Transforming OPC UA Information Models into Domain-Specific Ontologies

Gernot Steindl
Institute of Computer Engineering
TU Wien
Vienna, Austria
gernot.steindl@tuwien.ac.at

Wolfgang Kastner
Institute of Computer Engineering
TU Wien
Vienna, Austria
wolfgang.kastner@tuwien.ac.at

2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS) | 978-1-7281-6207-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPS49255.2021.9468254

***Abstract*—Semantics interoperability is important for cyber-physical systems to enable complex data processing and facilitating interworking. OPC Unified Architecture (OPC UA) provides an extensible information model but lacks formal semantics. Ontologies based on the Web Ontology Language (OWL) can provide such formal semantics. Thus, we present a transformation approach that converts OPC UA information models (or only parts of them) into domain-specific ontologies. The transformation process consists of two steps. The first step is a mapping from OPC UA to OWL Full. The second step performs a graph transformation, based on SPARQL rules, into the domain-specified target ontology. The adaption of this transformation to the source information model and the target ontology can be accomplished by only adapting these transformation rules. The presented approach is evaluated for a use case of an industrial heating process to show its flexibility.**

***Index Terms*—information modeling, ontology, model transformation, OPC UA**

I. Introduction

Data play a key role in the ongoing transition to Industry 4.0, trying to reach the goals of operational efficiency and productivity and a higher level of automation [1]. Cyber-Physical Systems (CPSs) are applied in the industrial domain, which gather these data and process them. The semantics of data and context information is important to further improve the abilities of the systems based on the underlying CPSs. The OPC Unified Architecture (OPC UA) standard provides, next to its communication specification, a standardized information model, providing additional structure and semantics to the data. Thus, information about equipment and the plant's topology can be made available in OPC UA and later be used to set data into context for further processing.

Because of its capabilities, OPC UA is a suitable technology for Industry 4.0 applications. A disadvantage of the OPC UA information model is its lack of formal semantics, the missing browsing capability [2] and a limited semantic expressiveness compared to more advanced knowledge representations [3]. However, these features are important to develop more sophisticated CPSs, like presented in [4], which are able to get an in-depth knowledge of the monitored system or make them self-configurable and self-adaptive.

This research was supported by the doctoral school SIC! - Smart Industrial Concept! at the TU Wien.

A more formal, machine-readable knowledge representation is provided by the Semantic Web with its technologies, like Resource Description Framework (RDF), Web Ontology Language (OWL) and SPARQL Protocol and RDF Query Language (SPARQL). Thus, transformations between the OPC UA information model and OWL have been proposed to use various features of OWL [5], [3], [6]. However, a simple OWL representation of the OPC UA information model is not always appropriate. For some applications, the information from the OPC UA model or only parts of it shall be used to instantiate a domain-specific ontology. Thus, this paper provides an answer to the question, how domain-specific ontologies can be instantiated based on already available OPC UA information models, considering changes of the source information model and the target ontology.

The remainder of the paper is structured as follows: Sec. II gives a short overview of related work in the area of OPC UA in combination with the Semantic Web. Sec. III explains the proposed process of transforming OPC UA information models into domain-specific ontologies. In Sec. IV, a proof of concept implementation is presented to evaluate the proposed transformation process for a use case of a thermal heating process. Finally, the proposed transformation process is discussed, and an outlook on our future work is given.

II. Related Work

Ontologies are used for knowledge representation and can be defined as a formal, explicit specification of a shared conceptualization [7]. Technologies and standards form the so-called Semantic Web Stack like RDF [8], Resource Description Framework Schema (RDFS) [9], OWL [10], and SPARQL [11] support the development of ontologies. RDF defines triples to formulate statements, consisting of a subject, a predicate, and an object. RDFS introduces new concepts based on RDF to increase its semantic expressiveness. OWL further enhances this expressiveness by introducing concepts of formal logic. There exists different OWL levels, like OWL Full and OWL DL with its profiles EL, RL, QL. It is important to know that OWL Full has the least restrictions for modeling, which leads to undecidability when it comes to reasoning. More details can be found in [12].

