Integrated Design of Simulation Models for Passive Houses

Petr Novák*,† Radek Šindelář*
*Christian Doppler Laboratory for Software Engineering Integration for Flexible Automation Systems,
Vienna University of Technology, A-1040 Vienna, Austria
{novak,sindelar}@ifs.tuwien.ac.at
†Department of Cybernetics, Faculty of Electrical Engineering,
Czech Technical University, Prague, Czech Republic

***Abstract*—Modern automation systems require both design-time and runtime integration of diverse engineering tools. Traditional integration approaches are based on repeating manual work, being time-consuming and error-prone. In this paper, applications of semantic integration, dealing with meaning of objects and their interfaces, is explained and shown on a real industrial use-case. Simulations are useful tools for process optimization or performance testing and the presented methodology makes their design for particular industrial plants flexible. The use-case shows that the design of simulation models for passive houses can be user-friendly and feasible even for non-experts as it is based on a graphical tool that enables to draw a passive house floor plan. Since neither this tool nor a universal simulation library, comprising atomic simulation blocks, were intended for simulation purposes, the presented methodology is a typical example of tool integration having heterogeneous data models.**

The goal of this paper is to propose an ontology-based formalization of knowledge representing structures of real industrial plants and simulation models. The paper also introduces the design of simulation models for passive houses from other engineering sources, which can be used by non-experts for simulation modeling. The practical usage is restricted by the fact that simulation parameters must be entered manually. The main contributions of the paper are the proposed structure of an automation ontology and a workflow of simulation model design that is not common in engineering disciplines.

***Keywords*-Semantic integration, simulation model, passive house, ontology, automation system design phase.**

I. Introduction

Simulation models emerged as a very efficient way to optimize process operation. They can be used for performance testing of control algorithms or the whole industrial systems under both normal and extreme conditions, as well as for many other engineering tasks. Nevertheless, several issues dealing with simulation models have not been satisfactorily solved. The integration of simulation models and the cooperation with other engineering tools still remain problems as well as the high requirements on engineering knowledge and skills to create and configure them. Therefore, simulation models are usually designed and performed only by simulation experts.

This paper contributes to improve the simulation model design phase, and it shows how engineering sources can be used to enhance simulation model usage for non-experts in simulations. The presented approach is based on Semantic Web technologies. Knowledge about the tools under integration is stored in ontologies. Ontology-based querying and reasoning techniques are used to retrieve the information and derive new pieces of engineering knowledge. The proposed solution realizes an ontology-based middle-ware layer between the tools. A use-case project, dealing with a design of simulation models for passive houses, is described in this paper. It motivates the research, the examples from this domain are given, and in the final part, this use-case project is discussed and evaluated.

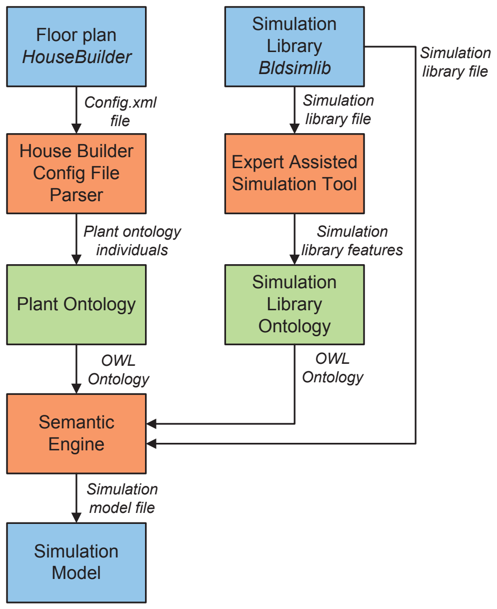

Figure 1. Workflow presented in this paper.

The goal of this paper is to present a formalization of real plant data in a machine-understandable way and to show the benefits of semantic integration of heterogeneous data models on a passive house model use-case.

