Utilizing Ontological Classification Systems and Reasoning for Cyber-Physical Systems

Stefan Zander[a], Yingbing Hua[b]

[a] *FZI Research Center for Information Technology*

[b] *Karlsruhe Institute of Technology*

Abstract

This work presents a novel ontology-based approach for the complementation of technical specifications of cyber-physical system components using ontological classification and reasoning. We build on the AutomationML standard and outline how data represented with it can be transformed into an RDF instance graph. We exemplarily show how complementary information about a component's functionality, its operation environment and purpose can be inferred through a combination of automatic classification with the linkage of different domain classification systems. The general applicability of the presented approach is demonstrated by a concrete use case from the ReApp project.

Keywords: Industry 4.0, Knowledge Representation, Cyber-Physical Systems, Semantic Web Technologies, AI Reasoning Methods

1. Introduction

In this work, we concisely present a novel ontology-based approach for complementing technical specifications of hardware components found in cyber-physical systems (CPS) using ontological classification and reasoning. The main feature that distinguishes this approach from most related ontology-based knowledge representation approaches in robotics and CPS is that we exclusively describe such information in the terminological part (*aka TBox*), i.e., the schema part of ontologies in order to fully exploit the formal, model-theoretic semantics of the underlying ontology language and the logical entailments that can be computed from it by a reasoner (Rudolph, 2011; Krötzsch et al., 2014; Zander and Awad, 2015). The deduced terminological knowledge can then be used to

Email addresses: zander@fzi.de (Stefan Zander), yingbing.hua@kit.edu (Yingbing Hua)

complement the description of CPS components with information that is not explicitly asserted in their original technical specifications. We specifically use Description Logics (DL) as knowledge representation framework since they provide well-understood reasoning complexity and tractability (Baader et al., 2003; Gil, 2005; Krötzsch et al., 2014). As a consequence, the semantics upon which complementary information is deduced can be shared, extended, or adopted by other CPSs in order to reason about component descriptions.

2. Expressing Technical Specifications using AutomationML

AutomationML[1] (AML) is a well-known and fast growing standard that already caught the attention of Industry 4.0 communities (RAM, 2015). It covers engineering aspects including topology, geometry, kinematics, logic and communication (Drath et al., 2008) that can be used for describing properties and functionalities of a CPS component. Data contained in the AML description of one component can be exposed to the communication network of a CPS system and consumed by other components (Schleipen et al., 2014). From a modelling perspective, AutomationML is based on the object-oriented paradigm and supports fundamental techniques like class-instance relationship and inheritance hierarchy. The architecture of AutomationML contains the following blocks:

- RoleClass library: Domain-specific concepts are modelled as RoleClasses and organized as taxonomy in the RoleClass library. AutomationML provides a predefined set of RoleClass libraries for the domain of automation and manufacturing that covers abstract concepts like product, process, and resource, and specific components like robot or conveyor. These can be extended to cover individual requirements of the application domain.
- InterfaceClass library: InterfaceClasses define relations between objects. AutomationML predefines some abstract interfaces for general automation systems, that can be further extended by user.
- SystemUnitClass library: Component models in the production system can be modelled as SystemUnitClasses. These are specific user-defined AML classes and typically manufacturer-dependent. A SystemUnitClass can reference multiple RoleClasses by means of SupportedRoleClass to demonstrate its semantics.
- InstanceHierarchy: Concret instances of component models can be stored in the InstanceHierarchy. These instances carry project data and represent the real plant setup in the digital world.

However, none of these elements exhibit an explicit machine-readable formalization of its semantics. To accurately interpret data, special software with

