the mentioned issue of achieving and maintaining a common information model. The first one is the well-established software engineering and the other one the ontological engineering which is gaining a lot of attention over the last years. Similarly, (Köcher et al. 2020) distinguish two different approaches for capability modelling: one utilizes ontologies as formal models while the other employs XML-based models. Capability modelling is an important part of the modelling for reconfiguration management as it is necessary to be able to compare the requirements of the products with the capabilities of the CPPS (Müller et al. 2021b).

The fact that ontologies can be informally or semi-formally specified, e.g. with UML class diagrams, entity-relationship models, and semantics nets, while they can also be formally specified, e.g. with OWL, fosters uncertainty concerning the term ontology eventually leading to a wide range of models, concepts and specifications published as ontologies (Feilmayr and Wöß 2016). The first mentioned type of representing an information model is also known as *lightweight ontology* (LWO) including concepts, their taxonomies and relations as well as their properties. In contrast, the second type is also known as *heavyweight ontology* (HWO), extending the

mentioned type by axioms and constraints (Constantin et al. 2020). Furthermore, the UML-based models (e.g. class diagrams), retrieved through software engineering, rather cover a lower semantic expressiveness as they constitute LWOs, while ontological engineering classically aims at achieving a high semantic expressiveness by developing HWOs. However, the development of a LWO does not exclude a further development toward an HWO, but it simultaneously represents a first step in this further development. It is important to notice that semantic expressiveness should be considered as a gradual rather than a binary magnitude.

According to (Sahlab et al. 2021) ontologies depict a possible realization of knowledge bases in artificial intelligence applications. While one of the most utilized definitions of ontologies is given as “a formal, explicit specification of a shared conceptualization”. (Studer, Benjamins, and Fensel 1998), the authors stick with the extended definition of (Feilmayr and Wöß 2016): ‘An ontology is a formal, explicit specification of a shared conceptualization that is characterized by high semantic expressiveness required for increased complexity’. The latter includes benefits of ‘full-fledged’ ontologies and thus helps to set them apart from other solutions and is consequently helpful for the discussion as well as for the decision for one or the other.

The authors of (Feilmayr and Wöß 2016) conducted a detailed analysis of ontologies and discuss why they have not been as successful as they could so far and how to address this issue. Furthermore, guidance on answering the question of when ontologies should and should not be used is provided, which is of particular interest to this contribution. In short, it comes down to (Feilmayr and Wöß 2016):

- Determining if a closed- or open-world assumption is appropriate,
- Choosing the representation model based on the required degree of expressiveness,
- And subsequently, based on this and the necessary degree of sharing, choosing the representation language.

Therefore, they suggest to initially consider whether an ontology is really mandatory, compared to a mere taxonomy, since the ontology implies more effort,

which in turn depends on its manifestation. However, a taxonomy itself constitutes the backbone of an ontology. One key statement to this end reads as: ‘An ontology-based development is not considered suitable for efficiently establishing homogeneous data views, data integration, and workflows’. (Feilmayr and Wöß 2016). Furthermore, ontologies tend to be complex, identifying similarities within existing ontologies versus the information to be mapped for the specific use case is a difficult and laborious task, thus hindering ontology reuse and instead leading to the development of new ontologies from scratch (Constantin et al. 2020).

There are a variety of approaches in the literature that can be roughly divided into those that use software engineering and those that use ontological engineering. First, a short excerpt of examples employing the more classical software engineering is given.

A virtual CPPS representation is used in (Hengstebeck, Barthelmey, and Deuse 2018) as a part of an assistance approach for the reconfiguration of CPPSs and to perform an attributive mapping of CPPS capabilities with production requirements for a human-robot interactive assembly process. The representation contains the CPPS structure in the AML format, which is then converted into a UML data model in order to enable and conduct the mapping.

The authors of (Siedelhofer et al. 2018) present an integrated planning tool combining a product life-cycle management system and a process simulation for reconfiguration planning of flexible assembly systems. They utilize an extended entity-relationship data model that contains a product-process-resource, a simulation and a production program partial model. Based on these, the tool provides assistance in the generation of the simulation model and in the simulation execution for different planning alternatives. Finally, the user receives evaluation parameters to enable a comparison of alternatives.

In contrast to the more common state of the art, (Hoang et al. 2019) describes a modelling approach for manufacturing resource capabilities where processes are described based on their input and output elements rather than by defined process parameters. The Product-Process-Resource (PPR) concept is employed by applying the formalized process

description (FPD) based on the VDI/VDE 3682 guideline (VDI/VDE Society for Measurement and Automatic Control).

Following some examples for the category of approaches utilizing the ontological engineering are given.

In (Ocker, Vogel-Heuser, and Paredis Felix, Vogel-Heuser, and Paredis 2019) an approach leading to a framework for feasibility feedback intended for early design phases is presented. The framework enables project-specific descriptions of resources and products that must adhere to a domain ontology. Diverse domain ontologies are further aligned with an intermediate engineering ontology, which in turn is aligned with the top-level ontology DOLCE to achieve reusability and generic queries independent of domain-specific knowledge.

The authors of (Järvenpää et al. 2019) outline the development of the OWL-based Manufacturing Resource Capability Ontology (MaRCO). A detailed resource description that, contrary to others, supports automatic inference of combined capabilities is provided. Furthermore, in (Järvenpää et al. 2018) they describe how information retrieved from a product model and a resource model is used to accomplish a matchmaking between product requirements and resource capabilities employing a process taxonomy model.

An approach for a formal model incorporating machine capabilities as well as executable skills is presented in (Köcher et al. 2020) and based on the utilization of diverse ontology design patterns (ODPs). Their modular approach aims to increase extensibility and reusability while reducing engineering effort. This is achieved since each ODP is based on a single industry standard, e.g. abstract capabilities are described by the means of processes in accordance to the FPD. To finally obtain an ontology that covers aspects of the diverse industry standards, a corresponding alignment ontology is derived, eventually serving as the capability model.

## Discussion

One issue that is covered quite well within the literature is matching product requirements with the capabilities of a (cyber-physical) production system, with a variety of approaches available. However, based on the conducted research, a lack regarding the inclusion

of non-functional aspects has been identified. Thus, applications of methodologies that incorporate non-functional criteria are not supported by these knowledge modelling and management approaches.

The main reason not to aim for a full-fledged, heavy-weight ontology for this particular use case is the effort that would be implied. This is also especially valid with respect to the lack of reusability highlighted above, which is further hampered due to the mentioned disregard of non-functional aspects. The requirements imposed by self-organized reconfiguration management in terms of knowledge representation can be met by a closed-world assumption, i.e. by providing a homogeneous, consistent view of the (distributed) information.

There is no real need for sharing a common understanding during the operational phase of the production system since there are rather syntactic and structural heterogeneities due to diverse (production module) vendors than a dynamic change of the underlying semantic. Hence, it is possible to achieve interoperability and to cope with possible vendor heterogeneity by the utilization of a common information model. In the first place, this information model defines the template for the actual models of the system components, e.g. the production modules. This approach can be further extended by the utilization of wrappers to extract and 'translate' the information of a diverse (module specific) modelling approach into the system-wide 'common language', thus coping with higher semantic heterogeneity. The diverse vendors may not want to share all their information by the means of a detailed ontology, which is typically centrally accessible. Therefore, a minimalistic shared information model is more appropriate, while further, non-shared information can be contained (within the same or additional models) and used, e.g. on the module level.

Obviously, this could also be realized by employing a full-fledged ontology. However, since the required minimalistic information model is in conflict with a high semantic expressiveness, an unused main benefit of the ontology, the effort involved is not justified.

Software agents represent a suitable technology to incorporate these different models as well as to realize the wrapper functionality mentioned above. Ontologies provide the opportunity for additional



management functions through queries and reasoning that exceed the capabilities of a mere UML/XML-based knowledge modelling approach. Even to overcome this issue, the software agent technology can be utilized to enhance the UML/XML-based approach.

From a technical point of view the following eight benefits are mentioned and described in (Feilmayr and Wöß 2016). In order to decide with respect to the given use case, it can be further distinguished which of these benefits can be exploited by a UML/XML-based approach representing an LWO and which are only provided by an HWO. Here an LWO is able to cover the ‘communication’, ‘knowledge organization’, ‘reusability’, ‘standardization’ and ‘identification’ aspects, as well as a ‘T-Box/A-Box separation’, at least to a certain extend. An HWO may further enable ‘inference’ and improve ‘constant evolution’ aspects.

However, there is no need to apply reasoning to derive new knowledge, or to enable a constant evolution by the means of an agile schema management during run time in the considered reconfiguration management use case.