The workflow of the presented use-case project is depicted in Fig. 1. The entries for the presented methodology are a representation of a graphical floor plan of a particular passive house created in "House Builder WPF" software and a universal simulation library "Bldsimlib" [1] comprising generic simulation blocks for passive house simulation [1]. Both these entries are suitable examples of engineering sources having heterogeneous data models that must be integrated involving meaning of data and interfaces. Floor plan of House Builder WPF software originally solves for visualizing passive house runtime data in a tool House Viewer WPF, but in our approach, we also use the same file without any modification for defining so-called plant ontology individuals, i.e. ontology-based representation of the passive house structure. Simulation blocks and their features are formalized in a so-called simulation ontology. Consequently, plant and simulation ontologies are used to semantically create a simulation model for a passive house.

The remainder of this paper is structured as follows: the second section summarizes a related work. The third section formulates research issues that are addressed in the fourth section, summarizing a methodology for formalization of plant, simulation and other engineering knowledge in general, and in the fifth section, describing the use-case project and its results. The sixth section concludes and proposes further work topics.

II. Related Work

Semantic integration is a perspective way to integrate diverse systems and tools, which is based on data meaning. Semantic level stands on top of a technical integration level, that is concerned with data transfers. Further explanation can be found e.g. in [2]. Although the semantic integration can be implemented in many ways, the wide-spread approach is based on Semantic Web technologies, especially representation of knowledge in ontologies.

The term ontology, originating from philosophy, is in engineering defined in many ways [3]. One of the most cited definitions is by Gruber: *"An ontology is an explicit specification of a conceptualization"* [4]. In the presented approach, ontologies are represented in OWL DL[2] format that provides a suitable compromise between expressive power and performance of reasoning. We use ontology querying language SPARQL[3] and ontologies are managed from Java code via framework ARQ[4], providing query engine on top of Jena API[5].

Although for *simulation integration* general-purpose techniques such as DCOM, CORBA, J233 could be used [5], there exist frameworks including standard vocabulary for simulation integration, such as DIS, SEDRIS or HLA [6]. Especially HLA framework is widely cited, but it does not support integration on semantic level. Therefore, we focused on ontology-based approaches.

[1]Acronym of "Building Simulation Library".
[2]http://www.w3.org/TR/owl-features/
[3]http://www.w3.org/TR/rdf-sparql-query/
[4]http://jena.sourceforge.net/ARQ/
[5]http://jena.sourceforge.net/ontology/

The *usage of ontologies in modeling and simulations* is introduced e.g. in [6]. In [7] the Ontology-driven Simulation Tool (ODS) is described. The approach is based on two ontologies: a domain ontology categorizing a knowledge including a problem vocabulary and its concepts are mapped onto a modeling ontology being used for the simulation model description. Our approach distinguishes between plant domain and simulation knowledge in a similar way.

An ontology-driven simulation model design is presented in [8]. The paper is focused on generating MATLAB-Simulink blocks and defining them via DAVE-ML according to the domain ontology. Connection of these blocks is done manually, thus this approach is complementary to the methodology explained in this paper.

The methodology presented in this paper uses so-called power bonds and signal bonds to classify a type of device interconnection. These terms originate from a *bond graph theory*, that is introduced e.g. in [9]. Power bonds are common in real physical systems where the flow of energy defines power transmissions, whereas signal bonds refer to interconnections where energetic interactions can be ignored. For example, there is usually assumed that sensors have no impact on measured variables, thus this kind of relationship is called signal, whereas e.g. tanks and outlet pipe interact by power bonds.

III. Research Issues

The problems, which are discussed in this paper, can be summarized into following research issues.

RI-1: Formalization of plant and simulation knowledge in general. As real plants have diverse structures and devices, there is a need for formalizing their description. Such a formalization is useful for reusability, flexibility in terms of process redesign, and automatic or semi-automatic methodologies for supporting both design and run-time phases of the automation system lifecycle.

