A HYBRID METHOD FOR AN INTEGRAL FUNCTION DESCRIPTION OF BUILDING SERVICES

M. Ihlenburg[1], T. Rist[1], G. Angelika Benndorf[1], N. Réhault[1]
[1]Fraunhofer Institute for Solar Energy Systems ISE, Freiburg, Germany

ABSTRACT

Automation systems have been becoming ubiquitous in modern buildings. Despite the benefits they can bring in terms of productivity, energy efficiency and comfort improvements, they often lack interoperable descriptions of the basis functions of the systems they control, for instance Heating, Ventilation and Air Conditioning systems (HVAC). This situation hinders the implementation of methods aiming at automatizing the commissioning and the supervision of the operation of building services and thus enabling a better energy performance of buildings. In an attempt to address this issue, we propose a hybrid approach for the digital description of control functions in Building Automation and Control Systems (BACS), combining the features of existing ontologies and of a global standard for industrial control programming.

INTRODUCTION

The automation level of buildings has not stopped growing over the last decades and regulatory drivers, for instance the new Energy Performance of Buildings Directive (EPBD) in Europe, might boost the adoption of digital driven technologies such as Building Automation and Control Systems (BACS), especially in non-residential buildings (Global Energy and Environment Research Team at Frost & Sullivan, 2019).

In buildings, BACS autonomously control and monitor a wide range of services including Heating, Ventilation and Air Conditioning (HVAC) systems, which account for approximately 40% of the total building energy consumption (Bosseboeuf, 2015). During the operation phase of buildings, BACS aim at continuously controlling and monitoring the energy operation of HVAC systems. However, in many cases, the energy performance of buildings lies far below the designed targets. One of the reasons for this low performance is the frequent presence of faults and malfunctions in the operation of HVAC systems. In many cases, these faults lead to energy wastes, comfort violations for users and premature wear of components.

Whereas new BACS integrate fault detection and analytics functions, legacy BACS can hardly help identifying these issues. Furthermore, tracking a possible fault today requires considerable efforts as well as an understanding of the underlying control functions of an HVAC system. Most of these functions are implemented at the automation level of BACS in e.g. programmable logic controllers (PLC). Currently, in most of the buildings, information on these functions are either not available or not updated. If available, they are not provided in an interoperable format enabling their interpretation and use by humans or in analytics tools. In summary, whereas BACS are by definition highly digitized systems, they suffer a lack of digitally structured and interoperable information of their control functions that can be easily exploited by technicians and facility managers. This situation hinders the cost-effective implementation of corrective or optimization tasks and thus energy efficiency gains.

Over the last years, several approaches have been developed to address the lack of digitized information description methods in BACS and interoperability issues occurring from this situation. Most of them are based on the use of semantic web technologies (SWT). One main argument for the use of SWT is their capability to integrate knowledge from heterogeneous domains through a set of concepts and relationships. They can thus allow to describe the rules that govern the operation of building services in BACS. This information can then be used to support different tasks and methods like rule checking during a commissioning phase or rule-based fault detection during the operation phase of buildings.

In this paper, we present a hybrid approach for the digital description of control functions in BACS, combining the features of existing ontologies and of the IEC 61131-3 global standard for industrial control programming. These works are parts of the German funded Energie.Digital project that aims at developing an integrated representation of building service systems including physical and functional semantics as well as control logic.

STATE OF THE ART

Kučera and Pitner, 2016, present the Semantic BMS (Building Management System) ontology that aims at providing novel semantic description of building automation systems. The objective is to allow thirdparty applications to efficiently collect building operation data based on parameters that current semantic description of BACS do not provide. To overcome current data heterogeneity issues in BACS, Terkaj et al., 2017, have developed an ontology-based modelling approach using semantic web technologies. Their modular ontology reuses existing domain