Nevertheless, applying a full-fledged ontology could add additional value in some regards, which goes beyond the requirements of the given use case. This is achieved through the higher semantic expressiveness, the possibility to increase the model complexity, the inference mentioned above, and constant evolution aspects, which result in additional values such as:

- Achieving semantic interoperability capable of coping with an even higher degree of semantic heterogeneity, due to a higher semantic expressiveness. This can be realized through ontology matching, more specifically by ontology merging or ontology reconciliation, resulting in an appropriate alignment,
- Moving from a closed-world to an open-world assumption, enabled by the possibility of a dynamic evolution of the information model during run time. For instance, to achieve context awareness, i.e. to be able to handle a dynamic change during run time, which might even affect the semantics of the ontology. On a system-internal level, missing interrelations between system components could be added, or existing ones could be changed. While on an external level, the coupling or decoupling between

collaborating systems can be incorporated. Thus, the decision making could be conducted in a context aware manner,

- Depicting an appropriate knowledge representation for more than one application, due to its higher semantic expressiveness and coverage of specific domains, which will be necessary for many industrial employed CPPS,
- Inference can be utilized to gain new insights or to improve model quality (in terms of semantic expressiveness).

In short, the modelling state of the art does not consider reconfiguration as well as production efforts by means non-functional criteria (here: time, cost and energy) appropriately.

Based on the above, the authors choose to follow a novel UML/XML-based approach, as it can meet the requirements of the reconfiguration management use case and further deliver an LWO. Moreover, the LWO covers the first steps of ontology development methodologies, as, e.g. proposed in (Constantin et al. 2020). Thus, the next Section provides the proposed CPPS architecture, which incorporates the chosen UML/XML approach.

### 3. Proposed CPPS architecture with integrated knowledge representation and management

The presented CPPS architecture as well as the knowledge modelling and management approach do not contradict existing approaches. However, the specific requirements that arise when realizing CPPS, which are furthermore capable of self-organized reconfiguration management, need to be addressed. To achieve this, several elements of existing approaches (refer to the related work) were combined. The results are described below, whereby some of the more unique features are highlighted.

#### 3.1. CPPS architecture for self-organized reconfiguration management

Figure 2 visualizes the proposed CPPS architecture and its ‘elements’ for self-organized reconfiguration management, which evolved from the ‘CPPS structure’ and ‘allocation of intelligence’ depicted in

(Müller et al. 2020). It integrates the knowledge modelling and management approach as well as the functionality to implement the intelligence necessary to realize the methodology and thus enhances CPPS with self-organized reconfiguration management. The incorporated elements of existing approaches delineated in the related work Section and the respective obtained benefits are outlined at the end of Section 3.1.

The architecture is fundamentally divided into two levels: The rough division is split into the 'Physical Layer' and the 'Cyber Layer', where the first covers the physical assets and their control software. The latter adds the cyber aspect, especially by means of the models and the intelligence of the CPPS. The refined division is given by the 'Asset Layer', 'Control Layer', 'Proxy Layer', and the 'Management Layer'. These refined layers are described, in accordance with the literature and especially with the RAMI 4.0 specification DIN SPEC 91,345 (Spec 2016), as follows:

The **Asset Layer** represents the physical world and thus the totality of real existing, so called 'Physical Assets'. Those are physical elements such as production

modules, conveyor belts or products, being the physical parts of the 'Cyber-Physical Production Modules' (CPPM), 'Cyber-Physical Transportation Modules' (CPTM) or 'Intelligent Products' (IP).

The **Control Layer** serves two purposes. On the one hand, it serves as a transition layer from the physical to the information world. On the other hand, it provides access to information using a uniform data format and is used to provide services, like, e.g. the execution of certain production process, by means of an SOA. This SOA is utilized to realize the control of the CPPS necessary to conduct the production sequence for the various production orders. Thus it is referred to as SOA-based control (SBC). To that end, the CPPMs and CPTMs act as service 'Providers' while the IPs act as service 'Consumers'. The 'Registry' is realized through a central discovery server where service providers can dynamically register and deregister. Regarding RAMI 4.0, the Control Layer abstracts and merges the two RAMI-layers 'Integration Layer' and 'Communication Layer'.

The **Proxy Layer** contains both the 'Virtual Representations' (VRs) of the 'Physical Assets' as well

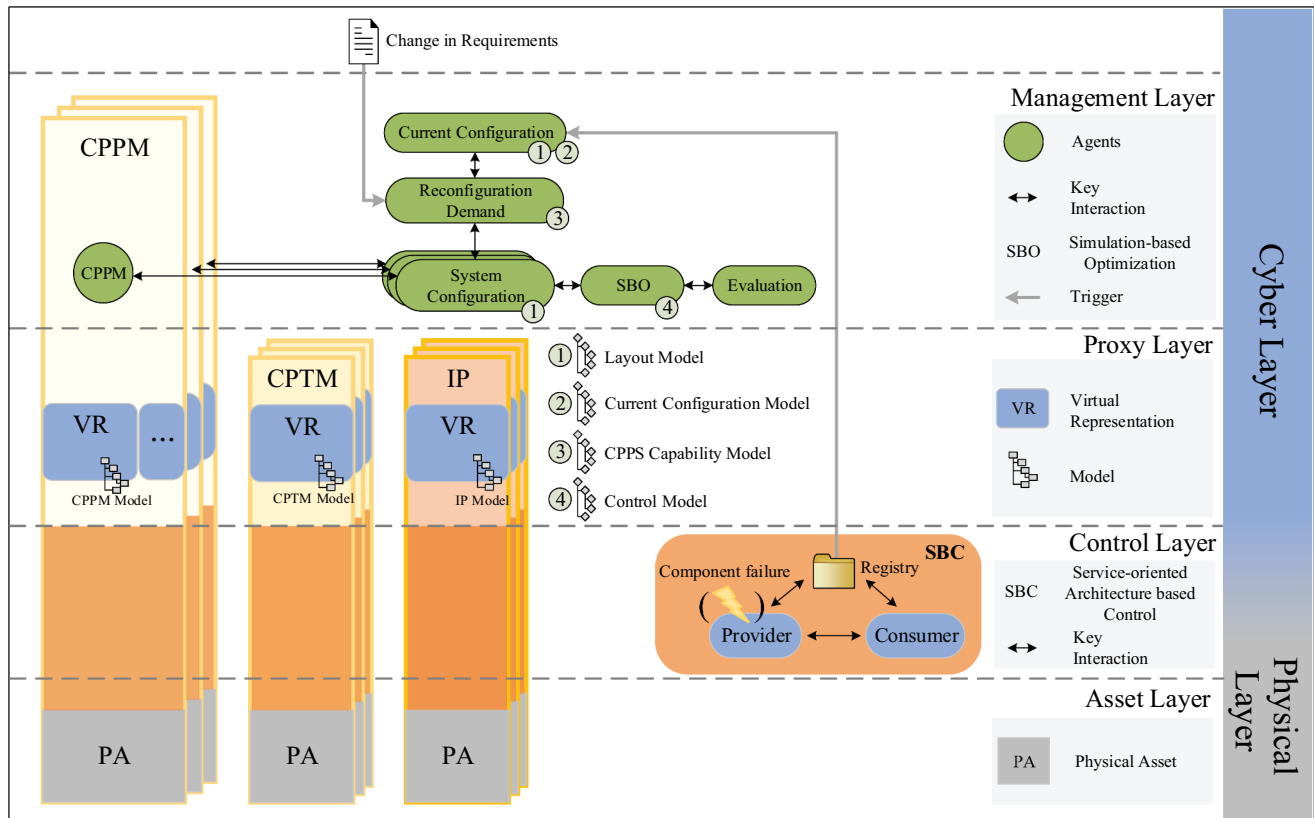


Figure 2. CPPS architecture for self-organized reconfiguration management.

as the collection of further models. The further system level knowledge necessary for the MAS, concerning the layout (1), the current configuration (2), the capabilities (3), and the control (4) of the CPPS is held in the respective models which are located in the 'Proxy Layer' and can be managed by their associated agents. This layer covers, among others, the following tasks, most of which are similar to the tasks designated to the information layer of RAMI 4.0:

- Provision of formally described models.
- Persisting the data contained in these models.
- Ensuring data integrity.

The **Management Layer** covers the intelligence to offer the desired reconfiguration management functionality. Digital representatives of each production module as well as representations of abstract assets like the system configurations interact, in a self-organized manner, to determine new system configurations. Thus, this layer covers the implementation of the methodology for the self-organized reconfiguration management, by means of a MAS. The Management Layer has similarities to the 'Functional Layer' of RAMI 4.0 and the 'Application Layer' described in (Musil et al. 2017) as mentioned in the related work.

The agents utilized within the 'Management Layer' and their interaction with the knowledge provided through the proxy layer will be briefly presented in the following:

- **Current Configuration Agent:** Administrates the 'Current Configuration Model' (2) and the 'Layout Model' (1) of the CPPS. Furthermore, it provides the current configuration to the 'Reconfiguration Demand Agent'.
- **Reconfiguration Demand Agent:** Performs the step *identification of reconfiguration demand*. To do so it reacts to occurring triggers (i.e. requirements changes or component failures) by the means of a comparison of the target production with the currently applied CPPS configuration. Therefore, it builds and manages the 'CPPS Capability Model' (3) based on the current configuration information and information from the CPPM models. Consequently, the CPPS capability model can be utilized to obtain the possible production sequences of the current configuration.