[1] www.automationml.org

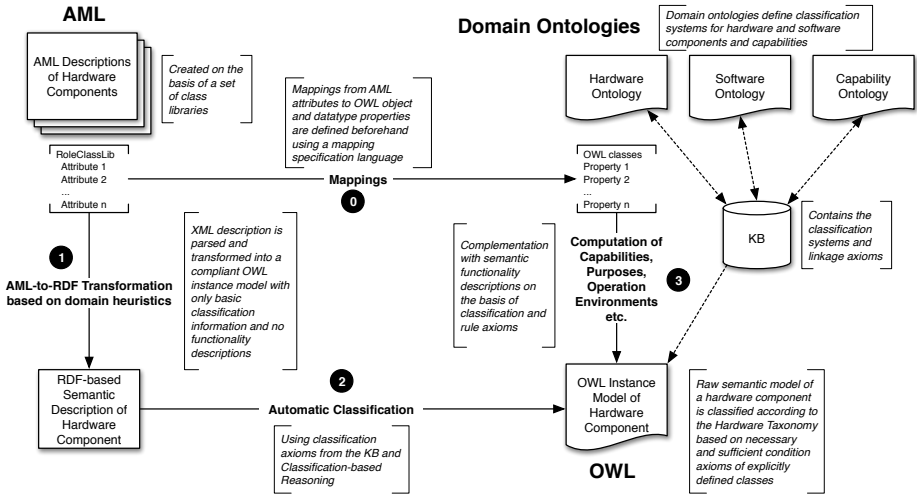

Figure 1: Overview of the presented approach showing uplifting, classification, and complementation phases for hardware component descriptions

hard-coded knowledge and application logic has to be implemented. Developers are assumed to have the correct understanding of the elements' semantics, which is often error-prone. Although AutomationML carries basic semantic information of a SystemUnitClass or its instance (in the InstanceHierarchy), the data exchange process is merely a serialization and deserialization of XML files that enables syntactic interoperability only (Biffl et al., 2014).

3. Approach

The presented approach presupposes the existence of an AML description of a hardware component.An AML description is analysed and transformed, i.e., *uplifted* into a compliant RDF description using transformation rules and domain heuristics (Section 3.1). The attributes contained in the generated RDF instance graph are then processed by a DL reasoner for automatic classification according to types defined in domain ontologies (Section 3.3). This is important as it builds the basis for inferring additional terminological knowledge by linking classification systems of different domain ontologies (Section 3.2). The inferred knowledge can the be added to the uplifted RDF instance model to provide a more complete and expressive representation (see Figure 1 for an overview).

3.1. Transforming AML Descriptions into RDF Graphs

Some initiatives already aimed at complementing AML with ontological semantics: Persson et al. (2010) introduced the transformation of AML documents

into RDF triple stores which can be queried with SPARQL. Kovalenko et al. (2015) tried to lift AML data into an ontology by directly converting the underlying XML schema of AML into the Web Ontology Language (OWL). A domain ontology was adopted in Björkelund et al. (2011) to extend semantic expressivity of AML for robotics. In contrast to these works, we use ontological reasoning in order to complement uplifted AML descriptions with additional semantics. We focus on SystemUnitClasses, because they represent reusable models of CPS components that contain richer information of functionalities. The following rules are applied to extract the implicit semantics of a SystemUnitClass into explicit form in RDF.

1. A SystemUnitClass will be transformed into one RDF graph.
2. Each SupportedRoleClass corresponds to a RDF TBox concept in the domain ontology. For each of them in the SystemUnitClass, we assign a TBox concept to the ABox instance.
3. We defined a direct mapping between all attributes and the properties defined in the ReApp domain ontologies.

Once all AML-specific notations are transformed, i.e., uplifted into an RDF graph that uses the terms defined in the domain ontologies to equivalently represent the AML classes, attributes, and values, it can be used for automated classification and the computation of implicitly contained knowledge.

3.2. Linking Classification Systems for Expressing Features of Components

In the ReApp-project[2] (Zander et al., 2015), we developed three different kinds of ontologies for encoding different types of domain knowledge. The *hardware* and *software ontologies* provide classification systems for technical components whereas the *capabilities ontology* defines a classification system for functionalities hard- and software components are able to perform. These different kinds of ontologies are linked together via a *base ontology* that defines the basic terms used to express the set of capabilities a certain hard- or software type exhibits per default.

As demonstrated in Zander and Awad (2015), capability information can be axiomatically linked to component classification systems via *role restriction axioms* to assert that a given capability (and all the capabilities it is subsumed by) is the default capability for a given classification type. Role restriction axioms in general interlink roles, concepts, and quantifiers[3] (Rudolph, 2011; Krötzsch et al., 2014) by forming an anonymous super class that contains all the individuals that satisfy the given restriction (Horridge et al., 2004). The following axioms exemplarily demonstrate how a hardware classification for SafetyLaserScanner