 DOI: 10.3217/978-3-85125-786-1-61

ontologies like the SOSA (Sensor, Observation, Sample and Actuator) and the BACS ontologies and integrates information from Building Information Modeling (BIM) and from BACS including sensors and actuators, devices of BACS and control logic (Terkaj et al., 2017, Haller et al., 2017). Benndorf et al., 2017, developed a method and a tool chain to extend IFC files with control functions descriptions. They showed how simple control logics could be described by using existing IFC classes and how this information can be transformed and provided via SWT using the ifcOWL Web Ontology Language (Beetz et al., 2009). Dibowski and Massa Gray, 2020, presented an approach in which a knowledge graph is established as an integrated information model that provides information for all phases of a building's life cycle in a machine-interpretable way. They used existing ontologies, such as the Brick ontology, and extended them in order to model BACS requirements and control functions. One of the most complete approach using SWT for BACS control logic descriptions is provided by Schneider et al. (Schneider et al., 2017, Schneider, 2019). The authors have developed the CTRLont information model, which uses a semantic model to support and automate the engineering and operation of control logic in BACS. The concepts and relationships defined by the CTRLont allow integrating an explicit specification of the control logic with information from other domains, such as BACS and BIM models. Additionally, CTRLont also allows the specification of state-based control logic through the use of UML state machines, state graphs and schedules. The authors of the different approaches provide several use cases derived from real life applications to demonstrate their usability and potentials.

Nevertheless, a main problem with the approaches described above is the necessity to adapt and extend the existing ontologies and schemas like IFC to allow the description of the control logic, which was not the main objective of these schemas. Based on this finding, in this paper, we present a novel approach for the digital description of control functions in BACS, taking advantage of the features of the CTRLont and of the IEC 61131-3 global standard for industrial control programming. We show which elements are described in the different parts of the developed model and show how we couple and advantageously use the information in both domains on the basis of a simple control loop in an air handling unit.

DATA MODEL FOR HVAC CONTROLS DESCRIPTION

In this section, we succinctly describe the data model that we have developed for the description of control functions in BACS, which is based on the coupling of existing ontologies, the CTRLont, the SOSA and the SEAS (Smart Energy Aware Systems) ontologies with the OpenPLC project based on the IEC 61131 standard for programmable controllers.

The CTRLont ontology is a key component of our approach (Schneider et al., 2017, Schneider, 2019). The concept, the classes and their relationships are depicted in figure 1. The ontology is used to describe controllers with their inputs, outputs, parameters and their connection to the application logic.

A `controlActor` can be linked to its `Inputs, Outputs` and `Parameters` via the `hasInput, hasOutput` and `hasParameter` object properties. Each of these `AnnotedElements` can be specified by a `Medium, Unit, Quantity, SemanticType` and `DataType`. The connection of the `ControlActor` to entities of the class `ApplicationLogic` can be created by the `hasApplicationLogic` property. `Input` and `Output` can be connected via the `isConnectedTo` object property.

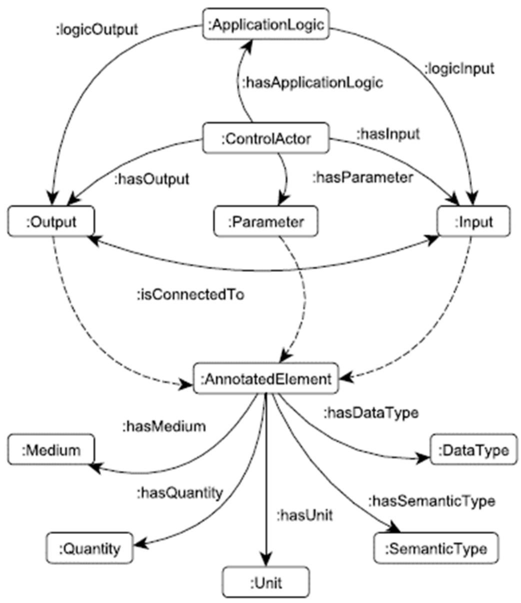

Figure 1: Concept and relationships of CTRLont ontology- own representation based on Schneider, 2019, p.67, fig.44

In addition to the CTRLont ontology, our model uses the SOSA ontology and the SEAS ontology respectively for the description of sensors and actuators and of systems (Terkaj et al., 2017, Lefrançois et al., 2017). The SOSA ontology is a core ontology of the Semantic Sensor Network (SSN) ontology. SOSA provides a standardised concept for describing sensors, their observations and actuators. The SEAS knowledge model is a family of ontologies that provide an abstract concept to describe physical systems on a generic level. In this work, we use one of its four core modules, the seas:SystemOntology, which defines a concept to define interrelated systems. The connections and interactions of different systems and their sub-systems can be described using the classes and properties of the SEAS ontology.