- **System Configuration Agents:** Realize the *generation of alternative configurations* in collaboration with the 'CPPM Agents' through:
  - the generation of possible system configurations for the respective production sequences.
  - determining layout variants for the generated system configurations.
  - the determination of reconfiguration efforts at system level, depending on the CPPMs reconfiguration efforts at machine level (based on the CPPM models), the current layout (1) and the determined layout variant.
  - the termination of themselves if the desired product cannot be produced.
- **CPPM Agents:** Represent CPPMs and furthermore determine whether the output product requested by a 'System Configuration Agent' can be reached through one of their services. Therefore, 'CPPM Agents' are aware of their alternative CPPM configurations, since they can incorporate multiple VRs, and the associated reconfiguration efforts (at machine level) to reach them. Thus, they even incorporate the services which could be offered in an alternative CPPM configuration.
- **Simulation based Optimization (SBO) Agent:** Performs the simulation-based multi-objective optimization of the production efforts through the optimization of production parameters (currently with respect to time, cost and energy) for each 'System Configuration Agent' representing a feasible solution. The 'SBO Agent' has the following duties:
  - Building the simulation model using the simulation-relevant data provided by the respective 'System Configuration Agent'.
  - Conducting the simulation-based multi-objective optimization considering the control logic of the real CPPS (i.e. the SBC), which is modeled in the 'Control Model' (4).
  - Passing the results to the 'Evaluation Agent'.

Due to the limitations of the available hardware and in particular the license used for the SBO, the architecture includes only one SBO agent. However, depending on the availability of hardware and licenses (or a license-independent implementation), the approach allows the introduction of additional

SBO agents and thus partial or complete parallelization. Once the 1:1 ratio between 'System Configuration Agents' and 'SBO Agents' would be achieved, the SBO role can be easily integrated into the 'System Configuration Agents', thus enabling self-optimization without external assistance.

- **Evaluation Agent:** Performs system configuration evaluation that considers both reconfiguration and production efforts and determines a utility value that furthermore incorporates the weighting of the chosen criteria (time, cost and energy). Once this has been accomplished for all the system configurations found, the system configuration with the best assigned utility is selected for deployment.

A more detailed description of the methodology for the self-organized reconfiguration management realized through the agents of the management layer is given in (Müller et al. 2021b) and (Müller et al. 2021a).

The above-mentioned essentially incorporates the representation of the MAPE-K concept. With regard to the reconfiguration of a CPPS the following applies:

- **Monitor:** Monitoring if a trigger by the means of a requirements change (e.g. a new production order) or a component failure occurs.
- **Analyze:** Conducting the *identification of reconfiguration demand*.
- **Plan:** Conducting the *reconfiguration planning*, i.e. the *generation of alternative configurations*, the *evaluation of configuration* and the *selection of a new configuration*.
- **Execute:** The *execution of reconfiguration measures* is classically achieved in a manual approach. However, the implementation of the soft- and even hardware changes may be conducted partially or fully autonomous depending on the self-(re)configuration capabilities of the CPPS components.
- **Knowledge components:** The knowledge components are realized in a distributed manner and are held within the proxy layer as described above. They enable the MAPE functionality by providing the knowledge about the CPPS and its components.

Note that the *execution of reconfiguration measures* is defined as an optional extension of the reconfiguration management. Therefore, an autonomous self-organized reconfiguration management is achieved by means of this decentralized, parallelizable approach.

The control of the production system is handled by a dedicated SOA (the SBC) rather than using the MAS. This 'separation of concerns' results in a dedicated, deterministic control that can meet real-time requirements since the communication between the two middlewares is strictly regulated.

The data access is capsulated through the agents of the MAS wherever heterogeneity could occur and can therefore be addressed by the application of wrappers as presented within the related work.

To that end, one can see that the architecture incorporates the CPTM as statically given and therefore neither have respective agents to cope with possible heterogeneity nor to integrate possible configuration alternatives within the reconfiguration management actively. This is due to the chosen realization scenario of a modular production system with a fixed matrix layout of statically specified conveyor belts. However, the concept easily allows to further cover these aspects.

Another possibility to apply wrapper functionality within the architecture is to enable the plug & produce through the SOA-based control approach despite heterogeneous service interfaces.

The proposed CPPS architecture offers a high reconfigurability and flexibility through the incorporation of basic principles as mentioned in the related work, such as the decomposition principle and especially the usage of the SOA-based control as well as of the MAS. This results in a respective prototypical implementation as presented in Section 4, which can be used to evaluate this claim. While the SOA offers reconfigurability and flexibility in terms of the underlying production system and its control, the MAS shifts these benefits to the conceptual level, i.e. to the methodology for the self-organized reconfiguration management and its implementation.

### 3.2. Proposed modelling approach

The information modelling essentially comprises the two information models for Cyber-Physical Modules (CPMs) and production orders. The proposed information model for CPMs, being either CPPMs or CPTMs, is

displayed in [Figure 3](#) and represents the core of the modelling approach. The following paragraphs describe the *classes* utilized for the information modelling.

A CPM can have multiple *module configurations* (corresponding to the multiple VRs visualized in [Figure 2](#)) and is aware which one is active as well as which components are used for a respective configuration.

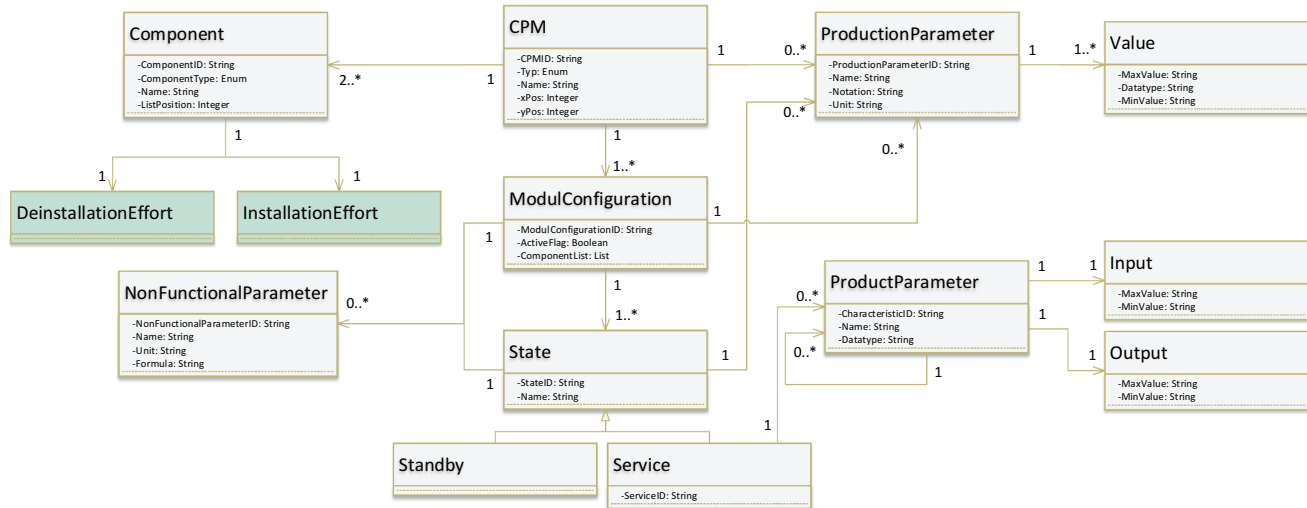
For each CPM the pool comprising all hardware and software *components* for all its possible *module configurations* is assigned with a 2.\* multiplicity. The respective *installation* and *deinstallation efforts* for each *component* with respect to the chosen non-functional criteria is linked and depicted in a simplified manner for the sake of clarity, since the criteria-based effort description includes additional aspects.

In order to describe the behavior of a CPM, the concept of state machines is employed, therefore each *module configuration* has a *standby state* and one or more *service states*. This information can be utilized to build a simulation model of respective alternative CPPS configurations and to subsequently perform the simulation-based multi-objective optimization as part of the self-organized reconfiguration management. To enable this, *non-functional parameters* can be added to the *module configurations* and to the *states*. The respective formulas express how the non-functional parameters, e.g. the energy consumption of a service, are determined. These

formulas can also consist of several *production parameters*. The *non-functional parameters* serve as intermediate parameters that are used to finally determine the criteria-specific efforts.

Furthermore, the information model allows to add *production parameters* either on the CPM, *module configuration* or *state* level. The 1.\* multiplicity of the relationship with the *Value* element allows to specify, even multiple, ranges as well as corresponding step sizes of the *production parameters*.

As stated in (Müller et al. 2021b), concerning the *Services*, the interface-oriented FPD, based on the VDI/VDE 3682 guideline is chosen. The main reason for this is that ‘it offers further degrees of freedom for an intelligent generation of configurations at machine level by the Cyber-Physical Production Modules (CPPMs) themselves, amongst other benefits’ (Müller et al. 2021b). To enable a matching of product requirements with the capabilities of the CPPS, or more precisely its CPPMs, the capabilities of the modules are described by means of process operators. Each *service* a CPPM can provide represents a process operator, based on the VDI/VDE 3682 guideline, which describes possible transformations a production resource can perform on a given input product. These transformations are defined based on a list of characteristics for the *input* state and one list for the *output* state of a product. To model this, the *product parameter* as well as the *input* and *output* elements are utilized. Moreover, the proposed modelling approach allows to refine characteristics with sub-characteristics through the relation of *product parameter* to itself.



