



# Semantic Web Services for AI-Research with Physical Factory Simulation Models in Industry 4.0<sup>a</sup>

Lukas Malburg<sup>1</sup><sup>b</sup>, Patrick Klein<sup>1</sup><sup>c</sup> and Ralph Bergmann<sup>1,2</sup><sup>d</sup>

<sup>1</sup>*Business Information Systems II, University of Trier, 54296 Trier, Germany*

<sup>2</sup>*German Research Center for Artificial Intelligence (DFKI), Branch University of Trier, Behringstraße 21, 54296 Trier, Germany  
{malburgl,kleinp,bergmann}@uni-trier.de, ralph.bergmann@dfki.de*

**Keywords:** Semantic Web Services, Industry 4.0, Artificial Intelligence, Flexible Cyber-Physical Workflows, OWL-S, WSMO.


**Abstract:** The transition to Industry 4.0 requires smart manufacturing systems that are easily configurable and provide a high level of flexibility during manufacturing in order to achieve mass customization or to support cloud manufacturing. To realize this, *Cyber-Physical Systems (CPSs)* combined with *Artificial Intelligence (AI)* methods find their way into manufacturing shop floors. For using AI methods in the context of Industry 4.0, semantic web services are indispensable to provide a reasonable abstraction of the underlying manufacturing capabilities. In this paper, we present semantic web services for AI-based research with physical factory simulation models in Industry 4.0. Therefore, we developed 70 semantic web services based on *Web Ontology Language for Web Services (OWL-S)* and *Web Service Modeling Ontology (WSMO)* and linked them to an already existing domain ontology for intelligent manufacturing control. Suitable for the requirements of CPS environments, our pre- and postconditions are verified in near real-time by invoking other semantic web services in contrast to complex reasoning within the knowledge base. Finally, we examine the feasibility of our approach by executing a cyber-physical workflow composed of semantic web services using a state-of-the-art workflow management system.


## 1 INTRODUCTION


Nowadays, the industry is in a transformation towards the fourth industrial revolution, also known as Industry 4.0 in the German-speaking area (Lasi et al., 2014). The predominant application of *Artificial Intelligence (AI)* methods in *Cyber-Physical Systems (CPSs)* (Broy et al., 2012) is a typical characteristic of this transformation (Lee et al., 2014). In this context, flexibility is one of the central aspects for manufacturing companies particularly because of ever shorter market launch times and increasing customer demands for individualization (Cheng et al., 2017; Lasi et al., 2014). In order to conduct close to reality Industry 4.0 research, we use

a physical *Fischertechnik (FT)*\* factory simulation model because companies are often not willing to provide data from and access to their production lines for research purposes. To use AI applications in practise, knowledge must necessarily be available in formal and machine-readable representations (Humm et al., 2020). *Semantic Web Services (SWSs)* address the problems of automatic discovering, composing, and executing by providing a declarative, ontological framework for describing them. Using AI methods (e. g., automated planning such as Marrella, 2018; Marrella and Mecella, 2018, multi-agent systems for decentralized manufacturing control such as Ciortea et al., 2018; Ocker et al., 2019, *Case-Based Reasoning (CBR)* such as Minor et al., 2014; Müller, 2018) to enhance flexibility in cyber-physical production workflows (Bordel Sánchez et al., 2018; Seiger et al., 2018) inevitably require such semantic annotations. Several related work (e. g., Lastra and Delamer, 2006; Puttonen et al., 2010, 2013) exist that already address these issues by using SWSs. However, the

<sup>a</sup> Contribution presented at IN4PL 2020 (<http://www.in4pl.org/?y=2020>). The final authenticated publication is available online at <https://doi.org/10.5220/0010135900320043>

<sup>b</sup>  <https://orcid.org/0000-0002-6866-0799>

<sup>c</sup>  <https://orcid.org/0000-0002-5077-2645>

<sup>d</sup>  <https://orcid.org/0000-0002-5515-7158>

\* <https://www.fischertechnik.de/en/simulating/industry-4-0>

currently available approaches that use SWSs in the context of Industry 4.0 focusing only on specific aspects and do not consider the entire context of manufacturing environments. Furthermore, the complex reasoning within the knowledge base makes real-time execution and monitoring of manufacturing processes difficult. In particular, we experienced these problems already in our physical simulation factory and thus it is probably not possible to use these approaches in real production settings efficiently. For this reason, we present an approach for applying SWSs to avoid complex reasoning in the knowledge base and that is therefore suitable for near real-time AI-based applications. We present three exemplary AI use cases namely multi-agent systems, automated planning, and business process management in CPSs that demand for semantically enriched *Service-Oriented Architectures (SOAs)*. Afterwards, we identify requirements from the use cases that should be fulfilled by the proposed architecture and show why SOAs are indispensable. In the following, Section 2 describes the layout of our used physical *Fischertechnik (FT)* simulation factory and presents use cases in which the application of semantic information provided by semantic web services could be important. Furthermore, related work concerning the use of web services in manufacturing and a domain ontology for physical simulation factories are presented. The developed semantic web services themselves are described in detail in Section 3 and a feasibility test is presented in Section 4. Finally, a conclusion is given and future work is discussed in Section 5.

## 2 FOUNDATIONS AND RELATED WORK

### 2.1 Semantic Web Services for Flexible Production

The paradigm of *Service-Oriented Architectures (SOAs)* can be used to achieve manufacturing flexibility as it is needed for Industry 4.0 mass customization (Lasi et al., 2014) and reconfigurability by decoupling functionality from the underlying implementation and location (Lu and Ju, 2017). For implementing SOAs, web service technologies can be used (Jammes and Smit, 2005) and, in particular, in the context of AI, these are semantically enriched, resulting in *Semantic Web Services (SWSs)*. Therefore, semantic technologies such as *Web Ontology Language for Web Services (OWL-S)* (Martin et al., 2007) for expressing the meaning of the web service interface can be used. In

general, this enables automatic discovering, composing, and executing that are required for using AI methods.

Several works propose the use of SWSs for smart manufacturing in Industry 4.0 but focusing only on partial aspects and do not consider the entire context of the shop floor. For instance, Puttonen et al. (2013) present an approach to use SWSs for executing manufacturing processes by means of three software agents represented as web services. One of these agents, referred to as Service Monitor, is a specialized web service that carries out semantic web service composition by using planning techniques w. r. t. a given production goal and the current state of the world that is provided by a domain ontology. Therefore, they use OWL for describing the state of the production system as well as OWL-S and SPARQL expressions for semantically describing the available web services that offer production capabilities. In our work, we use a digital representation connected to a corresponding physical simulation model instead of exclusively using artificially simulated data. Additionally, we focus on semantic descriptions of the underlying web services that are remotely accessible via REST in order to control manufacturing resources directly in contrast to use agents. Furthermore, we combine our SWSs with a comprehensive domain ontology of our production environment, whereas their modeled domain knowledge is limited to product definitions. Our work also focuses on a cyber-physical environment in which real-time sensor values are retrieved as part of service pre- and postconditions. Moreover, they just have two OWL-S processes, which result in 126 process variants after considering all possible permutations of descriptions.