RI-2: Applications of the formalization for design of simulation model for a particular passive house. The particular use-case project integrates the two standalone engineering tools. A universal library Bldsimlib is implemented in MATLAB-Simulink[6] and it comprises so-called generic simulation blocks. They approximate building elements, such as windows, walls, doors, or rooms. Usually, only simulation experts are able to create and perform the simulation model. We try to overcome this shortcoming by integrating the second tool, the graphical House Builder WPF application. The XML config file, being its output, is used to recognize a structure of the house. It is stored in a machine-understandable form and consequently, a simulation model is generated semi-automatically.

[6]http://www.mathworks.com/products/

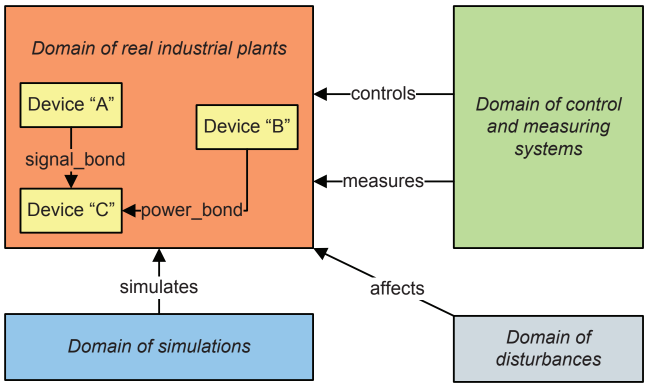

Figure 2. Formalization of problem domains and relationships of their elements.

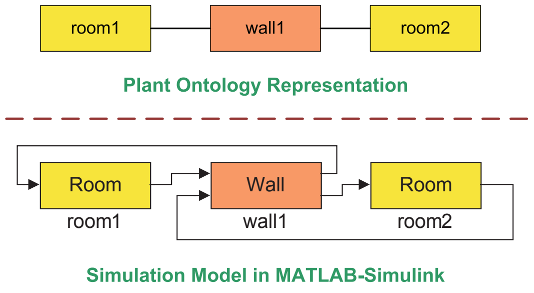
room1
wall1
room2
Plant Ontology Representation
Room
Wall
Room
room1
wall1
room2
Simulation Model in MATLAB-Simulink

Figure 3. Comparison of interconnections in terms of plant description and signal-oriented simulators.

IV. Formalization of Plant, Simulation, and Other Engineering Knowledge

Our approach is based on explicit distinction between plant, simulation, and other engineering knowledge. Plant knowledge is related to existing devices and elements, whereas simulation knowledge comprises features of available simulation blocks, e.g., their interfaces. We store plant knowledge in a so-called plant ontology, and simulation knowledge in a so-called simulation ontology. Further, we define supplemental ontologies such as a signal ontology.

Specification of fundamental domains and their relationships is introduced in Fig. 2. The upper left set represents a real plant domain. The figure depicts that real plants comprise devices, that are connected by two basic domain-specific terms: "signal_bond" and "power_bond", that we use in compliance with bond graph theory to define the real device interconnections. We consider a measuring and control system as one domain, which interacts with the real plant domain via properties "measures" and "controls". We explicitly define disturbances, i.e. factors that influence the real plant and that are usually not desired from a control point of view. For example, when controlling a temperature in a house, disturbances are the sun, humans, or opening and closing doors as they impact on the controlled variable. Simulation models are interconnected with a real plant via the relation "simulates", expressing that some real plant device is approximated by the particular generic simulation block. According to our industrial experiences, this relationship is not usually 1:1 but 1:n, i.e. one plant device can be simulated by more than one simulation blocks.

An important issue for simulation model design is a decomposition of power bonds for signal-oriented simulators, such as MATLAB-Simulink. Figure 3 depicts a description in the plant ontology and the corresponding schema in MATLAB-Simulink. We can see that each power bond is decomposed into two signal bonds, i.e. two interconnections in the signal-oriented simulator. In our approach, translation of the bonds is realized while assembling the simulation model in Java code, performing series of SPARQL queries.