**Figure 3.** CPM information model.



Within the self-organized reconfiguration management, this information is utilized for the *identification of reconfiguration demand* as well as for the *generation of alternative configurations* (i.e. to determine possible production sequences based on current and alternative module configurations of the CPPMs).

Since CPTM do not perform any transformation, their models do not cover this aspect. This fact is expressed through the 0-N relation between *service* and *product parameter*.

The information model of the production orders is shown in Figure 4. To enable the aforementioned, a *production order* is specified by the *product* to be produced. The *product*, in analogy to Figure 3, is described via the depicted part with the *product parameter* and the *input* and *output* states. The only difference here is that the *input* and *output* states are set to a concrete value. In addition, the *weighting* of the different (non-functional) *criteria* is assigned to the *production order*.

The IP Model contains a Bill-Of-Process (BOP) that is part of the result of the self-organized reconfiguration management and therefore contains the corresponding services and their optimized set of parameters. This is in contrast to other approaches where the production order contains the BOP, and the IP Model is derived from the production order.

Since the first three CPPS models of Figure 2, i.e. the 'Layout Model' (1), the 'Current Configuration Model' (2) and the 'CPPS Capability Model' (3) are utilized for the *identification of reconfiguration demand*, they are already discussed in (Müller et al. 2021b).

In short, the Layout Model covers the structure of the layout and the possible positioning of CPPMs within it by means of a graph. The 'Current Configuration Model' represents the information of

the currently applied CPPS configuration and is updated based on the discovery server information of the SOA. The 'CPPS Capability Model' is based on the 'Current Configuration Models' information and further utilizes the 'CPPM Models' information to derive the capabilities of the CPPS. Hence, it contains all possible production sequences that can be conducted by the currently applied CPPS configuration and it gets updated as soon as changes to the 'Current Configuration Model' occur. The last remaining model is the 'Control Model' (4), which is modeled through the utilization of state machines in order to represent the control concept of the real CPPS (i.e. of the SBC).

As derived in the discussion of the related work concerning the knowledge modelling, the proposed CPM information model defines the template for the actual models of the system components, especially for the production modules. In contrast to the literature, this modelling approach includes non-functional aspects and thus allows the application of methodologies that incorporate non-functional criteria and, in particular, enables the self-organized reconfiguration management.

While the information modelling approach can furthermore be used as a step for further development toward a standardization, in its current state it does not claim to comprehensively cover the necessary standardization aspects.

#### 4. Realization and tool support

In this Section, the prototypical implementation of the proposed approach for a CPPS architecture, with integrated knowledge modelling and management, for self-organized reconfiguration management is summarized. It serves as a proof of concept for the

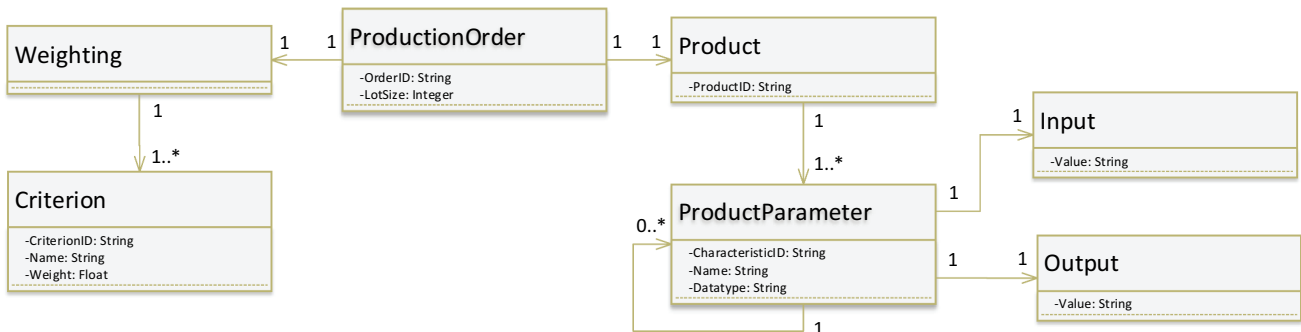


Figure 4. Production order information model.

enhancement of CPPSs with self-reconfiguration management. Throughout the Section, the focus is dedicated to the provided tool support.

Beginning from the top architecture layers, the management layer and its software agents are realized in Java, utilizing the Java Agent Development Framework (JADE). Thus, the implementation of the multi-agent system complies with the FIPA specifications (Foundation for Intelligent Physical Agents 2004) and mainly uses the concept of behaviors to realize the procedures of the software agents already extensively described in Section 3.1. The implementation was done with the Eclipse IDE, and thus a new framework for the self-organized reconfiguration management is provided. The SBO agent uses the 'MATLAB Engine API for Java' to automatically build the simulation models and perform the simulation-based, multi-objective optimization. For this purpose, each system configuration found is created and optimized in 'MATLAB Simulink' using 'Stateflow' to allow for modelling and for a discrete event simulation based on state machines running in parallel. As exemplified in Figure 5, the respective system configuration and its layout are modeled through the state

machines of the CPTMs and CPPMs. Furthermore, the products are first created within the Generator-Block (Source), then handled as messages by the CPTM- and CPPM-Blocks to conduct the required production sequences with respect to the control logic of the SBC, and at the end terminated through the Terminator-Block (Sink).

The simulation-based, multi-objective optimization minimizes the production efforts for each derived alternative system configuration with respect to time, cost and energy through an optimization that runs the simulation multiple times with varying parameters. For this purpose, the realized prototype offers several optimization approaches to choose from via GUI. Currently, it is possible to utilize a multi-objective simulated annealing or a multi-objective genetic algorithm besides the method of solving a scalarized problem described in (Müller et al. 2021a). An example of a running optimization using a simulated annealing algorithm is shown in Figure 6.

Concerning the proxy layer, upon system startup, for each available XML file, containing the information of a CPM, the 'Simple API for XML (SAX)' is utilized to

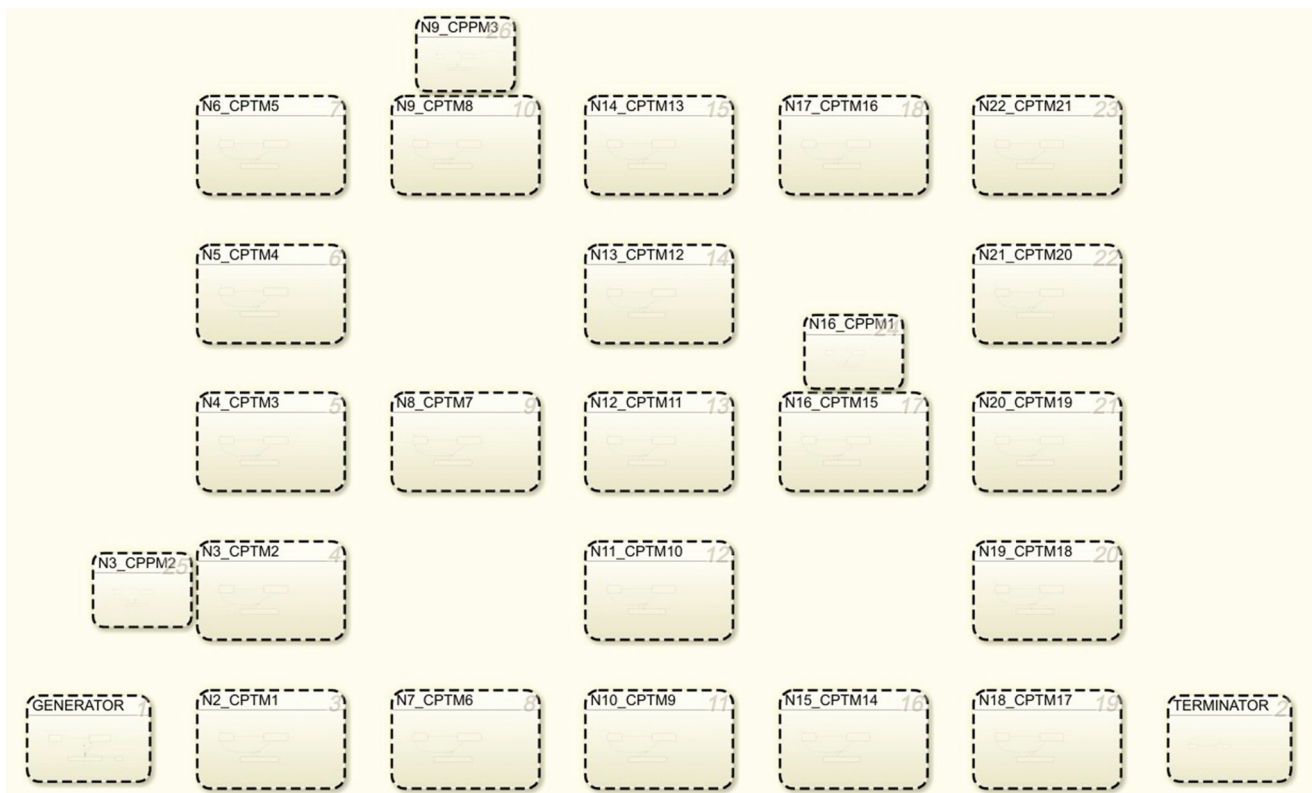


Figure 5. Stateflow simulation model of a system configuration (example).

parse the information of the XML files and integrate them into the system (Step 5 in Figure 7). Thus, the models are available and handled as described in Section 3.