Since modern *Cyber-Physical Production Systems (CPPSs)* (Monostori, 2014) consist of many different components and therefore many stakeholders are involved in their development process up to the later use in the manufacturing of products, Lobov et al. (2008) investigated the application of SWSs for orchestration of a flexible control. They propose OWL for modeling a Process Taxonomy, Product Ontology, Equipment Ontology, and Service Ontology and mainly discuss the responsibilities of involved persons for knowledge acquisition and maintenance rather than present their detailed semantic specification.

Most similar to our work, Cheng et al. (2017) presented an architecture and knowledge model for the integration of web services for flexible manufacturing systems. Their model includes ontologies that are similar to Lobov et al. (2008). However, the web services themselves are not semantically modeled, while Puttonen et al. (2013) develop semantic web services

but do not link them to a comprehensive domain ontology of the manufacturing environment. Based on our experiments, we assume that the continuous reasoning to evaluate the pre- and postconditions is too complex and time consuming for real world applications due to ongoing updates of the state of a CPPS and the huge number of axioms considered during the inference process.

Most recently, Järvenpää et al. (2019) present the *Manufacturing Resource Capability Ontology (MaRCO)* that aims to provide a semantic description of the capabilities of manufacturing resources. Capability parameters as configuration settings of manufacturing resources are also modeled and the capabilities are divided into simple capacities and aggregated combined capacities composed of several simple capacities. However, their approach is not related to a SOA and does not directly offer SWSs for the corresponding capacities to invoke them remotely. Thus, there is no direct connection between the modeled knowledge and the physical resources to control manufacturing. Furthermore and in contrast to our work, no preconditions and effects have been modeled for the capacities, which makes it difficult to use AI methods directly for intelligent control. Similarly, Schnicke et al. (2020) present a SOA in which services are described by their capabilities, the associated costs, and the achievable quality of the service invocation (e. g., time, costs etc.). For discovering a proper service, a match making by using tags that are linked to the services is applied. The approach does not use already established techniques to enrich web services with semantic descriptions. Thus, it is also difficult to use AI techniques directly.

All in all, there is a lack of research regarding the integration of semantically enriched web services with an existing domain ontology of a manufacturing environment to make production control more flexible as well as of research that considers the reasoning complexity in real-time applications sufficiently.

## 2.2 Industry 4.0 Simulation Factory Model

Similar to Cheng et al. (2017), we use a physical *Fischertechnik (FT)* factory model for the simulation of an Industry 4.0 manufacturing environment<sup>†</sup>. Such models are referred to as *Learning Factories* (Abele et al., 2017) and are used for education and Industry 4.0 research purposes (e. g., Calà et al., 2016; Klein and Bergmann, 2019). By using such physical simulation models, it is possible to conduct research under

laboratory conditions and to assess the suitability of AI methods and prototypical implementations before they are potentially used in practice. Furthermore, the direct impact on the physical world can be investigated and the step towards transferability to real factories is closer. When using a digital twin (Boschert and Rosen, 2016) or simulated data exclusively, this is only possible to a limited extent and with greater obstacles. Our factory consists of two identical shop floors that are linked for the exchange of workpieces as shown in Figure 1. Each shop floor consists of 4 workstations with 6 identical machines: a sorting machine with color detection, a multi-processing workstation with an oven, a milling machine, and a workstation transport that connects both of them, a high-bay warehouse, and a vacuum gripper robot. Additionally, each shop floor has individual machines, i. e., a punching machine and a human workstation in the first and a drilling machine in the second shop floor. Each shop floor is equipped with several light barriers, switches, and capacitive sensors for control purposes. Additionally, the first shop floor is enhanced with dedicated sensors such as acceleration, differential pressure, and absolute orientation sensors. RFID reader/writers are integrated in workstations on both shop floors and in the high-bay warehouse resulting in 28 communication points. This allows each workpiece to be tracked and the required manufacturing operations and parameters to be retrieved and adjusted during production as necessary. Furthermore, a camera is placed above the two shop floors to detect and track the workpieces.

The availability of a knowledge representation in form of an ontology that represents the manufacturing environment enables engineers to share, reuse, and make knowledge in formal and machine-readable way explicit. For Fischertechnik simulation factories, FTOnto (Klein et al., 2019) provides a semantic description of resources such as sensors and actuators, products and raw materials, as well as operations such as manufacturing capabilities and handling. Each physical part of the factory is represented as an individual and is ordered in a sub-class hierarchy based on an established ontology for modeling a manufacturing system called MASON (Lemaignan et al., 2006). Additionally, object properties are used to model relations between individuals. Moreover, relationships between sensors and actuators are modeled by the SOSA ontology (Janowicz et al., 2019).

<sup>†</sup> See [www.iot.uni-trier.de](http://www.iot.uni-trier.de) for more details.