As we defined several domain ontologies that describe the real plant and simulation, the relationship between these ontologies are realized via relations summarized in the previous paragraph. All of the ontologies and their relationships build a so-called automation ontology, being depicted in Fig. 4. The rectangular blocks represent ontology classes, whereas the rounded blocks are ontology individuals. For better readability, some individuals are omitted, but the fundamental ideas are covered by this figure. We can see the classes filled in blue color that represent an upper layer, shared within diverse automation systems. The plant ontology comprises real industrial plant devices, which are categorized into five classes. "Actuator" class involves controllable devices, "Passive Element" represents uncontrollable devices. "Disturbance" affects real plant, it defines boundary conditions and it is often non-measurable and random. "Measure point" defines sensors or softsensors that are software algorithms calculating a value from other variables. Blocks filled in yellow color depend on a type of a particular plant, in our case an example passive house classes are depicted (further classes were omitted for better readability). The simulation ontology comprises the description of available simulation libraries and final simulation models including their interfaces. Since industrial devices and tools usually requires information about not only the connections, but port numbers as well, the individuals of "Port" are depicted. They represent all available ports and specify their signal types. Last but not least, "Power Bond Decomposition" labels doubles of ports that decompose power bonds, i.e. they define signal routing in signal-oriented tools.

Although a structure of the automation ontology could seem complicated at first, it provides powerful support for diverse engineering tasks. Furthermore, it is not expected to be modified by hand in a general-purpose ontology editor, but either in specialized editors implemented for automation purposes or via specialized tools such that a user does not interact with this ontology at all. The latter case is used in the passive house use-case presented in this paper, as the plant ontology is created semi-automatically by the parser

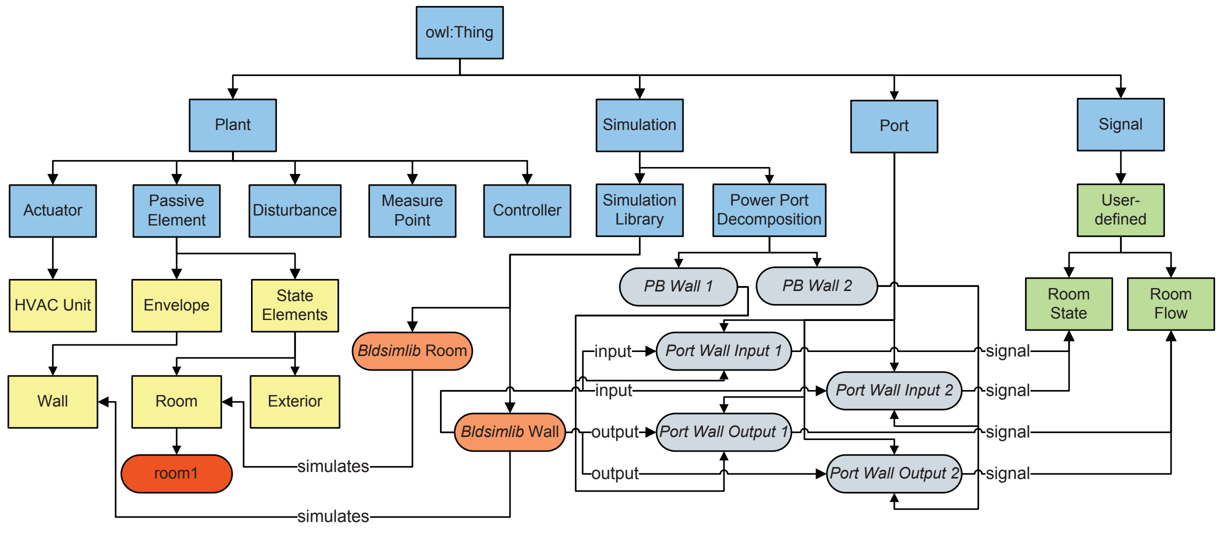

Figure 4. Automation ontology: Machine-understandable knowledge-base formalizing industrial plant, simulation and other engineering knowledge.

```
<RoomPolygon BackgroundImage="laminate_flooring2.png"
BackgroundImageWidth="200" BackgroundImageHeight="200">
<LinePoint X="6" Y="0" WallThickness="50" CircleNumber="-1" />
<LinePoint X="6" Y="1000" WallThickness="50" CircleNumber="-1" />
<LinePoint X="1006" Y="1000" WallThickness="50" CircleNumber="-1" />
<LinePoint X="1006" Y="0" WallThickness="50" CircleNumber="-1" />
</RoomPolygon>
```

Figure 5. An example of room description in a House Builder WPF configuration file `config.xml`.

of a passive house floor plan. Although the description of simulation model blocks, i.e. a simulation ontology, a port ontology and a power bond ontology must be entered by humans, we are implementing a tool that supports this work and makes it easy and user-friendly.