 DOI: 10.3217/978-3-85125-786-1-61

The last part of our model, which contains the control logic, is based on OpenPLC. OpenPLC is an open-source programmable logic controller, whose OpenPLC-Editor is the corresponding tool to create PLC programs in accordance with the IEC 61131 standard for programmable controllers (John and Tiegelkamp, 2009, Alves, 2020). The IEC 61131-3 defines three graphical (ladder diagram (LD), function block diagram (FBD), sequential function chart (SFC)) and two textual programming languages (structured text (ST), instruction text (IT)) that can be used in the OpenPLC-Editor.

METHODOLOGY

In this section, we first describe the general concept of our data model. Then, we show the implementation of the methodology on the basis of a simple use case.

Our general concept uses ontologies to describe the topological connections of the systems of which we aim to describe the control functions. The connections between sensors, controllers and the application logic are stored explicitly in an RDF (resource description framework) format. The SOSA ontology is used to characterize the sensors. The connections from the controller to sensors and actuators are described by using the SEAS ontology. The CTRLont ontology is used for the definition of the relationships between inputs, outputs and application logic of the controllers. Although ontologies have clear advantages to structure information and to maximize knowledge discoverability, they have not been adopted yet in BACS and we think that they are not suitable for the description of control logics. This is why we did not describe the actual control logic with an ontology-based method but instead used a programming language defined in the international standard IEC 61131-3 for programmable logic controllers. The program is written in the OpenPLC-Editor. The resulting source code, stored in a structured text-file (.st), is then linked to the entities of the CTRLont ontology (controllers, inputs, outputs, application logic) via a new ontology, specifically developed in our approach. Therefore it is possible to access and display the source code by using SWT. The classes and properties of this new OPLC ontology are described further below. A schema of the hybrid approach is shown in figure 2.

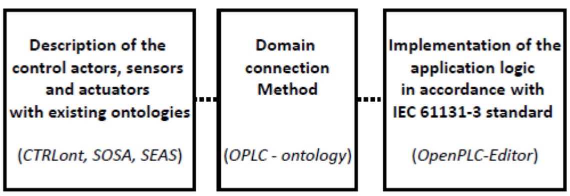

Figure 2: Schema of the hybrid approach

Use case description

We consider a simple use case of an Air Handling Unit (AHU) with a reset schedule for supply air temperature control.

The AHU comprises air filters, exhaust and supply air fans, a heat recovery system, a heating and a cooling coil. The heat recovery system can be bypassed. The bypass damper and the cooling and heating valves are controlled by actuators connected to PI controllers. In this simple use case, we illustrate our approach by only considering the cooling control loop including the two outdoor air temperature and supply air temperature sensors T1 and T2, the cooling coil valve and the controllers R1 and R2. The simplified control loop of this AHU is shown in figure 3 and its functions are described further below.

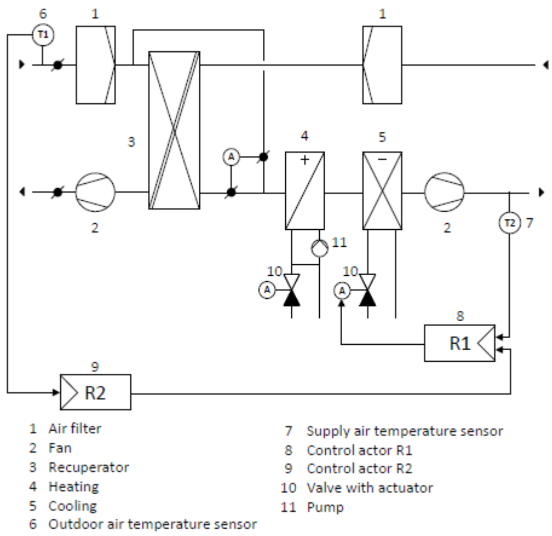

Figure 3: Simplified control loop of the implemented use case

Function description with OpenPLC

The control functions of the considered AHU control loops are only available as a textual description. In this work, we used a combination of the graphical function block diagram (FBD) and of the structured text (ST) programming language to implement the control logic. The textual description is converted into a function block diagram and the function blocks are implemented as structured text.