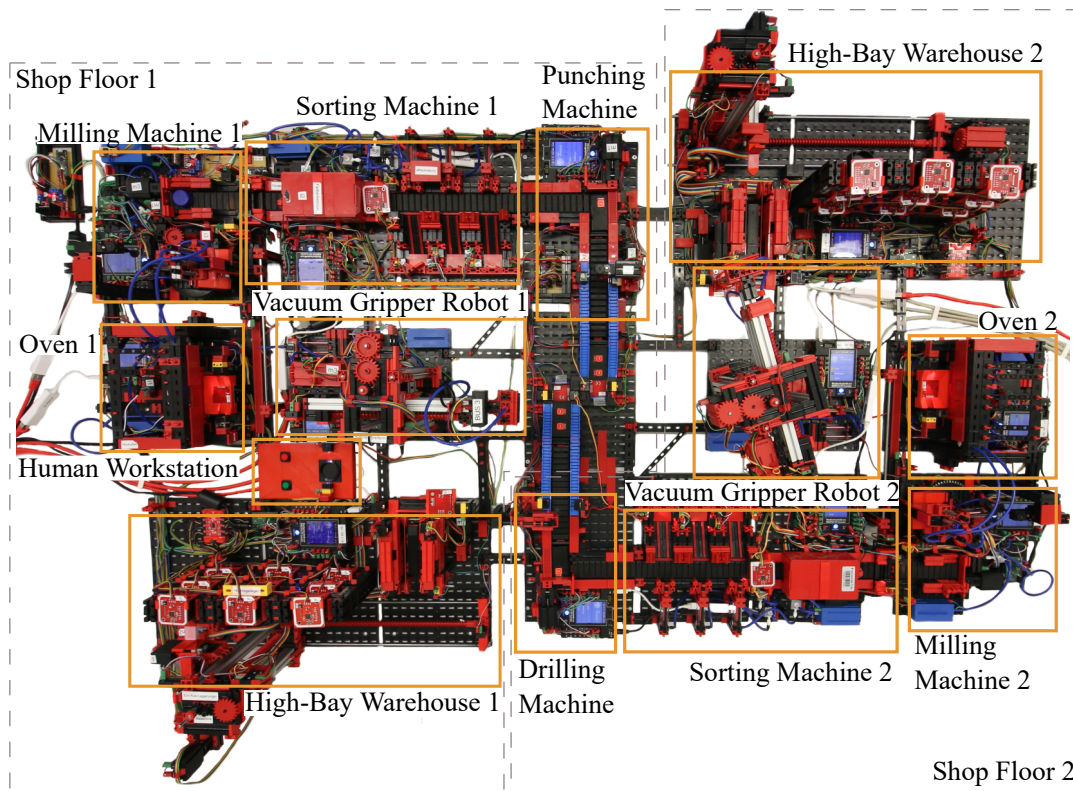


Figure 1: The Two Fischertechnik Factory Simulation Models.

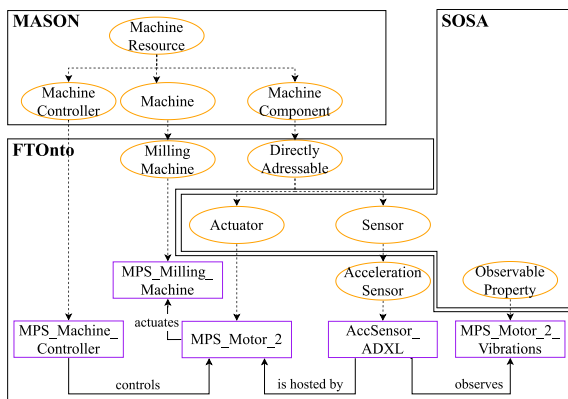


Figure 2: Subset of Domain Ontology FTOnto.

Figure 2 depicts a part of the domain ontology FTOnto. Classes are surrounded by an orange circle and instances by a purple rectangle. The dashed arrow between both states a subclass relationship or that the instance is from the type of this class. For example, *MPS\_MillingMachine* is the instance of the class *MillingMachine*. The statement that *MPS\_MillingMachine* is driven by *MPS\_Motor2* is modeled by the property *actuates*.

## 2.3 Use Cases

We now introduce three typical use cases by which we want to demonstrate the potential of semantic web services for AI-based research in the context of CPPSs.

### Multi-Agent Systems for Decentralized Control

*Multi-Agent Systems (MASs)* can be used to encapsulate functions, e. g., order processing, product design, production planning, or to represent physical manufacturing resources with their manufacturing operations. For instance, a planning agent either uses manually predefined plans that specify manufacturing operations and their sequence or generates a plan under consideration of the semantic descriptions of semantic web services that represent the manufacturing capabilities in order to achieve a given production goal such as a specific customer request (Ciordea et al., 2018). We consider a use case in which *Multi-Agent Systems (MASs)* utilize the SOA as an abstraction of physical devices to obtain information from the shop floor and to control the physical devices by invocation of web services.

## Workflow Planning of Manufacturing Processes

Similar to agent-based planning approaches, automated planning techniques can be used to build production workflows, i. e., a sequence of actions, from scratch or parts of them by using a complete domain model, an initial state, and a goal state (Marrella, 2018; Marrella and Mecella, 2018). Semantic web services enriched with preconditions and effects can be used for workflow planning. The work of Chen and Yang (2005) is one of the first in this area that introduces the basic principle. Puttonen et al. (2013) also use semantic web services and transfer the semantic descriptions into the *Planning Domain Definition Language (PDDL)* (McDermott et al., 1998) for planning of manufacturing processes. Similar to the latter approach, we propose a use case in which automated planning techniques can be used to generate cyber-physical workflows enabled by the definition of semantic web services and the transformation into PDDL.

## Business Process Management for Cyber-Physical Systems

The application of *Business Process Management (BPM)* techniques in cyber-physical environments can lead to numerous advantages. These are mainly due to the automatic retrieval of sensor data or events and are useful in determining the status of activities, making decisions about future process flow, and detecting deviations at an early stage (Janiesch et al., 2020). The SOA provides the services that can be used to execute process activities or to acquire sensor data values. For our use case, we consider that an intelligent system for BPM needs to access this data in different process steps or within a specified time frame in order to use it properly. For example, *Process-Oriented Case-Based Reasoning (POCBR)* (Minor et al., 2014; Müller, 2018) has shown great potential as an experience-based activity in similar research fields (e. g., in the cooking domain to represent cooking recipes as workflows in Müller (2018) or in the domain of scientific workflows in Zeyen et al. (2019)) by using best-practise workflows from a case base. This approach can perhaps also be used to increase workflow flexibility in CPPSs.

## 3 SEMANTIC WEB SERVICES FOR AI-BASED RESEARCH

In this chapter, our *Service-Oriented Architecture (SOA)* with semantically enriched web services for

applying them in the use cases previously described is presented. Note that the SOA provides the services to retrieve data for implementing a digital twin (Boschert and Rosen, 2016). The development process of our SOA follows the well-known ontology development methodology by Sure et al. (2009) that contains the four steps *Kickoff*, *Refinement*, *Evaluation*, and *Application & Evolution* (see Figure 3) and that can also be applied for developing semantic web services. The steps are described in detail in the following.

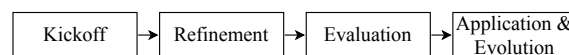


Figure 3: The Ontology Development Process by Sure et al. (2009).

### 3.1 Requirements

In this section, we introduce as part of the *Kickoff* phase *requirements (RQs)* that a SOA should meet in order to be suitable for AI-based research activities in Industry 4.0. The requirements have been primarily derived from the use cases presented in Section 2.3. We do not divide the requirements according to these use cases, because most requirements are essential for more than one use case.

**RQ 1 – Provide Interoperability and Interconnectivity.** The SOA should be developed to achieve interoperability and interconnectivity from several controller types and programming languages (Burns et al., 2019; Puttonen et al., 2013).