Since simulation parameters, i.e. parameters required for the parametrization of simulation blocks, can differ from parameters of real plant devices in both count and scale, every generic simulation block has the parameters described in the simulation ontology. A special kind of parameters are initial conditions that are managed in a similar way as parameters in our approach. Known parameters of real plant devices are stored in a plant ontology. Nowadays, the translation of plant parameters to simulation parameters have not been satisfactorily solved in the presented prototype. Therefore, the semantic engine and the parser are referred as semi-automatic, meaning that structural issues are solved automatically but the parameters are managed manually.

While an automation ontology is created, the semantic engine can semi-automatically assemble the simulation model for a particular plant. It finds all real plant devices (i.e. individuals of the plant ontology) and according to the ontology property "simulates", a set of simulation candidates and their interfaces are retrieved for each plant device via SPARQL queries. Consequently, the interconnections are set in a Java loop taking the free ports and decomposing the power signals by doubles of signal interconnections in case of signal-oriented tools.

V. Use-case Project: Integrated Design of Passive House Simulation Model

The goal of this section is to provide a solution for the research issue RI-2, i.e. to create a simulation model for a passive house, whose floor plan is drawn in House Builder WPF software. The simulation model is created by selecting and interconnecting generic simulation blocks from a universal library Bldsimlib, implemented in MATLAB-Simulink. This task is challenging as House Builder WPF and the universal simulation library have heterogeneous data models.

Figure 6 depicts a use-case passive house floor plan in House Builder WPF tool. The use-case passive house is a simplified house, consisting of two rooms, walls and interior equipment. As it is very simplified, it enables to show our approach in a clear way and it does not pose any restriction in generality. An exemplary piece of the House Builder `config.xml` file is shown in Fig. 5. The figure depicts the House Builder representation for the left room of the passive house floor plan. Rooms can be found via keyword `RoomPolygon`, whereas walls via keyword

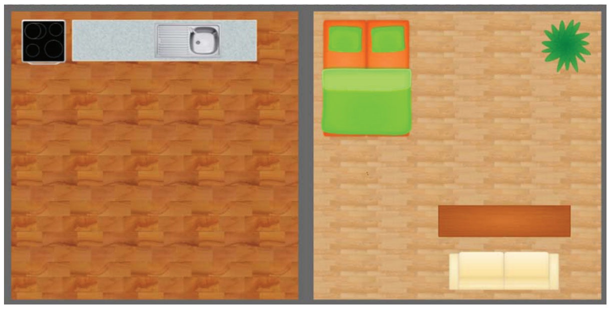

Figure 6. Floor plan of the use-case passive house created in House Builder WPF software.

`LinePoint`. First step of our solution is a parsing of House Builder `config.xml` file that gives information consequently used to create individuals of passive house ontology, i.e. the instances of rooms and walls are created and their interconnections are inserted as object ontology properties. The presented version of the algorithm supports rooms and walls only, but the functionality is planned to be extended for further elements support. Some elements must be recognized in a non-intuitive way, as House Builder does not express them. For example, there is no graphical element representing doors or windows, but there are so-called sensors measuring and visualizing position of sun-blinds or position of door. This is done by intended functionality of House Builder WPF oriented on passive houses, where windows are expected usually equipped with sun-blinds and furthermore, the floor plan need not to be absolutely exact in all details as it monitors the operation of the whole automation system of the house.