978-1-7281-6207-2/21/$31.00 ©2021 IEEE

OPC UA is an industrial communication standard that aims to solve interoperability problems at the transport and semantic layer in Industry 4.0 applications. Thus, next to the data communication, OPC UA facilitates the semantic description of data by defining an OPC UA information model. Everything can be described with this information model, from simple devices to very complex components in an object-oriented and semantically meaningful way. Therefore, OPC UA defines an address space model, which is the meta-model of the information model [13]. The address space model uses *Nodes* and *References* to form a graph structure. Based on this meta-model, the OPC UA information model defines the address space of the OPC UA server [14]. This address space is the information that is exposed by a specific OPC UA server. A detailed description of these concepts and their application can be found in [15].

A formal mapping between OPC UA and OWL DL is presented in [5]. It is shown, why a trivial mapping between OPC UA and OWL DL is not possible. One major issue is the usage of *Instance-Nodes* at the type definitions in OPC UA, which is not allowed in OWL DL. Thus, mappings have to be defined for such OPC UA design patterns, which can be specified in various ways, based on certain design choices. Another mapping between OPC UA and OWL DL has been presented in [3]. They identified axioms and properties which are available in OWL but not in OPC UA. Thus, they conclude that OWL is semantically more expressive than OPC UA and also specified a mapping between them.

The mapping problem can be partly avoided by using OWL Full instead of OWL DL, which allows one-to-one mappings for most of the OPC UA concepts, as shown in [6]. But such an approach is only applicable in cases where reasoning is not required.

However, all these approaches only allow a direct translation of the used OPC UA information model into an OWL representation, whereas our proposed approach also makes it possible to define mapping rules to extract the needed information from the OPC UA model and use it in a domain-specific target ontology. This makes our approach even more flexible.

III. Transformation Process

The proposed approach targets the situation where the information model is already deployed at a running OPC UA server, and the address space can be read by an OPC UA client.

For the model transformations, different levels of abstraction are of interest. A three-tier architecture with M2 as meta-model, M1 as model, and M0 as instance level can be used to describe these levels of abstraction, as shown in Fig. 1. The OPC UA address space model is located on Level M2 as the meta-model of the OPC UA information model [16]. For the ontology, the ontology definition is located on level M1 and the used ontology language, in this case OWL, on M2 [17].

The transformation of the OPC UA information model into a domain-specific ontology is done in two steps. The first step is transforming the OPC UA information model into OWL Full. Using OWL Full makes reasoning impossible, but the mapping is straight forward, by a one-to-one mapping of most concepts. In the second step, the OWL Full graph is transformed with the help of SPARQL rules into a domain-specific ontology. This domain-specific ontology can be based on OWL DL, which re-enables reasoning again.

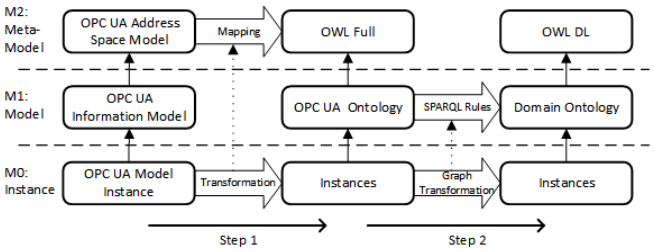

Fig. 1. Model transformation on different levels of abstraction.

The second transformation step is performed at a lower level of abstraction, namely on M1. SPARQL rules are used to define how concepts are transformed from one ontology into the other. Therefore, SPARQL construct clauses are applied to search for patterns in the ontology graph and create new concepts and instances based on these patterns. To adapt this approach to other domain ontologies or the OPC UA information models, only these SPARQL rules have to be changed. This leads to a flexible transformation approach.

A more detailed description of the whole transformation process, as well as the involved components, is depicted in Fig. 2 and described in the following:

1) The OPC UA information model is designed, and an XML-Nodeset file is created.
2) The information model is instantiated in an OPC UA server, which exposes the information via its address space.
3) The whole OPC UA address space is read by a client from the server and a OWL Full representation is generated, based on defined mapping rules.
4) The OWL Full representation is stored in a so-called ”Quad Store”. A quad store is a specific database for RDF triples (triple store), which additionally provides the feature of named graphs.
5) The SPARQL transformation rules are defined. These rules specify the transformation between the OWL Full representation of the OPC UA information model and the domain-specific target ontology.
6) The specified SPARQL rules are executed at the SPARQL endpoint of the quad store. These rules create a new ontology stored in a new named graph to separate it from the source ontology.
7) The created ontology can now be accessed. New information can be inferred by reasoners and retrieved with SPARQL queries.