To ensure that the XML files conform to the common CPM information model described in Section 3.2 and to provide tool support for their effort-reduced creation, the 'Eclipse Modeling Framework (EMF)' is utilized.

Therefore, the proposed CPM information model (Step 1) was first modeled as an Ecore model (Step 2) within the 'EMF Ecore editor'. Second, the Ecore model was transformed into Java code by EMF's built-in code generation.

After these two steps have been performed in the Eclipse IDE, the corresponding project can be executed as an 'Eclipse Application'. The user is now able to describe the respective CPM, within the defined schema of the CPM information model and its restrictions, via a GUI (Step 3). Once finished, the corresponding XML file for the CPM can be generated and saved (Step 4).

Figure 7 visualizes the entire described workflow and its result in form of an XML file, at the example of a CPPM offering a milling service. The workflow from Step 1 to Step 4 also applies concerning the production order, thus allowing for the creation of orders in the defined scheme, which can furthermore be easily utilized as an input of the prototype via a GUI.

The realization of the control and the asset layer are inspired by the physical modular production system and its SOA-based control introduced in (Schmidt, Müller, and Weyrich 2018) and utilizes the 'Prosys OPC UA SDK for Java'. The prototypical implementation covers a modular production system with a matrix layout, simulated in Unity (see Figure 8). The system provides discrete manufacturing services such as drilling, milling or punching in a highly reconfigurable and flexible fashion.

An Overview concerning the principle of the SBC is given in Figure 9. Since the modules (of the CPPMs and CPTMs) act as service providers, they are realized using both an OPC UA client and an OPC UA server component. The client component is concerned with the de-/

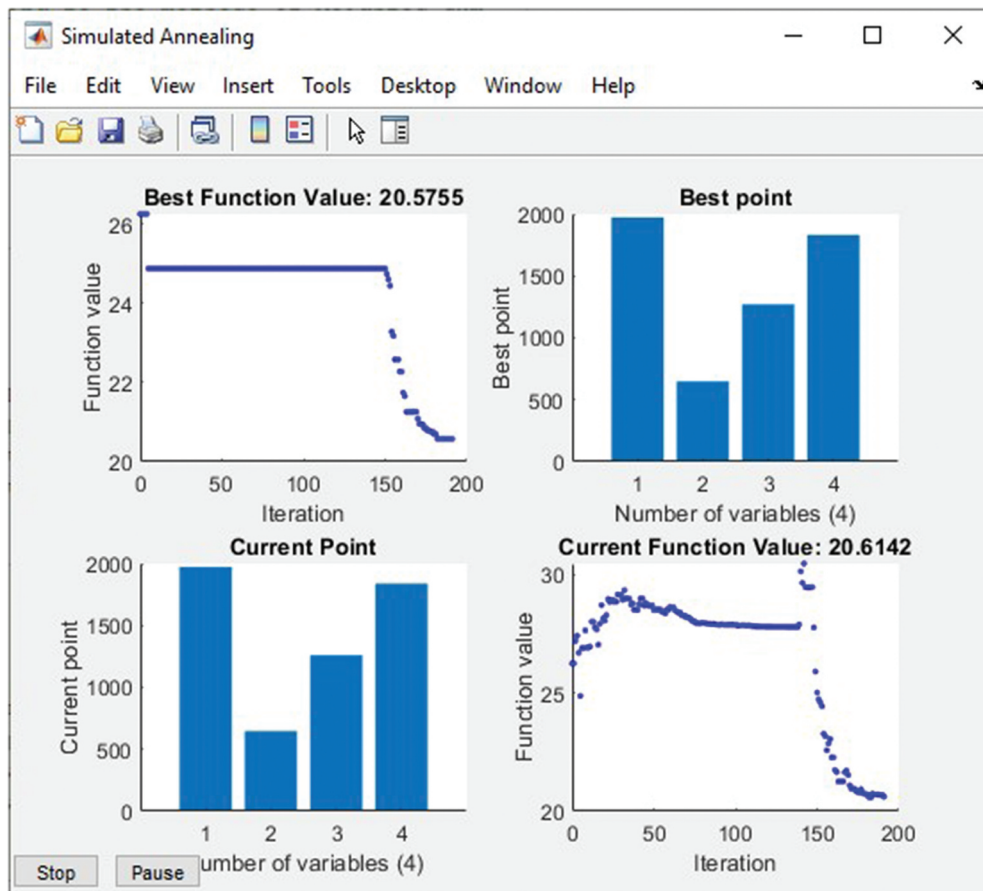


Figure 6. Optimization utilizing a simulated annealing algorithm (example).

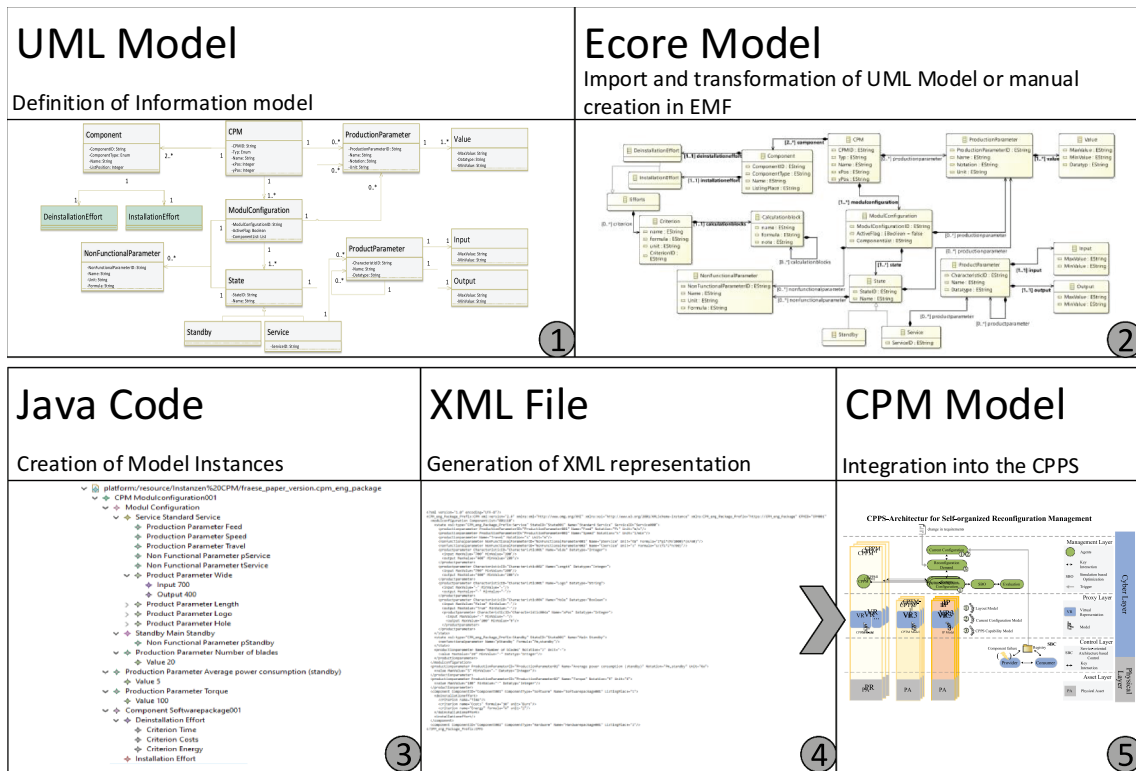


Figure 7. EMF based modelling and XML file creation workflow.

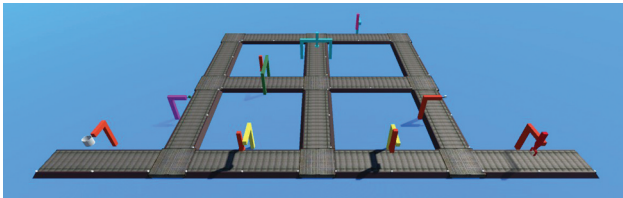


Figure 8. Unity simulation of the modular production system.

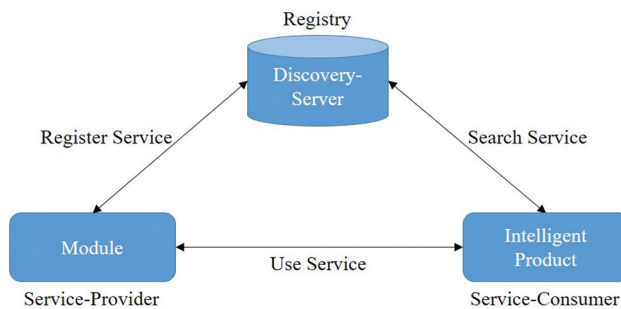


Figure 9. Principle of the applied SBC.

registration of the module and its services at the discovery server. In contrast, the server component offers the services, i.e. manufacturing or transportation services, which can be called by the intelligent products.