The parser reads the `config.xml` file and for known keywords creates plant ontology individuals. The parsing is done via Java code and creating individuals is realized by ARQ/Jena methods. Another alternative would be to express House Builder WPF elements in a specialized ontology and map its concepts onto plant ontology, but the first approach was used for this prototype as it is easier and reaches satisfactory results.

While having the passive house representation in the plant ontology, that is a tool-independent representation of the object, the semantic engine generates the simulation model. The simulator-independent part of the engine is implemented in Java. It is called from supporting MATLAB script, where simulation blocks and signal interconnections are entered via MATLAB API using functions `add_block` and `add_line`. As we assume that there is available a simulation library comprising generic simulation blocks, the creating of models means selecting appropriate library blocks, entering them into a simulation model file, setting their name uniquely and interconnecting them according to the real plant structure. To run the simulation model, also simulation parameters must be added, but as this issue

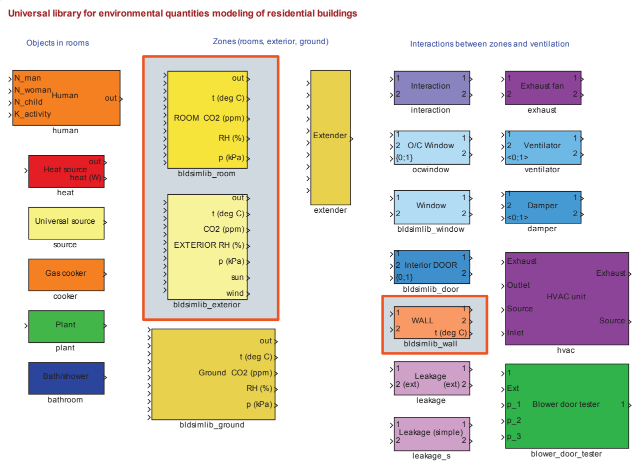

Figure 7. High-level overview of a universal simulation library Bldsimlib in MATLAB-Simulink.

exceeds the size of this paper, we will address it in future work. The high-level overview of the used Bldsimlib library is depicted in Fig. 7. The emphasized generic simulation blocks are supported in the current version, i.e., simulation blocks of exterior, room, and wall are used in generated model.

The semi-automatically generated model of the passive house is depicted in Fig. 8. The upper left block represents an exterior, i.e., it defines borderline conditions for the simulated house. Simulation blocks in the central column represent rooms and the blocks in the right part of the figure approximate walls. The depicted simulation model does not export its outputs into a file or MATLAB Workspace and although it would be possible to set all variables as outputs automatically, we expect either the user of a simulation model would define the outputs or in the planned extended version, the outputs will be defined by ontology property "measures" as well as the inputs will be entered via ontology property "controls". In this use-case simulation model, a limitation of the current implementation is apparent: The walls that separate two same areas are not merged into one wall. In other words, each room in the generated model has four walls, but they could be merged into two walls without changing the functionality - one wall to exterior and one to the other room. This issue could be solved on the level of the configuration file parser, but we consider this problem being general. As it occurs in many real-life systems, we plan to handle this situation on a plant ontology level. The problem exceeds this paper; it is planned for future work. Last but not least, the positions of blocks are done by a rectangular matrix and signal wires use auto-routing available in MATLAB; the positioning is planned to improve.

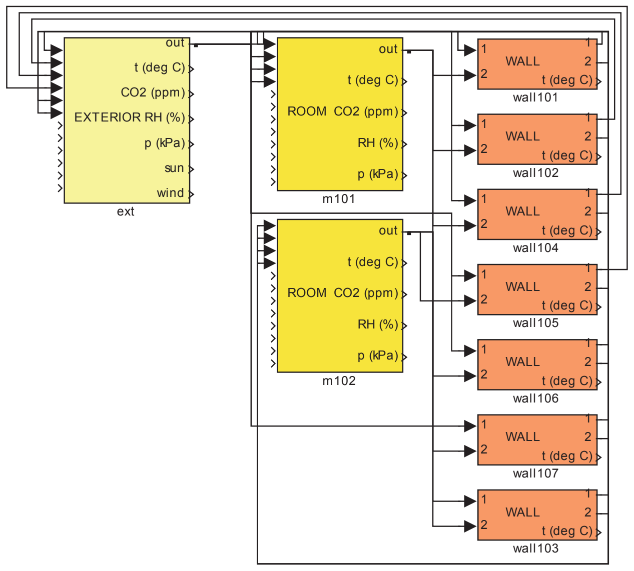

Figure 8. Simulation model of the use-case passive house that was generated semi-automatically by the parser and the semantic engine from passive house floor plan and generic blocks from Bldsimlib library.