**RQ 2 – Connect to Existing Knowledge Representations.** The services should be linked to an existing knowledge representation in form of a domain ontology. Domain ontologies are important for modeling production systems and for the transformation towards Industry 4.0 (Puttonen et al., 2013). Due to the connection of semantic web services with an existing ontology, it is possible to directly check the result of a web service that returns sensor values for plausibility. Furthermore, it is possible to identify services that are currently not available due to an error of a physical manufacturing resource.

**RQ 3 – Enable Ontology and Knowledge Base Updates and Real-Time Verification.** In the context of Industry 4.0, it is necessary to use real-time data to make proper decisions (Burns et al., 2019; Lasi et al., 2014). Thus, it is necessary to keep the instances in the knowledge base up to date (Lobov et al., 2008).

**RQ 4 – Abstract from Low-Level Control Commands.** It is important to abstract the web services from individual low-level control commands to create a hardware abstraction layer for CPSs. In order to use the web services in a business process management context, each service should additionally perform an atomic operation (Puttonen et al., 2019). It is a necessary requirement to be able to break down end-to-end processes (see Challenge 7 in Janiesch et al., 2020).

**RQ 5 – Enrich Web Services with Semantic Descriptions.** Due to the semantic enrichment of web services in a SOA, it enables the use of AI-based technologies such as those presented in the use cases for improving resource utilization and for controlling the execution of the whole manufacturing processes, i. e., preconditions and effects can be used to determine the impact of certain activities (see Challenges 15 and 16 in Janiesch et al., 2020 and also Hepp et al., 2005).

**RQ 6 – Model Relationships between Web Services.** Relationships and interconnections between atomic as well as composed web services should be modeled to determine dependencies between services of individual manufacturing resources and to identify semantically similar services (see Challenge 6 in Janiesch et al., 2020).

**RQ 7 – Parameterization of Web Services.** Reconfigurability of production systems is needed to satisfy changing process goals. By parameterizing web services that represent the possible capability configuration settings of physical manufacturing devices, different production goals can be achieved (Lobov et al., 2008).

**RQ 8 – Resolve Mutual Exclusion.** To prevent multiple access to a single physical manufacturing resource, the SOA should ensure that only one service may have simultaneous access to this resource to execute the corresponding operation (Lobov et al., 2008). During this time, however, further service requests should not be lost, but saved.

**RQ 9 – Orchestrate and Composite Web Services.** It should be possible to orchestrate or composite the developed services to more complex processes (Lobov et al., 2008) and in order to fulfill an overall process goal (Puttonen et al., 2013, 2019).

### 3.2 Architecture Overview

Our concept integrates a SOA composed of corresponding semantic web services with a domain ontology in an architecture of a CPS, which is, in fact,

the foundation for research on AI-based control for flexible manufacturing processes. More precisely, we adopt the layered architecture for managing cyber-physical workflows proposed by Marrella and Mecella (2018) depicted on the right side as a basis to define a SOA as a service layer. Additionally, we linked the SWSs of the service layer to an existing domain ontology of a manufacturing environment and to OWL-S. An overview of the proposed architecture is shown in Figure 4. Starting from the bottom, the *Cyber-Physical Layer* represents the manufacturing environment where several cyber-physical production workflows are executed. We propose to represent each resource (e. g., actuator, sensor etc.) as well as their relationships in a domain ontology. On top of this layer, the *Service Layer* is responsible for transmitting and executing control commands to the *Cyber-Physical Layer* and receiving responses (e. g., raw data from sensors). Therefore, this layer contains semantic web services on different abstraction levels for control commands as well as for sensor data retrieval. To describe the services semantically, each service contains a semantic description such as proposed by OWL-S or the *Web Service Modeling Ontology (WSMO)* (Roman et al., 2005) respectively. In our concept, we re-model parts from both of them for semantic enrichment of the web services. The resulting service ontology is merged with the domain ontology FTOnto to relate physical resources to their corresponding services. Finally, these web services are accessible via REST using a web server that can be seen as an example of a *Service Layer* implementation. Thereby, each service has a *Service Grounding* that describes the access to the actual service of the *Service Layer*. In this regard, the *Service Layer* constitutes the connection between the *Enactment Layer* and *Adaptation Layer* with the *Cyber-Physical Layer*, whereby the web server communicates with both upper layers and subsequently executes the desired web services. The *Enactment Layer* is responsible for the execution of modeled cyber-physical workflows by using the corresponding underlying SWSs. If modeled cyber-physical workflows can not be executed as planned, adaptations may be necessary. For this reason, the *Adaptation Layer* uses AI methods such as presented in the use cases for intelligent production control. Therefore, *Enactment Layer* and *Adaptation Layer* need access to the *Service Profile* for discovering services and to the *Service Model* for how the service works. In addition, the *Design Layer* allows users to exploit the knowledge modeled in the *Service Profile* to find appropriate web services during workflow modeling.

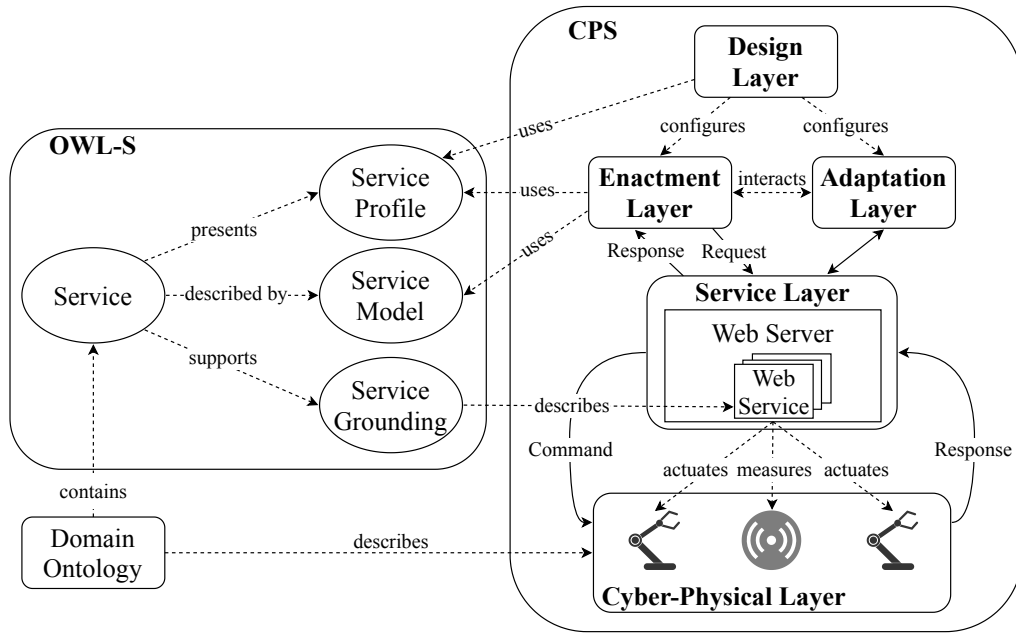


Figure 4: Service Conceptualization for Flexible Production Control (Based on Marrella and Mecella 2018).