The discovery server depicts a service broker and is utilizing a MariaDB database for data storage and an OPC UA server component to handle the requests dynamically. The intelligent products, as described in Section 3.2, are instantiated, based on the result provided by the evaluation agent, which covers the derived BOP with the optimized parameters. Building on this, the OPC UA clients of the single intelligent products search the respective services with the help of the discovery server. Subsequently, they call those services in order to conduct the necessary production sequence.

The execution of the services is implemented through a TCP/IP connection between the modules of the SBC and their counterpart in the unity simulation, using C# to realize the production flow within the simulation.

#### 4.1. System implementation

In this Subsection a detailed description of the used hardware and software is given:

The outlined realization was implemented and evaluated on a personal computer with a Intel(R)



Core(TM) i7-7700 CPU @ 3.60 GHz (8 CPUs) and 65536MB RAM memory, the OS is windows 10. The Eclipse IDE v4.16.0 was used to develop the MAS and the SBC. The IDE further serves to conduct the modelling and XML file creation workflow described above, for which the Eclipse Modeling Framework SDK v2.5.0 was installed. Concerning Java, the JDK v1.8.0\_291 as well as the JRE v1.8.0\_311 are utilized for the MAS, EMF and the SBC. The version of the mentioned Prosys OPC UA SDK for Java is v2.2.2-638. The database of the SBC's discovery server is MariaDB v10.1.37. Regarding MATLAB, the 2021b version is chosen with Simulink v10.1, Stateflow v10.2 and the Optimization Toolbox v8.5. The Unity version for the simulation of the modular production system is v2018.2.0f2.

## 5. Evaluation

The EMF based modelling and XML file creation workflow (visualized and described in Section 4) was utilized to obtain the XML files of the CPPMs. In the current state, seven reconfigurable CPPMs with 20 configurations are available, offering 64 services. Taking into account the specified layout and depending on the production order, up to 3.2 million possible solutions are managed.

The conducted evaluation is subdivided into the three steps 'definition of evaluation scenarios', 'execution of the evaluation scenarios', and 'assessment of the evaluation scenario results'.

The **definition of evaluation scenarios** covers ten scenarios (see Table 1), each of which is specified by the selected 'trigger', the implied 'complexity', the 'start configuration' used, the chosen 'production order', and a brief 'description'.

The defined scenarios cover a rather broad spectrum. Both trigger types, i.e. requirements changes (through new production orders) and component failures, were tested several times. Different

complex scenarios were investigated, where complexity is defined by the number of considered alternative system configurations that are optimized. Thereby 0 to 10.000 refers to low, 10.001 to 50.000 to medium, and >50.000 to high complexity. Since the system configurations change continuously in reality, different initial configurations were considered as a starting point. This results in different reconfiguration efforts that lead to different alternatives emerging as winners and thus being deployed. In addition, various production orders were used, covering, for example, the production of machine parts, spacer discs, or cover sheets in several manifestations.

The **execution of the evaluation scenarios** is realized according to the following pattern:

- (1) Setup current configuration: The defined start configuration is applied to the production system in unity. The initialization of the CPPS covers the automatic integration of the XML files, as well as the automatic reception of the currently applied system configuration.
- (2) Set defined trigger: The trigger of the respective scenario, e.g. a new production order is injected.
- (3) Reconfiguration management operates: The Reconfiguration management is running and finds an appropriate solution, if a reconfiguration is necessary, the result is provided in the XML format (see example in Figure 10). It contains the positions of the CPPMs within the layout and their respective configuration, the service sequence and optimized service parameters to conduct the production sequence as well as the total effort (i.e. reconfiguration and production effort).
- (4) Execution of reconfiguration measures: The proposed reconfiguration measures are

**Table 1.** Defined evaluation scenarios.

| #  | Trigger              | Complexity | Start config. | Production order | Description                        |
|----|----------------------|------------|---------------|------------------|------------------------------------|
| 1  | New production order | Low        | 1             | Cover sheet 1    | Cover sheet with logo and holes    |
| 2  |                      |            | 2             | Spacer disc 1    | Spacer disc with holes             |
| 3  |                      |            | 3             | Spacer disc 2    | Additional threads                 |
| 4  |                      | Medium     | 4             | Cover sheet 2    | Additional reduction in height     |
| 5  |                      |            | 5             | Cover sheet 2    | Additional reduction in height     |
| 6  |                      | High       | 6             | Machine part     | Workpiece with troughs and threads |
| 7  |                      |            | 7             | Machine part     | Workpiece with troughs and threads |
| 8  | Component failure    | Low        | 8             | Cover sheet 1    | Failure in PunshingB               |
| 9  |                      |            | 9             | Spacer disc 2    | Failure in PunshingCheckingC       |
| 10 |                      | Medium     | 10            | Cover sheet 2    | Failure in PunshingCheckingB       |



- executed to realize the transition into the respective new system configuration.
- (5) Conduction of the production: The production order is finally carried out through the unity simulation and the SBC.
  - (6) Recording the key performance indicators: Hereby, the monitoring capability of the unity simulation is utilized to measure the reconfiguration and production efforts in terms of time, cost and energy. Thus, the prediction accuracy of these efforts, as part of the reconfiguration management, can be examined.

The results of the **assessment of the evaluation scenario results** is summarized in Table 2. It could be observed that the execution of all ten evaluation scenarios led to correct results. On average, there is an accuracy of 92.3% with regard to the prediction of the total time, cost and energy efforts by the reconfiguration management, compared to the values determined by the monitoring functionality of the Unity simulation.

In short, the evaluation showed that since the CPPS is enhanced with a self-organized reconfiguration management, it was able to determine whether or not there was a need for reconfiguration. If so, it was able to provide an appropriate alternative system

```
<Evaluation>
<CPPMPosition>
  <CPPM Name="CPPMFrasen0" Konfiguration="FrasenA">7</CPPM>
  <CPPM Name="CPPMBohren0" Konfiguration="BohrenC">5</CPPM>
  <CPPM Name="CPPMBohrenFrasen0" Konfiguration="BohrenFrasenB">3</CPPM>
</CPPMPosition>
<Produktionsschritte>
  <Produktionsschritt Service="MetallLochBohrenD" Nummer="1">
    <Parameter Name="rotationalSpeed">1000</Parameter>
    <Parameter Name="depth">40</Parameter>
    <Parameter Name="quantity">4</Parameter>
  </Produktionsschritt>
  <Produktionsschritt Service="MetallGewindeBohrenC" Nummer="2">
    <Parameter Name="rotationalSpeed">700</Parameter>
    <Parameter Name="depth">40</Parameter>
    <Parameter Name="quantity">4</Parameter>
  </Produktionsschritt>
  <Produktionsschritt Service="MetallFrasenA" Nummer="3">
    <Parameter Name="rotationalSpeed">1600</Parameter>
    <Parameter Name="length">300</Parameter>
    <Parameter Name="depth">5</Parameter>
  </Produktionsschritt>
  <Produktionsschritt Service="MetallMuldeFrasenB" Nummer="4">
    <Parameter Name="rotationalSpeed">1800</Parameter>
    <Parameter Name="length">50</Parameter>
    <Parameter Name="depth">5</Parameter>
  </Produktionsschritt>
</Produktionsschritte>
<Gesamtaufwand>
  <Aufwand Name="time" Gewichtung="0.45">75.63173</Aufwand>
  <Aufwand Name="energy" Gewichtung="0.26">49.904846</Aufwand>
  <Aufwand Name="cost" Gewichtung="0.29">7.4857273</Aufwand>
</Gesamtaufwand>
<Nutzwert>0.9999999701976776</Nutzwert>
<Auftragsvolumen>5</Auftragsvolumen>
<Taktzeit>5000</Taktzeit>
</Evaluation>
```

Figure 10. Resulting XML file.

**Table 2.** Results of the evaluation scenarios.

| #  | Production order | Existing reconfiguration demand? | Correct result? | Prediction accuracy |
|----|------------------|----------------------------------|-----------------|---------------------|
| 1  | Cover sheet 1    | Yes                              | ✓               | 85,90%              |
| 2  | Spacer disc 1    | Yes                              | ✓               | 97,60%              |
| 3  | Spacer disc 2    | Yes                              | ✓               | 98,50%              |
| 4  | Cover sheet 2    | Yes                              | ✓               | 98,72%              |
| 5  | Cover sheet 2    | Yes                              | ✓               | 96,56%              |
| 6  | Machine part     | Yes                              | ✓               | 82,52%              |
| 7  | Machine part     | Yes                              | ✓               | 79,03%              |
| 8  | Cover sheet 1    | Yes                              | ✓               | 96,68%              |
| 9  | Spacer disc 2    | Yes                              | ✓               | 95,19%              |
| 10 | Cover sheet 2    | No                               | ✓               | -                   |

configuration as well as a corresponding BOP in order to conduct the required production.