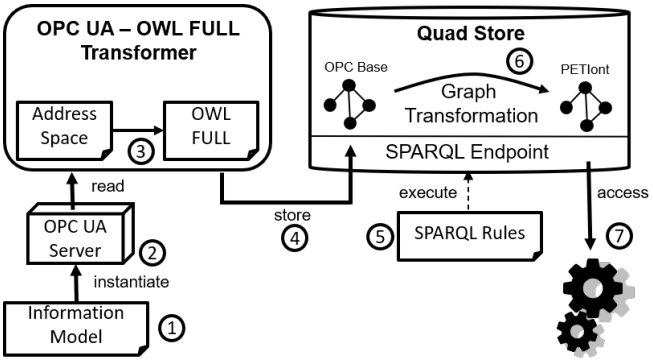

Fig. 2. Model transformation process steps and involved components.

IV. Use Case - Heating Process

To evaluate our proposed transformation approach, we selected a simple heating process as use case. The pipe and instrumentation (P&I) diagram of the heating process is shown in Fig. 3. The main components are an electric heater ”H1”, two fans ”F1” and ”F2”, a heat exchanger ”HE1”, and an open vessel named ”SiPro”. Various temperature sensors (TI) and two flow sensors (FI) are installed at the inlet and outlet pipes. The fans can only be switched to discrete speeds, while the heater's power can be controlled continuously. Fresh air is heated by the electric heater ”H1” and ventilated through the inlet pipe to the vessel. The heater is controlled by the temperature of the chamber, which is measured at the chamber outlet. The exhaust air of the process is used by the heat exchanger ”HE1” for heat recovery.

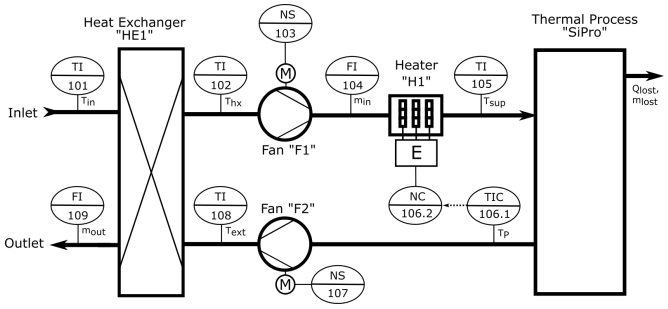

Fig. 3. P&I diagram of the heating process.

A. OPC UA Information Model

For the described use case, shown in Fig. 3, an OPC UA information model is designed and initiated in an OPC UA server. The information model captures the structure of the heating process, including its components, their topology, sensors, and actuators. The information model builds on existing industrial standards. To structure the plant, elements defined by ISO 10209 [18] are used. The plant equipment is named and structured based on EN ISO 10628 [19], which defines equipment for P&I diagrams, such as heat exchangers, blower fans, vessels, etc. A small excerpt from the resulting type hierarchy in the OPC UA information model is depicted in Fig. 4, limited to the exchanger type. The complete information model is available on GitHub[1].

As shown in Fig. 4, it can be specified whether the pressure or temperature change for a certain equipment should be neglected or not by two variables, called *NeglectPressureChange* and *NeglectTemperatureChange*. This information can be used later on for sensor data evaluation, which is out of scope for this paper. Nevertheless, this information is also used in the transformation step. Next to these variables, the *HeatExchangerType* defines additional objects for flow ports and methods for controlling the equipment.

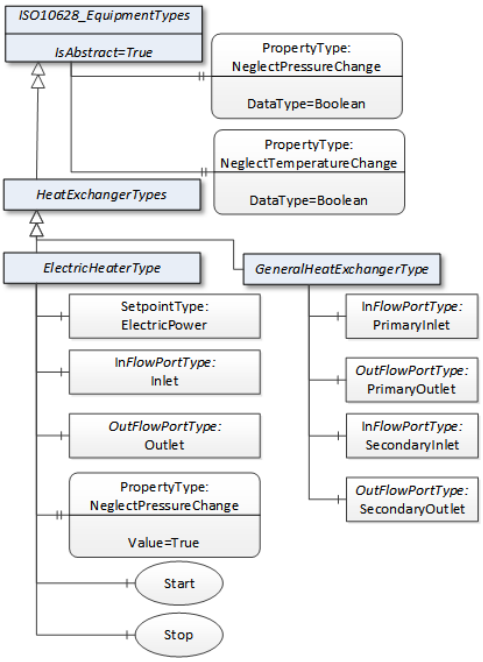

Fig. 4. OPC UA information model - excerpt of the specified equipment.