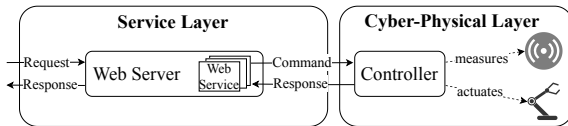


Figure 5: Illustration of the Execution Sequence of a Web Service Invocation.

Figure 5 shows how the invocation of a web service results in a concrete action in the *Cyber-Physical Layer*, where the functionality of the web service is executed. The controller that manages the corresponding physical manufacturing resource receives the request to execute a certain command, if necessary with special capability parameter settings. The controller actuates the managed device, measures the corresponding value of a sensor or performs a basic configuration of the device. Afterwards, the raw data or generally the response is transferred back to the web server, i. e., the sensor value or the message that the command has been executed both with a start and end time stamp.

### 3.3 Semantic Web Services for a Physical Factory Model

Based on the presented architecture, a *top-down* approach has been applied to identify and develop 70 SWSs that allow to use independently all functions of the physical devices of the complete shop floors presented in Section 2.2. The approach has been used for the identification of services and selection according

to the level of abstraction corresponding to the use cases (see RQ 4). All in all, 70 SWSs have been derived in the *refinement* phase and due to the variety of possible parameter values, there exist 364 different configurations of the web services (see RQ 7). During this phase, pre- and postconditions have been modeled as web services, results with included effects have been added, and all web services have been linked to the existing ontology (see RQ 2). For using AI planning techniques, the modeled preconditions and effects can be converted into a planning model. The provided services are divided into services that perform an activity in the physical factory and into services that are offered for measuring sensor data or for configuring device settings. Furthermore, the services are hierarchically ordered into several more specific classes (see RQ 6). This is illustrated in Figure 6 with a class diagram of the physical resources from our FT simulation factory (cf. Figure 1). The methods specified in the classes are encapsulated as web services and we classified the web services into sensing and configuration services as well as activity services according to their purpose and separated by a dashed line. Please consider that the *General Controller* interface only provides generic sensing and configuration services that are reused by the actual controllers. For example, the method *pickUpAndTransport* of the VGR has two capability parameters (*start* and *end*) for specifying the start and end position of a workpiece transport. The method is similarly implemented as a web service with the same

two parameters. Due to the variety of positions where a workpiece can be picked up and dropped off, this single web service results in 72 different configuration possibilities; the superscript number above the individual methods illustrates this. Each of these possibilities of different positions has specific preconditions and effects depending on the given parameter values for *start* and *end*. By encapsulating services of different controllers of the cyber-physical layer, our SOA contributes to achieving interconnectivity and interoperability by providing RESTful web services (see RQ 1). To describe the services semantically, we have re-modeled the important concepts of OWL-S and WSMO for our work and have especially tailored them to our simulation context. This means that we only have one service class, corresponding to the Service Profile in OWL-S, that describes the functionality of the service w. r. t. its parameters, inputs, outputs, preconditions, and results. Additionally, the description of postconditions is added and re-modeled from WSMO. These semantic descriptions can be used to determine what functionality the service provides, what requirements have to be fulfilled for execution, and how its successful execution can be verified (see RQ 5). Whereas most previous work updates the knowledge base continuously and verifies the condition expressions based on the current real world state, we link to other semantic web services and use their response for our condition verification. This means that conditions are evaluated in near real-time based on sensor data that is accessed via a web service invocation (see RQ 3). This functionality is similar to what one would expect from a digital twin. The web service used to check the conditions can in turn require further web service invocations for condition verification (see RQ 6 and RQ 9). To handle multiple parallel requests to one physical resource, a priority queue according to the First-In-First-Out principle is implemented and services are only executed by the web server when all required resources are not blocked by a previous request (see RQ 8). The goal of the queue is to store incoming web service requests to ensure that no conflicts between multiple requests occur during execution of different resources. After the service is carried out, we control the execution with postconditions that have to be satisfied to determine whether a particular activity has been successfully performed. If all postconditions are fulfilled, the successful response is given to the client, otherwise, the client receives a message which postconditions are not satisfied.

In the following, we present one of the modeled semantic services in more detail. For this purpose, the already mentioned *pickUpAndTransport* service of

---

```

1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl: <http://www.w3.org/2002/07/owl#>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
5 PREFIX ftono: <http://iot.uni-trier.de/FTOnto#>

7 SELECT ?service ?precondition ?preconditionCheckService ?
      checkURL ?requiredKeyInServiceResponse ?
      requiredValueInServiceResponse
8   {
9     ?service ftono:hasURL "http://127.0.0.1:5000/vgr/
      pick_up_and_transport?machine=vgr_1&start=sink_1
      &end=oven"^^xsd:string .
10    ?service ftono:hasPrecondition ?precondition .
11    ?precondition ftono:isCheckedBy ?preconditionCheckService
      .
12    ?preconditionCheckService ftono:hasURL ?checkURL .
13    ?precondition ftono:requiredKeyInServiceResponse ?
      requiredKeyInServiceResponse .
14    ?precondition ftono:requiredValueInServiceResponse ?
      requiredValueInServiceResponse .
15  }

```

---

Listing 1: SPARQL Expression for Retrieving Preconditions.

the VGR is selected since it contains most of the annotated semantic elements such as pre- and postconditions as well as results. Figure 7 illustrates the semantic annotations and their relationships as a graph. In this context, green rectangles with rounded corners represent data properties, violet rectangles represent instances of classes that are in turn represented by orange ellipses. If invoked, this service fulfills the function of picking up a workpiece at sink one of the sorting machine (*start*), transporting it to the oven, and eventually dropping it off at the oven (*end*). Before the execution starts, the request is scheduled in the queue and as soon as no other request comes first, the execution of the service starts. At this point, access to the physical resource for other clients is blocked and intermediate requests are stored in the queue. The first part of the execution is the check of the preconditions. In this case, the SPARQL query as shown in Listing 1 is executed and returns five preconditions that must be fulfilled (see Figure 7). For instance, the oven must be available and ready and the light barrier that monitors the end position of the transport must not be interrupted, because that indicates an empty storage space. In particular, these preconditions refer to other semantic web services and their responses can be evaluated outside of the knowledge base but using the verification rules modeled in the knowledge base. This procedure enables to perform the verification in near real-