In this use case, we have defined the function blocks for the two controllers R1 and R2. The control logic of the controller R2 describes the calculation of the supply air temperature set point and is implemented as structured text. The supply air temperature control curve is shown in figure 4. If the outdoor air temperature is below the lower threshold of 19°C the set point of the supply air is set to 19°C. If the outdoor air temperature is above 32°C the set point is set to 26°C. In the range between these two thresholds the set point value is interpolated linearly. R1 is a PIDcontroller. Its control logic is already implemented

 DOI: 10.3217/978-3-85125-786-1-61

in the OpenPLC-Editor. The valve is controlled by the output value of the controller R1. The function block diagram of the controller R2 is shown in figure 5. It has three input variables (the outdoor air temperature and two temperature threshold values) and one output variable (the calculated supply air temperature set point).

An excerpt of the resulting structured-text (.st) file containing the implementation of the heating characteristic is shown in code 1.

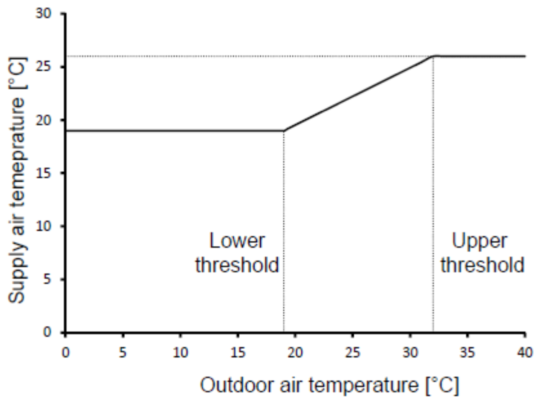

Figure 4: Supply air temperature set point in dependency of the outdoor air temperature

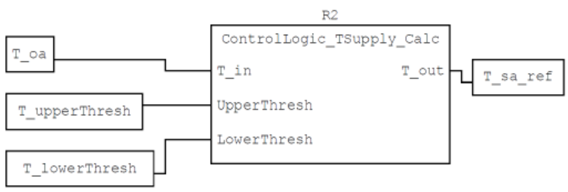

Figure 5: OpenPLC-Editor Function Block Diagram of the function block R2 that calculates the supply air temperature set point. Input: Outdoor air temperature (`T_oa`), upper temperature threshold (`T_upperThresh`), lower temperature threshold (`T_lowerThresh`); Output: Supply air temperature set point (`T_sa_ref`)

Code 1: Excerpt of structured-text file – Implementation of the function block R2 that calculates the supply air temperature set point

```
FUNCTION_BLOCK ControlLogic_TSupply
  VAR_INPUT
    T_in  :  REAL;
    UpperThresh  :  REAL;
    LowerThresh  :  REAL;
  END_VAR
  VAR_OUTPUT
    T_out  :  REAL;
  END_VAR

  IF (T_in <= UpperThresh) THEN
    T_out:= 19.0;
  ELSE
    IF (T_in >= LowerThresh) THEN
      T_out:= 26.0;
    ELSE
      T_out:= 0.538 * T_in + 8.769;
    END_IF;
  END_IF;
END_FUNCTION_BLOCK;
```

Domain connection method - the OPLC ontology

In the method we propose here, the OPLC ontology provides the tools to connect the elements of the CTRLont ontology to the control logic implemented in the OpenPLC-Editor. Up to now the ontology has not yet been formally designed for general completeness, but rather to realize the use case at hand. The following new classes and properties are defined:

Classes:

`PLCFile`

File containing the source code that describes the control logic

`PLCInputVariable`

Input variable of the `PLCFile`

`PLCOutputVariable`

Output variable of the `PLCFile`

Object Properties:

`hasImplementation`

Connection of the `ctrl:ApplicationLogic` to the PLC-file containing the source code of the control logic

`definesInputVariable`

Connection of the `PLCFile` to the `PLCInputVariable`

`definesOutputVariable`

Connection of the `PLCFile` to the `PLCOutputVariable`

`correspondsTo`

Connection of an input/output variable in the `PLCFile` to an input/output in the CTRLont ontology

DataType Properties:

`hasFilepath`

Connects the `PLCFile` to literal value of the file path

`hasVariableName`

Connects the `PLCVariable` to a literal variable name

 DOI: 10.3217/978-3-85125-786-1-61

Graph structure

An ontology-based description of the use case is created by combining the existing CTRLont, SOSA and SEAS ontologies with the OPLC ontology. The graph has been designed as a general graphical representation based on graphml first (Brandes et al., 2002). This has then been converted into a turtle syntax representation, which again can be loaded and queried using the Python rdflib library (RDFlib Team, 2020). The resulting graph structure is presented in figure 6.