## 6. Conclusion and future work

In order to enhance CPPSs with a self-organized reconfiguration management, a suitable CPPS architecture, which incorporates the knowledge modelling and management concerns as well as the reconfiguration management methodology is derived in this contribution.

The discussed (reference) architectures and architectural patterns for CPPS are, while being valuable, by nature rather abstract. Thus, several elements of these existing approaches were combined to derive the novel, more specific, proposed CPPS architecture for self-organized reconfiguration management. Concerning the knowledge modelling and management, the two disciplines of software engineering and ontological engineering were discussed and compared. It was derived that an UML/XML-based approach is appropriate to meet the requirements dictated by the reconfiguration management use case, while simultaneously delivering an LWO.

The proposed, easy to generalize, CPPS architecture with integrated knowledge modelling and management has the following main features:

- Divided into two levels: 'Physical Layer' and the 'Cyber Layer'. Subdivided into four layers: 'Asset Layer', 'Control Layer', 'Proxy Layer', and 'Management Layer'.
- Incorporation of the MAPE-K concept, thus realizing autonomous behavior while realizing a 'separation of concerns'.
- Enabling the usage of wrapper functionality on two layers: on the management layer through

the MAS as well as on the control layer through the SOA.

- Utilization of the proposed information model (UML class diagram), which defines the templates for the actual CPM models, through the respective XML files used for the integration of the models.

Furthermore, the tool support realized with the 'Eclipse Modelling Framework' enables a user to describe CPMs, within the defined schema of the CPM information model and its restrictions, in an effort-reduced manner via a GUI. Thus, it is ensured that the XML files, and therefore the integrated CPM models, conform to the proposed CPM information model.

In the grand scheme of things, the main benefits of this paper can be summarized as:

- An autonomous self-organized reconfiguration management is achieved through an appropriate representation as well as an automated execution of the reconfiguration management steps within the cyber-physical production system by means of a decentralized, parallelizable approach.
- The generation and evaluation of alternative configurations considers reconfiguration and productions efforts through the inclusion of non-functional criteria (time, cost and energy). This is realized through the exploitation of the cyber-physical production systems' potentials, i.e. the models and the connectivity.
- Guidance on a general level concerning the derivation of use case specific architectures and modelling approaches for cyber-physical production systems is provided through the examination and discussion of the respective state of the art.
- A dedicated, deterministic control that can meet real-time requirements is obtained since the communication between the two middlewares (SOA-based control and MAS) is strictly regulated.
- Heterogeneity can be addressed by using wrappers, once to maintain interoperability within the MAS regarding data access of the models and on the other hand, to ensure plug & produce

through the SOA-based control regarding service interfaces.

- Offering a high reconfigurability and flexibility through the incorporation of basic principles, such as the decomposition principle and especially the usage of the SOA-based control and the MAS.

Overall, a contribution is made to increasing availability and thus to more cost-effective and efficient production.

As future work the development and integration of a full-fledged ontology, based on the ontology development methodology proposed in (Constantin et al. 2020), is planned. The highlighted incorporation of industry standards such as ISA-88 shall be addressed. The authors assume that the proposed information modelling approach is easily adaptable since it is already inspired by multiple standards. The usage of existing ODPs shall be considered as far as possible, and thereby the existing ODPs can either be evaluated or enhanced where necessary. Especially for the non-functional aspects addressed in the presented approach, the need for a new or adapted ODP is expected. Since an LWO is already available by means of an UML class diagram, a first step in this direction is already made.

On the other hand, the mentioned emergence of uniform OPC UA companion specifications, as well as standardized AAS submodels, are promising developments. However, to the best of the authors knowledge the current state of the art does not fully cover the requirements for self-organized reconfiguration management. In general, the concept of AASs can be applied as another alternative to achieve a more standardized knowledge representation. Furthermore, AASs can be combined with an UML/XML-based as well as with an ontology-based approach. Therefore, even more directions for possible future work occur.

## Acknowledgments

This work is funded by the German Federal Ministry for Digital and Transport (Synergieregion 165GU114B).

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## ORCID

Timo Müller  <http://orcid.org/0000-0002-3189-4823>

## References

- Arm, J., T. Benesl, P. Marcon, Z. Bradac, T. Schröder, A. Belyaev, T. Werner, et al. 2021. "Automated Design and Integration of Asset Administration Shells in Components of Industry 4.0." *Sensors* 21 (6): 2004. doi:10.3390/s21062004.
- Balzereit, K., and O. Niggemann. 2020. "Automated Reconfiguration of Cyber-Physical Production Systems Using Satisfiability Modulo Theories." In 2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS), Tampere, Finland. Vol. 1, 461–468.
- Beregi, R., G. Pedone, B. Háy, and J. Váncza. 2021. "Manufacturing Execution System Integration Through the Standardization of a Common Service Model for Cyber-Physical Production Systems." *Applied Sciences* 11 (16): 7581. doi:10.3390/app11167581.
- Colombo, A. W., T. Bangemann, and S. Karnouskos. 2014. "IMC-AESOP Outcomes: Paving the Way to Collaborative Manufacturing Systems." In 2014 12th IEEE International Conference on Industrial Informatics (INDIN), Porto Alegre, Brazil, 255–260.
- Colombo, A. W., and S. Karnouskos. 2009. "Towards the Factory of the Future: A Service-Oriented Cross-Layer Infrastructure." *ICT Shaping the World: A Scientific Vie*, Chapter 6, 65–81.
- Engelsberger, M., and T. Greiner. 2018. "Dynamic Reconfiguration of Service-Oriented Resources in Cyber-physical Production Systems by a Process-Independent Approach with Multiple Criteria and Multiple Resource Management Operations." *Future Generation Computer Systems* 88: 424–441. doi:10.1016/j.future.2018.06.002.
- Feilmayr, C., and W. Wöß. 2016. "An Analysis of Ontologies and Their Success Factors for Application to Business." *Data & Knowledge Engineering* 101: 1–23. doi:10.1016/j.datak.2015.11.003.
- Foundation for Intelligent Physical Agents. 2004. *FIPA Agent Management Specification*. Geneva, Switzerland. <http://www.fipa.org/specs/fipa00023/SC00023K.pdf>
- Gharbi, M., A. Koschel, A. Rausch, and G. Starke. 2017. *Basiswissen Für Softwarearchitekten: Aus-Und Weiterbildung Nach ISAQB-Standard Zum Certified Professional for Software Architecture-foundation Level: Dpunkt. verlag*.
- Grochowski, M., H. Simon, D. Bohlender, S. Kowalewski, A. Löcklin, T. Müller, N. Jazdi, A. Zeller, and M. Weyrich. 2020. "Formale Methoden Für Rekonfigurierbare Cyber-Physische Systeme in Der Produktion." *At-Automatisierungstechnik* 68 (1): 3–14. doi:10.1515/auto-2019-0115.
- Hankel, M., and B. Rexroth. 2015. "The Reference Architectural Model Industrie 4.0 (Rami 4.0)." *Zvei* 2 (2): 4.
- Hengstebeck, A., A. Barthelmey, and J. Deuse. 2018. "Reconfiguration Assistance for Cyber-Physical Production Systems." in *Tagungsband Des 3. In Kongresses Montage*