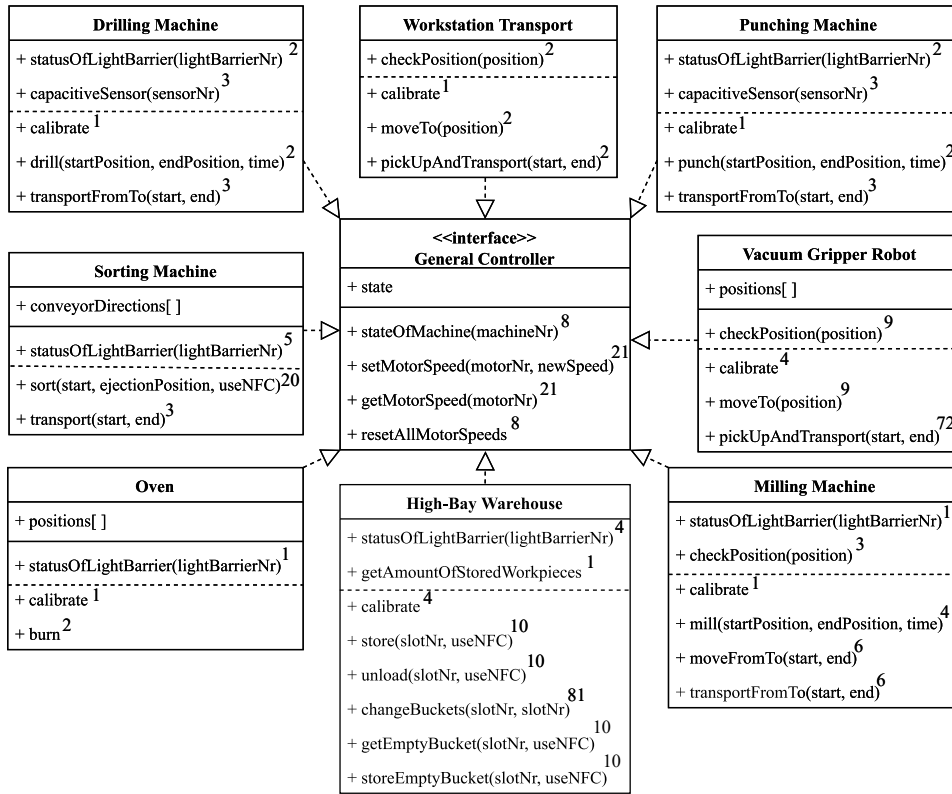


Figure 6: Class Diagram of the Physical Manufacturing Resources.

time without importing large amounts of raw sensor data produced with high frequency required for complex reasoning within the knowledge base for precondition verification. This could otherwise lead to considerable overhead for reasoning and possibly wrong, not real-time information for decision making. For providing real-time data, web services to retrieve the status of a sensor (e. g., a light barrier) are handled by a separate queue as the web services that initiate manufacturing operations. This division enables an immediate result even if the corresponding machine is still performing a manufacturing operation. The described principle is also applied to postconditions that are semantic web services, too. The exemplary service contains one postcondition that checks whether the service has been executed successfully. The postcondition checks whether the light barrier, which was not interrupted for the corresponding precondition, has now been interrupted, i. e., it is verified that the workpiece has been transported from the first sink of the sorting machine to the oven and thus the execution was successful. For each service, regardless of whether it is used as a pre- or postcondition or neither, the respective URL to invoke the service is represented as instance (see Figure 7).

## 4 FEASIBILITY TEST

In this section, we demonstrate the usage of our developed SOA as part of the *Evaluation* phase of the ontology development methodology for usefulness for the described use cases in Section 2.3. Therefore, we have prototypically implemented the third use case by using the workflow management system Camunda<sup>‡</sup>. Camunda is able to invoke web services by using *Business Process Model and Notation (BPMN)* conform *Service Tasks*. We implement a small cyber-physical production workflow that transports a workpiece from the first sink of the sorting machine to the oven, burns it, and after a quality inspection by an employee, it is transported and stored in the high-bay warehouse. Figure 8 illustrates the described manufacturing process as a BPMN diagram. The corresponding web server receives the web service invocations from Camunda and forwards the execution command to the controller of the specified physical device. Before execution, an implemented Python class checks if the preconditions and after execution, the corresponding postconditions, of the service are satis-

<sup>‡</sup> <https://camunda.com/>

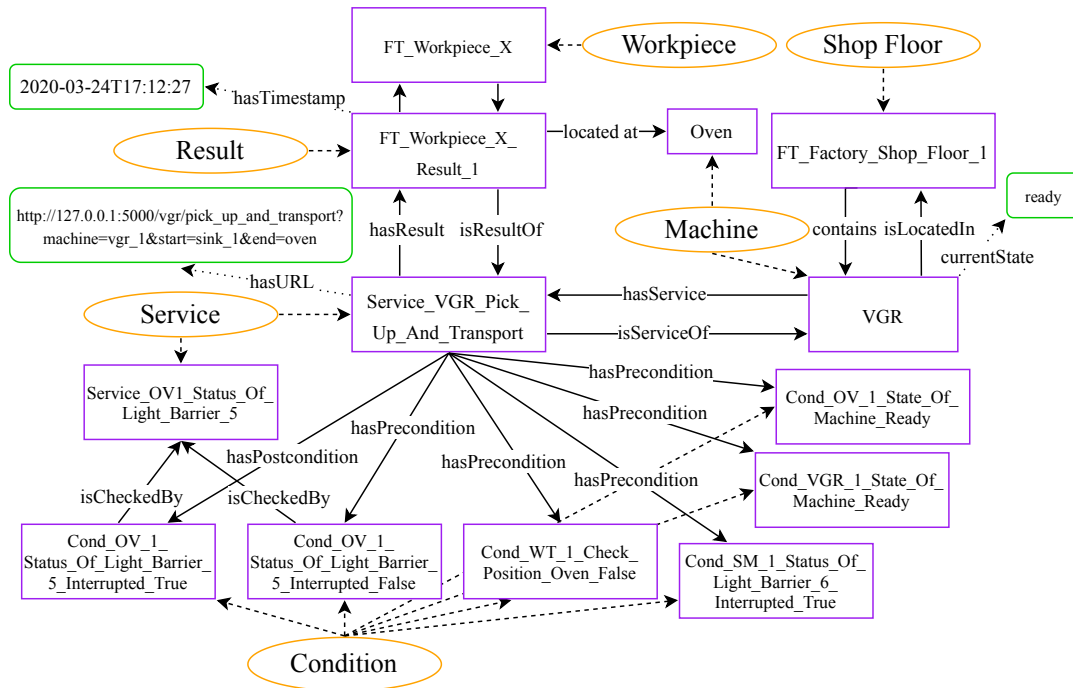


Figure 7: Semantic Annotations of the Pick Up and Transport Service from Vacuum Gripper Robot as a Graph.