USE CASE APPLICATIONS

The graph structure can be used in different further applications that require an access to the control logic information. Possible use cases are simple consistency checks of the available static information, automated PLC programming and commissioning, use for fault detection and diagnostics of building services based on sensors and actuators time-series data provided by BACS.

One application is the discovery and verification of sensor identifiers used within the control logic. This check uses the information graph to relate the domain of sensor data with the one of the control logic. To reach this, several steps are required starting with the look up of active building components (i.e. the Actuator), the search for defined control outputs associated to this component and the capture of related controllers and their inputs. We use here a single SPARQL (Prud'hommeaux and Seaborne, 2008) query to realize these steps. By the use of SPARQL Paths and the *OPTIONAL* keyword, a generic yet effective query can be formulated (see Code 2). These queries rely on the use of types and relations from specific name-spaces; i.e. the SEAS and SOSA ontologies. Depending on the use case, these types and relations could also be replaced with entities from other ontologies if needed.

Code 2: Example SPARQL query for sensor input labels of the control logic

```
SELECT *
WHERE {
?dev rdf:type seas:Actuator;
# get connected controller:
(seas:connectsTo|^seas:connectsTo)/
^ctrl:hasOutput/

#get further connected
# controllers (if applicable):
(ctrl:hasInput/
^ctrl:isConnectedTo/
^ctrl:hasOutput)*

?ctrl.

#get the sensors (if applicable):
OPTIONAL {
?ctrl ctrl:hasInput ?inp.
?inp ?pred ?sens.
?sens rdf:type sosa:Sensor.

OPTIONAL {
?sens rdfs:label ?dp.
}}}
```

Once the sensor labels are obtained they can be collated with the ones available from the data acquisition system. Regular checks for availability, quality and validity of historic time series data ensure proper foundations for the controls. Like the labels of sensors used as inputs to the control logic, the labels of output signals are obtained in a similar manner. In the above use case the output of controller R2, is used as an input of R1. Since the dependency of the controllers has been determined in the previous step the remaining input and the output of controller R1 can be shown in relation the outside air temperature, the sole input to R2 (see Figure 7). Such graphical representations can again be used by a user to get better insight in the systems behavior.

Finally the code of the control logic implementation, referenced in the graph, can be accessed and shown to a user and thus enable a full understanding and verification of the designed control strategy.

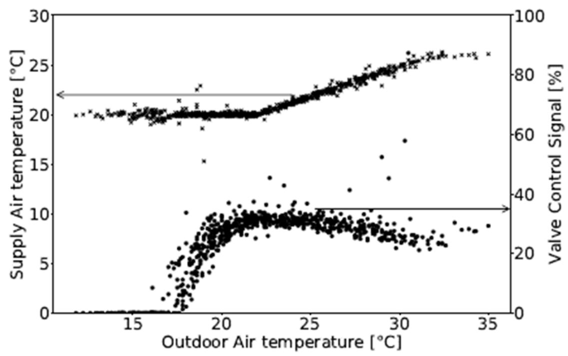

Figure 7: One month of measurement data: Input (left axis) and output (right axis) of controller R1 are lotted against the outside air temperature, the independent input to the control loop

DISCUSSION

The combination of Semantic Web Technologies with an IEC 61131-3 compatible programming language for the function description in BACS offers several advantages in comparison to the state of the art. Semantic Web Technologies allow to structure semantic information of controllers, sensors and actuators as well as their input and output data. They also allow knowledge discovery by means of queries and would thus enable retrieving information about the implemented control strategies from an unified information location during the whole life cycle of buildings. This feature would enable to streamline the commissioning and the maintenance of BACS by