[2] http://www.reapp-projekt.de

[3] More expressive DLs even allow for the specification of multiplicity constraints

can be linked to a capability classification SafeMonitoringOf2DFields that acts as its default via a role restriction along the property hasCapability (Axiom 1) and what additional capabilities can be inferred via subsumption reasoning (Axiom 2 and 3):

$$\mathsf{SafetyLaserScanner} \sqsubseteq \exists \mathsf{hasCapability}.\mathsf{SafeMonitoringOf2DFields} \quad (1)$$

$$\mathsf{SafetyLaserScanner} \sqsubseteq \mathsf{LaserScanner} \quad (2)$$

$$\mathsf{LaserScanner} \sqsubseteq \exists \mathsf{hasCapability}.\mathsf{MonitoringOf2DFields} \quad (3)$$

$$\mathsf{MonitoringOf2DFields} \sqsupseteq \mathsf{SafeMonitoringOf2DFields} \quad (4)$$

With Axiom 2 the reasoner can deduce that a SafetyLaserScanner also has the default capability MonitoringOf2DFields and materialize this information for classified components. Once different domain classification systems are linked together, certain conditions can be defined that need to be satisfied by an individual component in order to be classified accordingly.

3.3. Defining Conditions for Automated Classification

By specifying *necessary* and *sufficient conditions* for class membership, a reasoner is capable to automatically classify random instances on the basis of the relationships they participate in (Horridge et al., 2004). This enables the automated classification of uplifted RDF component specifications according to the attributes extracted from its AML representation (see Figure 1). The formal interpretation of which is that the given conditions are not only necessary for determining class membership, they are also sufficient in a way that any random individual that satisfies these conditions becomes member of a class. The following axioms allow a reasoner to classify an uplifted component as 7AxisRobotArm, iff it is a RobotArm and participates in a numberOfAxis relationship, the value of which must be 7, i.e., iff it has exactly seven rotational axes:

$$\mathsf{7AxisRobotArm} \equiv \mathsf{RobotArm} \sqcap \exists \mathsf{numberOfAxis}.(=, 7) \sqcap (\leqslant 1\ \mathsf{numberOfAxis}) \quad (5)$$

$$\mathsf{7AxisRobotArm} \sqsubseteq \mathsf{RobotArm} \quad (6)$$

$$\mathsf{RobotArm} \sqsubseteq \exists \mathsf{hasCapability}.\mathsf{ReachPose} \quad (7)$$

$$\mathsf{7AxisRobotArm} \sqsubseteq \exists \mathsf{hasCapability}.\mathsf{FlexibleConfiguration} \quad (8)$$

When a component is classified as 7AxisRobotArm, the reasoner can infer that it offers the two default capabilities ReachPose and FlexibleConfiguration. Formally, this means that the class 7AxisRobotArm is subsumed by the two complex class expressions ∃hasCapability.ReachPose and ∃hasCapability.FlexibleConfiguration.

These additional subsumption relations can then be used for complementing a component's specification by *materializing* (see Domingue et al. (2011)) all the implicit knowledge inferred by a reasoner and add it to the component's instance model. The materialization is important as it allows all the inferred knowledge

about a component to be indexed by a triple store and retrieved through RDF query languages such as SPARQL (SPARQL, 2013). For the materialization of capability information, we employ the concept of *DL nominals* (cf. Baader et al. (2003); Rudolph (2011); Krötzsch et al. (2014)) to use them both in ABox and TBox axioms and ensure their *singularity*.

In the last part, we demonstrate how this axiomatic knowledge can be used for complementing the technical AML specification of a Sick S30B laser scanner. The following code excerpt shows the uplifted RDF representation[4]:

```
<urn:uuid:f81d4fae-7dec-11d0-765-00a0c91e6bf6>
  :hasManufaturer "Sick" ; :hasModelName "S30B-2011GA" ;
  :startAngle "135"^^xsd:integer ; :endAngle "135"^^xsd:Integer ;
  :maxMeasurementRangeInMeter "40"^^xsd:integer ;
  :maxProtectiveFieldRangeInMM "2000"^^xsd:integer ;
  :maxWarningFieldRange "8000"^^xsd:integer ;
  :maxSimultaneousFieldEvaluations "0"^^xsd:integer .
```

By employing automatic classification in combination with reasoning on the terminological part of domain ontologies, the RDF representation can be complemented by the following inferred and materialized information:

```
<urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6>
  rdf:type :2DSaftyLaserSensor , :LaserSensor , :Hardware Component ;
  :hasCapability :{SafeMonitoringOf2DFields} , :{MonitoringOf2DFields} , :{Monitoring} ;
  :hasPurpose :{HazardousAreaProtection} , :{AccessProtection} , :{PersonnelSafety} ;
  :hasOperationEvironment :iIndoor, iOutdoor .
```

The annotation model now contains information about its concrete type, capabilities, purposes and the operation environments in which it can be used.