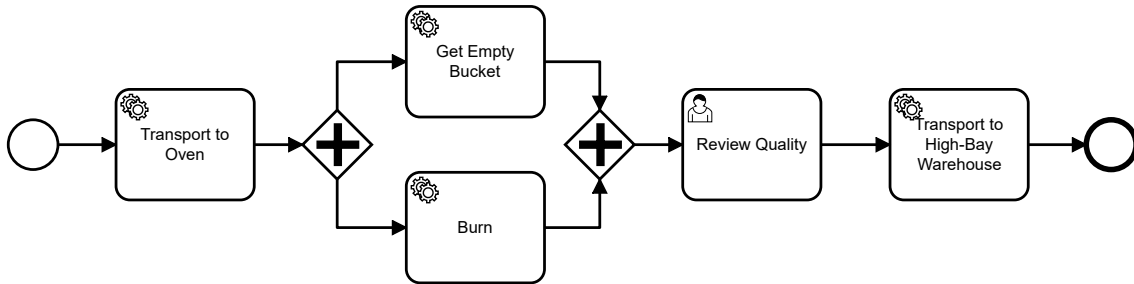


Figure 8: A BPMN Diagram of a Manufacturing Process.

fied by using OWLReady<sup>§</sup> (Lamy, 2017). Moreover, we also evaluated a failure scenario where the workpiece is lost during the transport to the oven. In this case, the precondition that the workpiece is located at the working platform on the oven to execute the burn activity is not met and thus not executed. After locating the workpiece at the required place, the burning procedure is executed immediately. With this example, we aim to demonstrate the feasibility of our approach. In the same way, we have also evaluated the functionality for all other services as well as for more complex cyber-physical production workflows. In summary, the example application demonstrates the feasibility that our SOA is appropriate to execute SWSs with a state-of-the-art workflow management system. Additionally, the re-modeled pre- and postconditions from OWL-S and WSMO and the un-

<sup>§</sup> <https://pypi.org/project/Owlready2/>

derlying concept to determine their verification with other SWSs is suitable for the presented use cases and works as designed in near real-time. The developed SWSs were checked for conformity and correctness by using the *Ontology Pitfall Scanner! (OOPS!)*<sup>¶</sup>. In this process, no errors or inconsistencies have been detected.

## 5 CONCLUSION AND FUTURE WORK

In this work, we present a *Service-Oriented Architecture (SOA)* with semantic web services for AI-based research with physical factory simulation models in Industry 4.0. First, we described use cases in which

<sup>¶</sup> <http://oops.linkeddata.es/>

service-oriented architectures with semantic web services are valuable. Based on the described use cases, requirements have been derived that should be met. As a result, we modeled 70 semantic web services based on standards such as OWL-S and WSMO. The semantic services are enriched with inputs, outputs, preconditions, and results with postconditions. Additionally, we have integrated our approach in a cyber-physical business process management architectural model. In our feasibility test, we have exemplarily demonstrated that the developed SOA is suitable to be used in state-of-the-art workflow management systems to build valid cyber-physical production workflows. By using semantic web services as pre- and postconditions of other semantic services, a near real-time verification for executing cyber-physical workflows is ensured. Additionally, the developed services provide the foundation to support low-code applications (Sanchis et al., 2020).

In future work, we investigate the described use cases further to enhance workflow flexibility in cyber-physical production systems. We assume that *Process-Oriented Case-Based Reasoning (POCBR)*, for example, could be used for this purpose because it has already shown great potential in various other domains. Additionally, automated planning techniques can be used to further increase workflow flexibility in Industry 4.0. Therefore, the pre- and postconditions of the semantic web services can be transferred into a planning model and directly used during planning. In this context, we plan to conduct an comprehensive experimental evaluation that assess the appropriateness of the presented approach in more detail. With the future application, the semantic web services are continuously improved according to their environment and requirements (see *Application & Evolution* phase).

## ACKNOWLEDGEMENTS

This work is funded by the German Research Foundation (DFG) under grant No. BE 1373/3-3. The basic components of the semantic web services were developed in a student research project by Felix Reither and Julian Sawatzki and revised by Marcel Mischo.

## REFERENCES

Abele, E., Chryssolouris, G., Sihn, W., Metternich, J., ElMaraghy, H., Seliger, G., Sivard, G., ElMaraghy, W., Hummel, V., Tisch, M., and Seifermann, S. (2017). Learning factories for future oriented research and education in manufacturing. *CIRP Ann.*, 66(2):803–826.

Bordel Sánchez, B., Alcarria, R., Sánchez de Rivera, D., and Robles, T. (2018). Process execution in Cyber-Physical Systems using cloud and Cyber-Physical Internet services. *J. Supercomput.*, 74(8):4127–4169.

Boschert, S. and Rosen, R. (2016). Digital Twin—The Simulation Aspect. In *Mechatron. Futur.*, pages 59–74. Springer.

Broy, M., Cengarle, M. V., and Geisberger, E. (2012). Cyber-Physical Systems: Imminent Challenges. In *Large-Scale Complex IT Syst. Dev., Operat. and Manag. - 17th Monterey Workshop*, volume 7539 of *LNCS*, pages 1–28. Springer.

Burns, T., Cosgrove, J., and Doyle, F. (2019). A Review of Interoperability Standards for Industry 4.0. *Procedia Manuf.*, 38:646–653.

Calà, A., Ryashentseva, D., and Lüder, A. (2016). Modeling approach for a flexible manufacturing control system. In *21st Int. Conf. on Emerg. Technol. and Factory Automat.*, pages 1–4. IEEE.

Chen, L. and Yang, X. (2005). Applying AI Planning to Semantic Web Services for Workflow Generation. In *Int. Conf. on Semant., Knowl. and Grid*, page 65. IEEE.

Cheng, H., Xue, L., Wang, P., Zeng, P., and Yu, H. (2017). Ontology-based web service integration for flexible manufacturing systems. In *15th Int. Conf. on Ind. Inf.*, pages 351–356. IEEE.

Ciortea, A., Mayer, S., and Michahelles, F. (2018). Repurposing Manufacturing Lines on the Fly with Multi-agent Systems for the Web of Things. In *Proc. of the 17th Int. Conf. on Autonomous Agents and Multi-Agent Systems*, pages 813–822. Int. Found. for Autonomous Agents and Multiagent Systems / ACM.

Hepp, M., Leymann, F., Domingue, J., Wahler, A., and Fensel, D. (2005). Semantic business process management: a vision towards using semantic Web services for business process management. In *Int. Conf. on e-Business Eng.*, pages 535–540. IEEE.