 DOI: 10.3217/978-3-85125-786-1-61

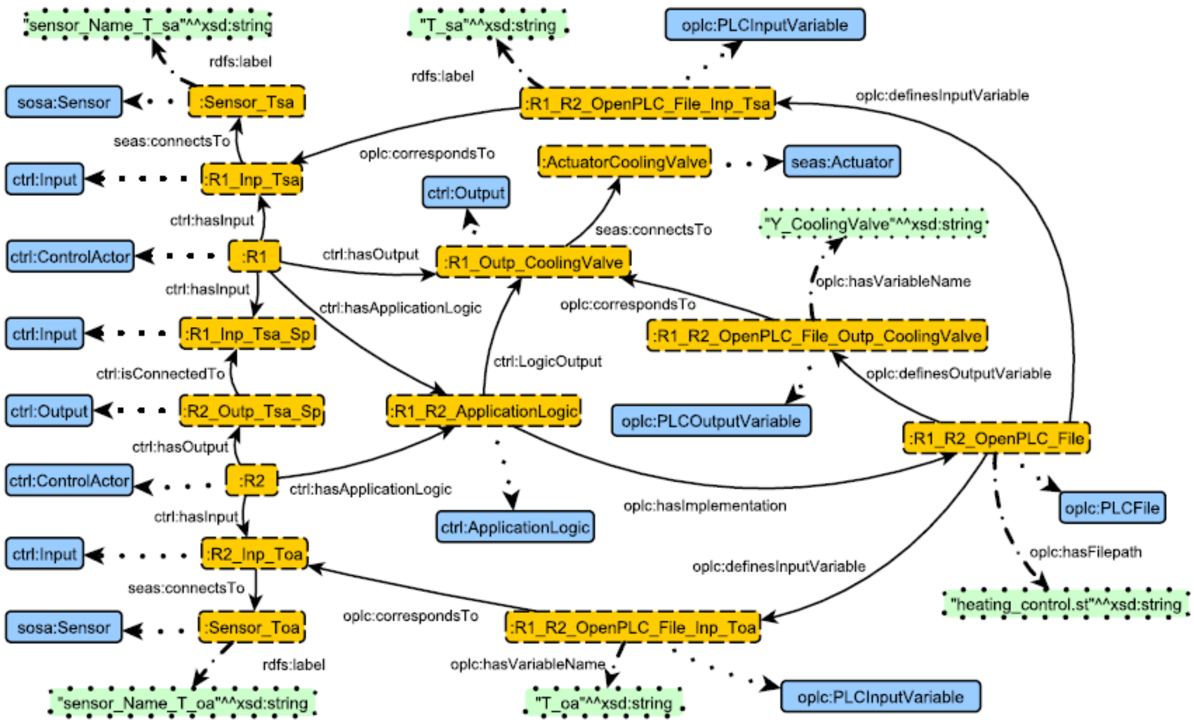

Figure 6: Ontology-based description of the use case. Legend: individual (yellow, dashed box); class (blue, solid box); literal (green, dotted box); object property (solid line); type (dotted line); data type property (dot-dashed line)

the maintenance of BACS by providing accessible and structured information to different stakeholders. However, from our point of view, ontologies are not well appropriate to describe control logics of building services and the use of programming language as defined in the IEC 61131-3 standard enables a better transfer of programmed control rules in real controllers. The developed hybrid approach combines thus the advantages of an ontology-based description of the topology of the systems with a language directly usable in Programmable Logic Controllers. It should then allow a better adoption by practitioners to support the design, commissioning and operation of BACS.

CONCLUSION AND FURTHER WORKS

In this paper, we presented a novel hybrid approach for the digital description of control functions in Building Automation and Control Systems by combining the features of existing ontology-based methods and of the IEC 61131-3 global standard for industrial control programming. We used the CTRLont, SOSA and SEAS ontologies to describe the topology of the system and the connections between sensors, controllers and actuators. We programmed the control logic description, namely the underlying control rules, with a programming language defined in the international standard IEC 61131-3 and the OpenPLC-Editor. We realised the connection between the existing CTRLont ontology with the control logic by means of a new ontology, the OPLC ontology. We then demonstrated the application of the hybrid approach on the basis of a simple use case of an air handling unit and showed how the information can be used to check the control logic of the air handling unit.

The presented approach is still in an early development stage and we have only tested it for a simple use case. In a next step, we plan to apply the method to a more complex use case and to validate our approach by interacting with a BACS manufacturer. Another objective is the extraction of fault detection rules from the model that would enable to continuously supervise the operation of building services. Another possible application is for example the use of the control logic information to emulate a controller - that is to test all its functions and remove possible errors in a virtual environment - before its installation in a building and thus enhance the quality insurance of the commissioning phase. Currently the focus is on improving the presented hybrid approach but we plan to develop a software tool which allows a user to apply the method.