4. Conclusion

In this work, we presented an approach for complementing the technical specifications of CPS' components by utilizing the formal, model-theoretic semantics encoded in DL ontologies using reasoning. We introduced AML as an XML-based technical specification language that facilitates data exchange between CPSs but has only limited semantic expressivity. We outlined how different classification systems can be linked together to express features and other relevant characteristics, how conditions can be defined to enable an automated classification of uplifted RDF component descriptions, and how implicitly inferred information can be used to complement component descriptions. In use cases from the ReApp project, we could demonstrate that the technical AML specifications can be complemented in useful ways to enable a formal verification and validation of component orchestrations, to foster reusability, and to compute advanced capabilities for compound components.

[4]We omitted namespaces for reasons of readability and comprehensibility.

References

Sebastian Rudolph. Foundations of description logics. In *Reasoning Web. Semantic Technologies for the Web of Data – 7th International Summer School 2011*, volume 6848 of *LNCS*, pages 76–136. Springer, 2011.

Markus Krötzsch, František Simančík, and Ian Horrocks. Description logics. *IEEE Intelligent Systems*, 29:12–19, 2014.

Stefan Zander and Ramez Awad. Expressing and reasoning on features of robot-centric workplaces using ontological semantics. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. to be published, 2015.

Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003. ISBN 0521781760.

Yolanda Gil. Description logics and planning. *AI Magazine*, 26(2):73–84, 2005. URL `http://dblp.uni-trier.de/db/journals/aim/aim26.html#Gil05`.

Reference architecture model industrie 4.0. Technical report, ZVEI, 2015. URL `http://www.zvei.org/en/subjects/Industry-40/Pages/The-Reference-Architectural-Model-RAMI-40-and-the-Industrie-40-Component.aspx`.

R. Drath, A. Luder, J. Peschke, and L. Hundt. Automationml - the glue for seamless automation engineering. In *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*, pages 616–623, Sept 2008. doi: 10.1109/ETFA.2008.4638461.

M. Schleipen, J. Pfrommer, K. Aleksandrov, D. Stogl, S. Escaida, J. Beyerer, and B. Hein. Automationml to describe skills of production plants based on the ppr concept. In *3rd AutomationML user conference*, 2014.

S. Biffl, O. Kovalenko, A. Luder, N. Schmidt, and R. Rosendahl. Semantic mapping support in automationml. In *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*, pages 1–4, Sept 2014. doi: 10.1109/ETFA.2014.7005276.

Jacob Persson, Axel Gallois, Anders Bjoerkelund, Love Hafdell, Mathias Haage, Jacek Malec, Klas Nilsson, and Pierre Nugues. A knowledge integration framework for robotics. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 1–8, June 2010.

O. Kovalenko, M. Wimmer, M. Sabou, A. Luder, F.J. Ekaputra, and S. Biffl. Modeling automationml: Semantic web technologies vs. model-driven engineering. In *Emerging Technologies Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pages 1–4, Sept 2015. doi: 10.1109/ETFA.2015.7301643.

Anders Björkelund, Jacek Malec, Klas Nilsson, and Pierre Nugues. Knowledge and skill representations for robotized production. In *Proceedings of the 18th IFAC World Congress*, August 28–September 2 2011.

Stefan Zander, Georg Heppner, Georg Neugschwandtner, Ramez Awad, Marc Essinger, and Nadia Ahmed. A model-driven engineering approach for ros using ontological semantics. In *Proceedings of the 6th International Workshop on Domain-Specific Languages and models for ROBotic systems (DSLRob-15) co-located with the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

Matthew Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe. *A Practical Guide To Building OWL Ontologies With The Protege-OWL Plugin*. University of Manchester, 1 edition, 2004. URL `http://home.skku.edu/~samoh/class/sw/ProtegeOWLTutorial.pdf`.

John Domingue, Dieter Fensel, and James A. Hendler, editors. *Handbook of Semantic Web Technologies*. Springer, Berlin, 2011. ISBN 978-3-540-92912-3. doi: 10.1007/978-3-540-92913-0.

SPARQL. SPARQL 1.1 Query Language. Technical report, W3C, 2013. URL `http://www.w3.org/TR/sparql11-query`.