The standard IEC 62424 [20] defines ”process control engineering (PCE) requests”, representing sensors, actuators, and control functions. These elements are used in the OPC UA information model, as depicted in Fig. 5. The designation of a PCE request consists of its category and its function. Also, a unique reference number is assigned. A standardized character string specifies the category and function. For example, the PCE request with the reference number 106.1 in Fig. 3 has the designation ”TIC”, where category ”T” means temperature, the letter ”I” stands for indication and ”C” for a control function. More Information about the application of IEC 62424 can be found in [21].

Non-hierarchical references are also specified and used to describe the topology of the pant. They are shown in Fig. 6. Equipment can be connected via a mass transport (*hasMassFlowTo*) or via an information flow (*hasSignalTo*). The *hasPCE_Request* and *hasProcessConnection* references

[1]https://github.com/Smart-Industrial-Concept/HeatingProcess_OPC_UA_Model

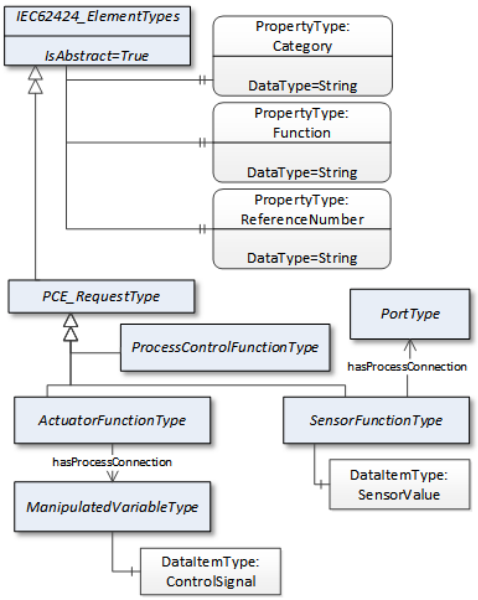

Fig. 5. OPC UA information model - PCE request type

are used to connect sensors and actuators with their corresponding plant equipment.

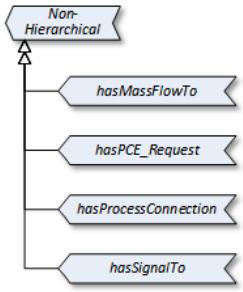

Fig. 6. OPC UA information model - reference hierarchy

For the use case of the thermal heating process, depicted in Fig. 3, an information model is loaded into an OPC UA server. This model instance is partly shown in Fig. 7. The server exposes its address space, where a client can access it to perform the transformation process.

B. Domain Ontology - PETIont

As target ontology for the presented proof of concept, the Pipe, Equipment, Topology, and Instrumentation Ontology (PETIont) is used. It has similar concepts as the OPC UA information model as it is partly based on the same standards, like IEC 62424 with its PCE requests. The ontology is implemented in OWL DL, and its main class concepts and relations are depicted in Fig. 8. The ontology distinguishes between hydraulic and thermal equipment. Thermal equipment changes the temperature between its input port and output port, whereas hydraulic equipment changes the pressure. The complete ontology can be found on GitHub[2].

[2]https://github.com/Smart-Industrial-Concept/PETIont

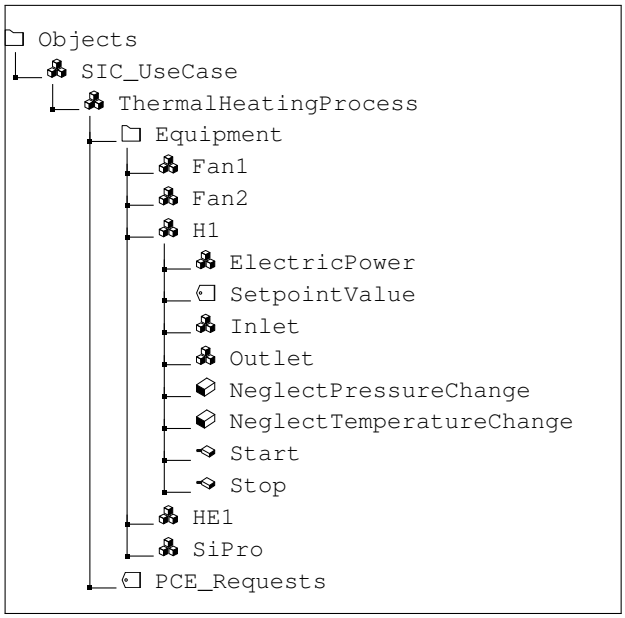

Fig. 7. Instantiating the OPC UA information model for the simple thermal heating process.