ACKNOWLEDGEMENTS

These works have been carried out within the project Energie.Digital funded from the German Ministry for Industry and Energy under grant agreement no. 03ET1611. The authors would like to thank the RWTH Aachen and Viega Holding GmbH & Co. KG for their contribution to the project.

REFERENCES

Alves, T. Openplc project, 2020. URL https://www.openplcproject.com/.

Beetz, J., van Leeuwen, J., and de Vries, B. Ifcowl: A case of transforming express schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 23(1):89–101, 2009. ISSN 0890-0604. doi: 10.1017/S0890060409000122.

Benndorf, G., Réhault, N., Clairembault, M., and Rist, T. Describing hvac controls in ifc – method and application. *Energy Procedia*, 122:319–324, 2017. ISSN 18766102. doi: 10.1016/j.egypro.2017.07.330.

Bosseboeuf, D. Energy efficiency trends and policies in the household and tertiary sectors: An analysis based on the odyssee and mure databases. Technical report, 2015.

Brandes, U., Eiglsperger, M., Herman, I., Himsolt, M., and Marshall, M. GraphML Progress Report Structural Layer Proposal. In *Graph Drawing*, volume 2265 *of Lecture Notes in Computer Science*. Springer, Berlin and Heidelberg, 2002. ISBN 978-3-540-45848-7.

Dibowski, H. and Massa Gray, F. Applying Knowledge Graphs as Integrated Semantic Information Model for the Computerized Engineering of Building Automation Systems. In Harth, A., Kirrane, S., Ngonga Ngomo, A.-C., Paulheim, H., Rula, A., Gentile, A. L., Haase, P., and Cochez, M., editors, *The Semantic Web*, volume 12123 of *Lecture Notes in Computer Science*, pages 616–631. Springer International Publishing, Cham, 2020. ISBN 978-3- 030-49460-5. doi: 10.1007/978-3-030-49461-2 36.

Global Energy and Environment Research Team at Frost & Sullivan. European building automation systems (bas) market, forecast to 2025. Technical report, Frost & Sullivan, 2019.

Haller, A., Janowicz, K., Cox, S., Le Phuoc, D., and Tayor, Kerry, Lefranc¸ois, Maxime, editors. *Semantic Sensor Network Ontology*, 2017. URL https://www.w3.org/TR/vocab-ssn/.

John, K.-H. and Tiegelkamp, M. Die Programmiersprachen der IEC 61131-3. In Tiegelkamp, M. and John, K. H., editors, *SPS-Programmierung mit IEC* 61131-3, pages 103–211. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-642-00268-7. doi: 10.1007/978-3-642-00269-4 4.

Kučera, A. and Pitner, T. Semantic bms: Ontology for analysis of building automation systems data. In Camarinha-Matos, L. M., Falc˜ao, A. J., Vafaei, N., and Najdi, S., editors, *Technological innovation for cyber-physical systems*, IFIP advances in information and communication technology, pages 46–53, Cham, 2016. Springer. ISBN 978-3-319-31165-4.

Lefrançois, M., Kalaoja, J., Ghariani, T., and Zimmermann, A. D2.2 seas knowledge model. Technical report, ITEA 2, 2017.

Prud'hommeaux, E. and Seaborne, A. SPARQL Query Language for RDF: W3C Recommendation, 2008. URL https://www.w3.org/TR/rdfsparql-query/.

RDFlib Team. rdflib 5.0.0, 2020. URL https://rdflib.readthedocs.io/en/stable/#.

Schneider, G. F. *Semantic Modelling of Control Logic in Automation Systems: Knowledge-Based Support of the Engineering and Operation of Control Logic in Building and Industrial Automation Systems.* Dissertation, Karlsruher Institut f¨ur Technologie, 2019.

Schneider, G. F., Pauwels, P., and Steiger, S. Ontology-based modeling of control logic in building automation systems. *IEEE Transactions on Industrial Informatics*, 13(6):3350–3360, 2017. ISSN 1551-3203. doi: 10.1109/TII.2017.2743221.

Terkaj, W., Schneider, G. F., and Pauwels, P. Reusing domain ontologies in linked building data : the case of building automation and control. In *Proceedings of the Joint Ontology Workshops 2017 Episode 3: The Tyrolean Autumn of Ontology*, volume 2050, page 12, 2017.

 DOI: 10.3217/978-3-85125-786-1-61