VI. Conclusion and Future Work

This paper describes the usage of ontology-based representation of plant, simulation, and other engineering knowledge. The information is structured in order to generate simulation models semi-automatically and to support the design-phase of automation system. The presented methodology is demonstrated on the real industrial use-case project dealing with the design of simulation model for a passive house. As the approach is oriented (but not limited) for signal-oriented simulators, such as MATLAB-Simulink, the attention is paid into decomposition of power bonds to the doubles of signal bonds, supported by such tools. The use-case illustrates generating simulation model structure from the floor plan created in the particular graphical tool. As the data models of available simulation library and floor plan configuration file have different structure, it is shown that the information is recognized in the parser and afterwards, plant ontology individuals are created. Consequently, the simulation model is created semi-automatically by the semantic engine that is general-purpose in terms of applicability on various types of industrial plants.

The proposed methodology guarantees avoiding structural errors, reduces manual and error-prone work, and saves development time and costs. The restriction of the presented methodology is a need for entering simulation parameters manually as well as some pieces of information are not covered in House Builder WPF, such as a topology of the HVAC system. In *future work*, we plan to extract the parameters from a so-called Passive House Planning Package (PHPP)[7] that is the Excel script widely used in civil engineering to calculate and evaluate thermal and other properties of passive houses. Other future work issues are the support for simulation model input and output management, that will work not only on the simulation model and visualization level but as well on the run-time level. Further future work topics are a positioning of simulation blocks and signal wires to obtain better readability for humans, and automatic merging simulation blocks on the ontology level.

[7] http://www.passiv.de/07_eng/phpp/PHPP2007_F.htm

Acknowledgments

The authors would like to thank to the partners from the Christian Doppler Laboratory for Software Engineering Integration for Flexible Automation Systems for the discussions and feedbacks. This work has been supported by the Christian Doppler Forschungsgesellschaft and the BMWFJ, Austria.

References

[1] P. Novák, "Modelling and Control of Environmental Parameters of Passive Houses," in *POSTER 2010 - Proceedings of the 14th International Conference on Electrical Engineering*, 2010.

[2] T. Moser, R. Mordinyi, W. Sunindyo, and S. Biffl, *Canadian Semantic Web: Technologies and Applications*. Springer, 2010, ch. Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constraints.

[3] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*, second printing ed. London: Springer-Verlag, 2004.

[4] T. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5(2), 1993.

[5] J. Hu and H. Zhang, "Ontology based collaborative simulation framework using HLA and Web Services," in *World Congress on Computer Science and Information Engineering*, vol. 5, 2009, pp. 702–706.

[6] L. Lacy and W. Gerber, "Potential modeling and simulation applications of the web ontology language - OWL," in *Proceedings of the 2004 Winter Simulation Conference*, vol. 1, Dec 2004.

[7] G. Silver, O.-H. Hassan, and J. Miller, "From domain ontologies to modeling ontologies to executable simulation models," in *Proc. of the 2007 Winter Simulation Conference*, 2007, pp. 1108–1117.

[8] U. Durak, S. Güler, H. Oğuztüzün, and S. K. İder, "An exercise in ontology driven trajectory simulation with MATLAB SIMULINK (R)," in *Proceedings of the 21th European Conference on Modelling and Simulation*, Prague, 2007.

[9] P. Gawthrop and G. Bevan, "Bond-graph modeling," *Control Systems Magazine, IEEE*, vol. 27, no. 2, pp. 24 –45, 2007.