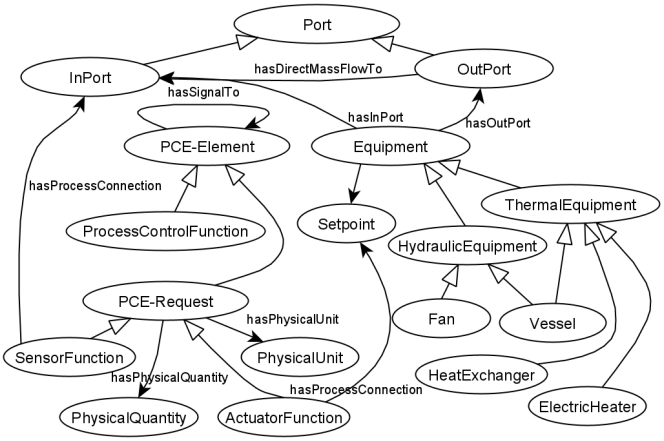

Fig. 8. PETIont - main concepts and their relations.

C. Transformation Rules

The first step of the transformation process is reading the address space form the OPC UA server and creating the OWL Full representation of it. Basically, an OPC UA client connects to the server and extracts the address space from it. The mapping creates an *owl:Class* for the OPC UA node classes *ObjectType*, *VariableType* and *DataType*. Nodes of node class *ReferenceType* are mapped to *owl:ObjectProperty*. A more detailed description of all these transformation rules can be found in [6].

The second step in the transformation process is performing a graph transformation on the created OWL Full representation of the OPC UA information model. This is achieved with the help of the SPARQL construct feature, which creates RDF triples based on a SPARQL query. As the quad store supports named graphs, the results are stored in a new graph inside

the quad store. Thus, the new graph can be built step-wise by executing rules which search for certain patterns inside the source graph. For our proof of concept, we used Apache Jena Fuseki version 3.17.0 as the quad store.

For the transformation in our use case between the presented OPC UA information model and PETIont, we specify nine different SPARQL rules. To make the rules more understandable, not all created relations between the concepts are explicitly mentioned in the following rule description. The transformation output is indicated with a "→":

1) Equipment where the pressure change is not neglected → *peti:HydraulicEquipment*
2) Equipment where the temperature change is not neglected → *peti:ThermalEquipment*
3) A *hasMassFlowto* reference between two ports belonging to different components → *peti:hasDirectMassFlowTo*
4) A *hasMassFlowTo* reference between two ports belonging to the same component → *peti:hasInternalMassFlowTo*
5) *ActuatorFunctionType* → *peti:PCEActuatorFunction*
6) *SensorFunctionType* → *peti:PCESensorFunction*
7) *SensorFunctionType* including a control function (letter "C") → *peti:ProcessControlFunction* and a *peti:Setpoint*
8) A "HasProcessConnection" between a PCE request and a component → *peti:hasProcessConnection*
9) A *HasSignalto* reference between two PCE requests → *peti:hasSignalTo*

To give an impression how these rules are implemented in SPARQL, Listing 1 shows rule number 3 as an example. This rule is chosen because of its traceability to understand the general principles without being too complex.