- Handhabung Industrieroboter*, edited by Thorsten Schüppstuhl, Kirsten Tracht, and Jörg Franke, 177–186. vols. 177–86. Berlin, Germany: Springer.
- Hennecke, A., and M. Ruskowski. 2018. "Design of a Flexible Robot Cell Demonstrator Based on CPPS Concepts and Technologies." In 2018 IEEE Industrial Cyber-Physical Systems (ICPS), St. Petersburg, Russia, 534–539.
- Herzog, R., M. Jacoby, and I. P. Žarko. 2016. "Semantic Interoperability in IoT-Based Automation Infrastructures." *At-Automatisierungstechnik* 64 (9): 742–749. doi:10.1515/auto-2016-0067.
- Hildebrandt, C., A. Köcher, C. Küstner, C.-M. López-Enríquez, A. W. Müller, B. Caesar, C. S. Gundlach, and A. Fay. 2020. "Ontology Building for Cyber-physical Systems: Application in the Manufacturing Domain." *IEEE Transactions on Automation Science and Engineering* 17 (3): 1266–1282. doi:10.1109/TASE.2020.2991777.
- Hoang, X.-L., S. Backhaus, R. Bense, A. Fay, D. Küstner, and B. Schulze. 2019. "An Interface-Oriented Resource Capability Model to Support Reconfiguration of Manufacturing Systems." In 2019 IEEE International Systems Conference (SysCon), Orlando, FL, USA, 1–8.
- Hoffmann, M., T. Meisen, D. Schilberg, and S. Jeschke. 2013. "Multi-Dimensional Production Planning Using a Vertical Data Integration Approach: A Contribution to Modular Factory Design." In 2013 10th International Conference and Expo on Emerging Technologies for a Smarter World (CEWIT), Melville, NY, USA, 1–6.
- Järvenpää, E., N. Siltala, O. Hylli, and M. Lanz. 2018. "Product Model Ontology and Its Use in Capability-Based Matchmaking." *Procedia CIRP* 72: 1094–1099. doi:10.1016/j.procir.2018.03.211.
- Järvenpää, E., N. Siltala, O. Hylli, and M. Lanz. 2019. "The Development of an Ontology for Describing the Capabilities of Manufacturing Resources." *Journal of Intelligent Manufacturing* 30 (2): 959–978. doi:10.1007/s10845-018-1427-6.
- Järvenpää, E., N. Siltala, and M. Lanz. 2016. "Formal Resource and Capability Descriptions Supporting Rapid Reconfiguration of Assembly Systems." In 2016 IEEE International Symposium on Assembly and Manufacturing (ISAM), Fort Worth, TX, USA, 120–125.
- Jirkovský, V. Jirkovsky, M. Obitko, and V. Marik. 2016. "Understanding Data Heterogeneity in the Context of Cyber-Physical Systems Integration." *IEEE Transactions on Industrial Informatics* 13 (2): 660–667. doi:10.1109/TII.2016.2596101.
- Kamm, S., N. Jazdi, and M. Weyrich. 2021. "Knowledge Discovery in Heterogeneous and Unstructured Data of Industry 4.0 Systems: Challenges and Approaches." *Procedia CIRP* 104: 975–980. doi:10.1016/j.procir.2021.11.164.
- Kephart, J. O., and D. M. Chess. 2003. "The Vision of Autonomic Computing." *Computer* 36 (1): 41–50. doi:10.1109/MC.2003.1160055.
- Köcher, A., C. Hildebrandt, L. Miguel Vieira da Silva, and A. Fay. 2020. "A Formal Capability and Skill Model for Use in Plug and Produce Scenarios." In 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vienna, Austria. Vol. 1, 1663–1670.
- Lee, J., B. Bagheri, and H.-A. Kao. 2015. "A Cyber-Physical Systems Architecture for Industry 4.0-Based Manufacturing Systems." *Manufacturing Letters* 3: 18–23. doi:10.1016/j.mfglet.2014.12.001.
- Leitão, P., J. Barbosa, A. Pereira, J. Barata, and A. W. Colombo. 2016. "Specification of the PERFoRM Architecture for the Seamless Production System Reconfiguration." In IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 5729–5734.
- Leitão, P., N. Rodrigues, C. Turrin, and A. Pagani. 2015. "Multiagent System Integrating Process and Quality Control in a Factory Producing Laundry Washing Machines." *IEEE Transactions on Industrial Informatics* 11 (4): 879–886. doi:10.1109/TII.2015.2431232.
- Liu, S., X. V. Wang, and L. Wang. 2022. "Digital Twin-Enabled Advance Execution for Human-Robot Collaborative Assembly." *CIRP Annals* 71 (1): 25–28.
- Löcklin, A., T. Jung, N. Jazdi, T. Ruppert, and M. Weyrich. 2021. "Architecture of a Human-Digital Twin as Common Interface for Operator 4.0 Applications." *Procedia CIRP* 104: 458–463. doi:10.1016/j.procir.2021.11.077.
- Lu, Y., H. Zheng, S. Chand, W. Xia, Z. Liu, X. Xu, L. Wang, Z. Qin, and J. Bao. 2022. "Outlook on Human-Centric Manufacturing Towards Industry 5.0." *Journal of Manufacturing Systems* 62: 612–627.
- Martin, R. C. 2000. "Design Principles and Design Patterns." *Object Mentor* 1 (34): 597.
- Mourtzis, D., N. Milas, and N. Athinaios. 2018. "Towards Machine Shop 4.0: A General Machine Model for CNC Machine-Tools Through OPC-UA." *Procedia CIRP* 78: 301–306. doi:10.1016/j.procir.2018.09.045.
- Müller, T., B. Lindemann, T. Jung, N. Jazdi, and M. Weyrich. 2021a. "Enhancing an Intelligent Digital Twin with a Self-Organized Reconfiguration Management Based on Adaptive Process Models." *Procedia CIRP* 104: 786–791. doi:10.1016/j.procir.2021.11.132.
- Müller, T., J.-P. Schmidt, N. Jazdi, and M. Weyrich. 2020. "Cyber-Physical Production Systems: Enhancement with a Self-Organized Reconfiguration Management." In 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering (ICME), Naples, Italy.
- Müller, T., S. Walth, N. Jazdi, and M. Weyrich. 2021b. "Identification of Reconfiguration Demand and Generation of Alternative Configurations for Cyber-Physical Production Systems." In *Advances in Automotive Production Technology-theory and Application*, edited by Weißgraeber Philipp, Heieck Frieder and Ackermann Clemens, 63–70. Springer.
- Musil, A., J. Musil, D. Weyns, T. Bures, H. Muccini, and M. Sharaf. 2017. "Patterns for Self-Adaptation in Cyber-Physical Systems." In *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*, edited by Biffi Stefan, Lüder Arndt and Gerhard Detlef, 331–368. Springer.

- Ocker, F., B. Vogel-Heuser, and C. J. J. Paredis. 2019. "Applying Semantic Web Technologies to Provide Feasibility Feedback in Early Design Phases." *Journal of Computing and Information Science in Engineering* 19 (4). doi:10.1115/1.4042839.
- Onori, M., N. Lohse, J. Barata, and C. Hanisch. 2012. "The IDEAS Project: Plug & Produce at Shop-Floor Level." *Assembly Automation* 32 (2): 124–134.
- Sahlab, N., S. Kamm, T. Müller, N. Jazdi, and M. Weyrich. 2021. "Knowledge Graphs as Enhancers of Intelligent Digital Twins." In 2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS), Victoria, BC, Canada, 19–24.
- Schmidt, J.-P., T. Müller, and M. Weyrich. 2018. "Methodology for the Model Driven Development of Service Oriented Plant Controls." *Procedia CIRP* 67: 173–178. doi:10.1016/j.procir.2017.12.195.
- Schmidt, J.-P., T. Müller, and M. Weyrich. 2020. "Einsatz Einer Service-Orientierten Architektur Zur Orchestrierung Eines Dezentralen Intralogistiksystems." In *Handbuch Industrie 4.0: Produktion, Automatisierung Und Logistik*, edited by M. ten Hompel, B. Vogel-Heuser, and T. Bauernhansl, 1–28. Berlin, Heidelberg: Springer.
- Schmied, S., D. Großmann, R. K. Mueller, S. G. Mathias, and U. Jumar. 2020. "Erstellung Und Management Von Informationsmodellen Für Bestehende Produktionssysteme." *At-Automatisierungstechnik* 68 (5): 325–336. doi:10.1515/auto-2020-0021.
- Siedelhofer, C., J. Schallow, P. Wolf, S. Mayer, and J. Deuse. 2018. "Simulationsbasierte Rekonfigurationsplanung Flexibler Montagesysteme." *Zeitschrift Für Wirtschaftlichen Fabrikbetrieb* 113 (4): 216–219. doi:10.3139/104.111895.
- Spec, D. I. N. 2016. "91345: 2016-04 Reference Architecture Model Industrie 4.0 (RAMI4.0)." *Din* 4: 2016.
- Studer, R., V. R. Benjamins, and D. Fensel. 1998. "Knowledge Engineering: Principles and Methods." *Data & Knowledge Engineering* 25 (1–2): 161–197. doi:10.1016/S0169-023X(97)00056-6.
- VDI/VDE 3682: Formalised process descriptions, Beuth Verlag, VDI/VDE, 2015. <https://www.beuth.de/de/technische-regel/vdi-vde-3682-blatt-1/230173798>
- Vogel-Heuser, B., M. Böhm, F. Brodeck, K. Kugler, S. Maasen, D. Pantförder, M. Zou, J. Buchholz, H. Bauer, F. Brandl, U. Lindemann, et al. 2020. "Interdisciplinary Engineering of Cyber-Physical Production Systems: Highlighting the Benefits of a Combined Interdisciplinary Modelling Approach on the Basis of an Industrial Case." *Design Science* 6. doi:10.1017/dsj.2020.2.
- Vogel-Heuser, B., A. Fay, M. Seitz, and F. Gehlhoff. 2019. *Agenten Zur Realisierung Von Industrie 4.0*. Düsseldorf, Germany: VDI/VDE-Gesellschaft Mess-und Automatisierungstechnik.
- Wan, J., Y. Boxing, L. Di, A. Celesti, F. Tao, and Q. Hua. 2018. "An Ontology-Based Resource Reconfiguration Method for Manufacturing Cyber-Physical Systems." *IEEE/ASME Transactions on Mechatronics* 23 (6): 2537–2546. doi:10.1109/TMECH.2018.2814784.
- Weyrich, M., and C. Ebert. 2015. "Reference Architectures for the Internet of Things." *IEEE Software* 33 (1): 112–116. doi:10.1109/MS.2016.20.
- Zhang, Y., C. Qian, J. Lv, and Y. Liu. 2016. "Agent and Cyber-Physical System Based Self-Organizing and Self-Adaptive Intelligent Shopfloor." *IEEE Transactions on Industrial Informatics* 13 (2): 737–747. doi:10.1109/TII.2016.2618892.