Humm, B., Bense, H., Bock, J., Classen, M., Halvani, O., Herta, C., Hoppe, T., Juwig, O., and Siegel, M. (2020). Applying machine intelligence in practice. *Informatik Spektrum*.

Jammes, F. and Smit, H. (2005). Service-Oriented Paradigms in Industrial Automation. *IEEE Trans. Ind. Inf.*, 1(1):62–70.

Janiesch, C., Koschmider, A., Mecella, M., Weber, B., Burattin, A., Ciccio, C. D., Fortino, G., Gal, A., Kannengiesser, U., Leotta, F., Mannhardt, F., Marrella, A., Mendling, J., Oberweis, A., Reichert, M., Rinderle-Ma, S., Serral, E., Song, W., Su, J., Torres, V., Weidlich, M., Weske, M., and Zhang, L. (2020). The Internet-of-Things Meets Business Process Management. A Manifesto. *IEEE Syst. Man Cybern. Mag.*

- Janowicz, K., Haller, A., Cox, S. J., Le Phuoc, D., and Lefrançois, M. (2019). SOSA: A lightweight ontology for sensors, observations, samples, and actuators. *J. Web Semant.*, 56:1–10.
- Järvenpää, E., Siltala, N., Hylli, O., and Lanz, M. (2019). The development of an ontology for describing the capabilities of manufacturing resources. *J. Intell. Manuf.*, 30(2):959–978.
- Klein, P. and Bergmann, R. (2019). Generation of Complex Data for AI-Based Predictive Maintenance Research With a Physical Factory Model. In *16th Int. Conf. on Inform. in Control Automat. and Rob.*, pages 40–50. SciTePress.
- Klein, P., Malburg, L., and Bergmann, R. (2019). FTOnto: A Domain Ontology for a Fischertechnik Simulation Production Factory by Reusing Existing Ontologies. In *Proc. of the Conf. LWDA*, volume 2454, pages 253–264. CEUR-WS.org.
- Lamy, J.-B. (2017). Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artif. Intell. Med.*, 80:11–28.
- Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., and Hoffmann, M. (2014). Industry 4.0. *BISE*, 6(4):239–242.
- Lastra, J. L. M. and Delamer, I. M. (2006). Semantic Web Services in Factory Automation: Fundamental Insights and Research Roadmap. *IEEE Trans. Ind. Inf.*, 2(1):1–11.
- Lee, J., Kao, H.-A., and Yang, S. (2014). Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment. *Procedia CIRP*, 16:3–8.
- Lemaignan, S., Siadat, A., Dantan, J.-Y., and Semenenko, A. (2006). MASON: A Proposal For An Ontology Of Manufacturing Domain. In *Workshop on Distrib. Intell. Syst.: Collect. Intell. and Its Appl.*, pages 195–200. IEEE.
- Lobov, A., Lopez, F. U., Herrera, V. V., Puttonen, J., and Lastra, J. L. M. (2008). Semantic Web Services framework for manufacturing industries. In *Int. Conf. on Rob. and Biomim.*, pages 2104–2108. IEEE.
- Lu, Y. and Ju, F. (2017). Smart Manufacturing Systems based on Cyber-physical Manufacturing Services (CPMS). *IFAC-PapersOnLine*, 50(1):15883–15889.
- Marrella, A. (2018). Automated Planning for Business Process Management. *J. Data Semant.*
- Marrella, A. and Mecella, M. (2018). Cognitive Business Process Management for Adaptive Cyber-Physical Processes. In *Bus. Process Manag. Workshops*, LNBP, pages 429–439. Springer.
- Martin, D. L., Burstein, M. H., McDermott, D. V., McIlraith, S. A., Paolucci, M., Sycara, K. P., McGuinness, D. L., Sirin, E., and Srinivasan, N. (2007). Bringing Semantics to Web Services with OWL-S. *World Wide Web*, 10(3):243–277.
- McDermott, D. V., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). PDDL - The Planning Domain Definition Language: Technical Report CVC TR-98-003/DCS TR-1165.
- Minor, M., Montani, S., and Recio-García, J. A. (2014). Process-oriented Case-based Reasoning. *Inf. Syst.*, 40:103–105.
- Monostori, L. (2014). Cyber-physical Production Systems: Roots, Expectations and R&D Challenges. *Procedia CIRP*, 17:9–13.
- Müller, G. (2018). *Workflow Modeling Assistance by Case-based Reasoning*. Springer Fachmedien, Wiesbaden.
- Ocker, F., Kovalenko, I., Barton, K., Tilbury, D., and Vogel-Heuser, B. (2019). A Framework for Automatic Initialization of Multi-Agent Production Systems Using Semantic Web Technologies. *IEEE Rob. Autom. Lett.*, 4(4):4330–4337.
- Puttonen, J., Lobov, A., and Lastra, J. L. M. (2013). Semantics-Based Composition of Factory Automation Processes Encapsulated by Web Services. *IEEE TII*, 9(4):2349–2359.
- Puttonen, J., Lobov, A., Soto, M. A. C., and Lastra, J. L. M. (2010). A Semantic Web Services-based approach for production systems control. *Adv. Eng. Inf.*, 24(3):285–299.
- Puttonen, J., Lobov, A., Soto, M. A. C., and Lastra, J. L. M. (2019). Cloud computing as a facilitator for web service composition in factory automation. *J. Intell. Manuf.*, 30(2):687–700.
- Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., and Fensel, D. (2005). Web Service Modeling Ontology. *Appl. Ontol.*, 1(1):77–106.
- Sanchis, R., García-Perales, Ó., Fraile, F., and Poler, R. (2020). Low-Code as Enabler of Digital Transformation in Manufacturing Industry. *Appl. Sci.*, 10(1):12.
- Schnicke, F., Kuhn, T., and Antonino, P. O. (2020). Enabling Industry 4.0 Service-Oriented Architecture Through Digital Twins. In *Softw. Archit.*, volume 1269 of *Commun. Comput. Inf. Sci.*, pages 490–503. Springer.
- Seiger, R., Huber, S., and Schlegel, T. (2018). Toward an execution system for self-healing workflows in cyber-physical systems. *Softw. Syst. Model.*, 17(2):551–572.
- Sure, Y., Staab, S., and Studer, R. (2009). Ontology Engineering Methodology. In *Handb. on Ontologies*, Int. Handb. on Inf. Syst., pages 135–152. Springer.
- Zeyen, C., Malburg, L., and Bergmann, R. (2019). Adaptation of Scientific Workflows by Means of Process-Oriented Case-Based Reasoning. In *Case-Based Reason. Res. and Dev. - 27th Int. Conf.*, volume 11680 of *LNCS*, pages 388–403. Springer.