Line 1 to 3 are just defining abbreviations for the namespaces. The "WHERE" clause is used to find patterns inside the OWL representation of the OPC UA information model. In the first block (lines 14 to 18), all *hasMassFlowTo* connections between an outport and an inport are retrieved. In the second block (lines 20 to 26), the associated components' names are queried. These names are used to define the URIs used in the target ontology, stored in the variables *?startPortURI* and *?endPortURI*. This definition happens in the last block, starting in line 28. In the "INSERT" clause, the object property *peti:hasDirectMassFlowto* and its inverse property *peti:hasDirectMassFlowto* are instantiated in the target ontology (lines 9 and 10). The resulting target ontology is created in a named graph, specified in line 8, which separates it from the source graph.

```
1  PREFIX uaBase: <http://auto.tuwien.ac.at/~ontologies/opcua#>
2  PREFIX : <http://auto.tuwien.ac.at/~ontologies/useCaseOPCUA#>
3  PREFIX peti: <http://auto.tuwien.ac.at/sic/PETIont#>
4
5  INSERT{
6   #create instances of the mass flow connection between the
7   #ports of connected component
8   GRAPH <http://auto.tuwien.ac.at/petiGraph> {
9    ?startPortURI peti:hasDirectMassFlowTo ?endPortURI.
10   ?endPortURI peti:hasDirectMassFlowFrom ?startPortURI.
11   }
12 }
13 WHERE{
14   #retrieve the mass flow connection between a outport of a component
15   #and the inport of another component
16   ?startPort :hasMassFlowTo ?endPort.
17   ?startPort a :outFlowPortType.
18   ?endPort a :inFlowPortType.
19
20   #retrieve the names of the connected components
21   ?startPort uaBase:BrowseName ?startPortName.
22   ?endPort uaBase:BrowseName ?endPortName.
23   ?startPort uaBase:ComponentOf ?startComp.
24   ?startComp uaBase:BrowseName ?startCompName.
25   ?endPort uaBase:ComponentOf ?endComp.
26   ?endComp uaBase:BrowseName ?endCompName.
27
28   #create the URIs of the instances in the target ontology
29   BIND("http://auto.tuwien.ac.at/sic/PETIont#" as ?petiURI)
30   BIND(uri(?petiURI + str(?startCompName) +"_" + str(?startPortName))
       as ?startPortURI)
31   BIND(uri(?petiURI + str(?endCompName) +"_" + str(?endPortName)) as
       ?endPortURI)
32 }
```

Listing 1. SPARQL rule 3 - Transforming a mass flow connecting two different components.

D. Information Retrieval

After the transformation process is performed, the information is accessible via the SPARQL endpoint of the Fuseki Server. As the target ontology used OWL DL, reasoning can also be applied if necessary. The full potential of the approach will be achieved if the created domain-specific ontology will be interlinked with other domain ontologies to build a large knowledge graph. This knowledge graph would be accessible for various applications or services within Cyber-Physical Systems (CPSs).

To prove that our transformation was successful, a simple example is given in Listing 2. It shows how information can be retrieved from the instantiated target ontology with the help of SPARQL. Therefore, the plant topology information, represented in the domain-specific ontology, is accessed. The SPARQL query retrieves the installed equipment. Additionally, directly connected components located before and after the equipment are also retrieved. The answer to that query is also depicted at the end of Listing 2. To reduce the query's complexity, property paths are used in lines 8 and 11, supported in SPARQL version 1.1.

```
1  PREFIX : <http://auto.tuwien.ac.at/sic/PETIont#>
2
3  SELECT *
4  WHERE {
5    ?equipment a :Equipment
6
7    optional{
8     ?equipment :hasOutPort/:hasDirectMassFlowTo/^:hasInPort ?nextComp
9    }
10   optional{
11    ?equipment :hasInPort/^:hasDirectMassFlowTo/^:hasOutPort ?prevComp
12   }
13 }
```

?equipment	?nextComp	?prevComp
:H1	:SiPro	:Fan1
:Fan2	:HE1	:SiPro
:HE1	:Fan1	:Fan2
:Fan1	:H1	:HE1
:SiPro	:Fan2	:H1

Listing 2. SPARQL query to retrieve the topology information with its corresponding result.

V. Discussion and Future Work

We evaluated the proposed approach for transforming OPC UA information models into domain-specific ontologies with a proof of concept. The adaption of the transformation process to other OPC UA information models or domain ontologies can be achieved by only adapting the SPARQL rules. The remaining steps in the process are executed automatically, which makes it flexible to an adaption of the source model or target ontology. An additional benefit of our approach is that it is also applicable for already existing OPC UA servers, as the information model is directly retrieved from the address space of a running OPC UA server.

We also briefly introduced the information retrieval by applying a SPARQL query to the endpoint. Having all this information about the plant topology, the available equipment, and instrumentation in a machine-readable, formal representation is very beneficial for a broad spectrum of applications or services. It can be used to facilitate the monitoring, diagnosis, prediction, and control of a plant. The interlinking of this information with other domain-specific ontologies will enable the proposed transformation approach's full potential.

A pitfall of applying SPARQL rules for the graph transformation could be, that dependencies between rules can easily be introduced. These dependencies would require a certain execution order for the transformation. This should be avoided by design or clearly documented if it cannot be avoided.

In the future, we want to use the available topology information from the OPC UA information model to evaluate sensor data in CPS, based on the information in the created domain-specific ontology and other interlinked domain ontologies, capturing further knowledge about the plant. Various services can make use of and contribute to such an interlinked, shared knowledge graph, which will enable new capabilities within CPSs.

References

[1] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *Journal of Industrial Information Integration*, vol. 6, pp. 1–10, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.jii.2017.04.005

[2] R. Schiekofer and M. Weyrich, "Querying OPC UA information models with SPARQL," *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, vol. 2019-September, pp. 208–215, 2019.

[3] J. Bakakeu, M. Brossog, J. Zeitler, J. Franke, S. Tolksdorf, H. Klos, and J. Peschke, "Automated reasoning and knowledge inference on OPC UA information models," *Proceedings - 2019 IEEE International Conference on Industrial Cyber Physical Systems, ICPS 2019*, pp. 53–60, 2019.

[4] J. Lee, B. Bagheri, and H. A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.mfglet.2014.12.001

[5] R. Schiekofer, S. Grimm, M. M. Brandt, and M. Weyrich, "A formal mapping between OPC UA and the semantic web," *IEEE International Conference on Industrial Informatics (INDIN)*, vol. 2019-July, pp. 33–40, 2019.

[6] G. Steindl, T. Früwirth, and W. Kastner, "Ontology-Based OPC UA Data Access via Custom Property Functions," in *24th Interantional Conference on Emerging Technologies and Factory Automation*, Zaragoza, Spain, 2019.

[7] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge Engineering : Principles and methods," *Data & Knowledge Engineering*, vol. 25, pp. 161–197, 1998.

[8] G. Klyne and J. J. Carroll, "Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation," World Wide Web Consortium, Tech. Rep., 2004. [Online]. Available: https://www.w3.org/TR/rdf-concepts/

[9] D. Brickley and R. Guha, "RDF Schema 1.1. W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., 2014. [Online]. Available: https://www.w3.org/TR/rdf-schema/

[10] W3C OWL Working Group, "OWL 2 Web Ontology Language. Document Overview," World Wide Web Consortium (W3C), Tech. Rep., 2012. [Online]. Available: https://www.w3.org/TR/owl2-overview

[11] S. Harris and A. Seaborne, "SPARQL 1.1 Query Language. W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., 2013. [Online]. Available: https://www.w3.org/TR/sparql11-query/

[12] M. Krötzsch, *OWL 2 Profiles: An Introduction to Lightweight Ontology Languages*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 112–183.

[13] OPC Foundation, "OPC Unified Architecture Specification. Part 3: Address Space Model. Release 1.04," Standard, 2017.

[14] OPC Foundation, "OPC Unified Architecture Specification. Part 5: Information Model. Release 1.04," Standard, 2017.

[15] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. [Online]. Available: http://link.springer.com/10.1007/978-3-540-68899-0

[16] B. Lee, D. K. Kim, H. Yang, and S. Oh, "Model transformation between OPC UA and UML," *Computer Standards and Interfaces*, vol. 50, pp. 236–250, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.csi.2016.09.004

[17] K. H. Van Dam and J. Keirstead, "Re-use of an ontology for modelling urban energy systems," *3rd International Conference on Next Generation Infrastructure Systems for Eco-Cities, INFRA 2010 - Conference Proceedings*, 2010.

[18] EN ISO 10209:2012, "Technical product documentation - Vocabulary - Terms relating to technical drawings, product definition and related documentation," Tech. Rep., 2012.

[19] EN ISO 10628, "Diagrams for the chemical and petrochemical industry," European Standards, Tech. Rep., 2012.

[20] I. 62424, "Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools," International Electrotechnical Commission, Tech. Rep., 2016.

[21] T. Bindel and D. Hofmann, *R&I-Fließschema*, ser. essentials. Wiesbaden: Springer Fachmedien Wiesbaden, 2016. [Online]. Available: http://link.springer.com/10.1007/978-3-658-15559-9