Applications of Ontologies for Assembling Simulation Models of Industrial Systems

Petr Novák[1,2] and Radek Šindelář[1]

[1] 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
[2] Department of Cybernetics, Faculty of Electrical Engineering,
Czech Technical University in Prague, 121 35 Prague, Czech Republic;

Abstract. Simulation models are important parts of industrial system analysis and control system design. Despite the wide range of possible usage, their formalization, integration and design have not been satisfactorily solved. This paper contributes to the design phase of simulation models for large-scale industrial systems, consisting of a large number of heterogeneous components. Nowadays, it is the simulation expert who assembles the simulation model for a particular industrial plant manually, which is time-consuming and error-prone. We propose to use a semi-automated semantic engine that assembles the simulation model. We represent a structure of a real industrial plant in a plant ontology and available simulation blocks in a simulation ontology. Signals of each simulation block are specified via signal ontology. As the knowledge is formalized, the simulation model can be assembled automatically, based on the ontologies and SPARQL queries. Since each real plant device can be represented by more than one simulation blocks, the selection of suitable simulation candidates is based on matching interfaces of neighboring blocks. In the presented case, simulation models can be efficiently redesigned and their components can be reused and shared; the methodology contributes to avoiding errors in both run-time and design phases. Evaluation on a real-life industrial use-case, dealing with design of simulation models of passive houses shows improvements in both reducing development time and avoiding design errors. Major results of this paper are the proposed structures of the ontologies and the SPARQL query realizing the selection of the appropriate simulation blocks.

Keywords: ontology, simulation models, industrial automation, SPARQL querying, design and integration

1 Introduction

Modern industrial automation systems are sophisticated systems comprising various computer algorithms and tools. Although the production automation has been investigated for several decades, the integration of diverse tools and sharing knowledge between particular engineering disciplines [8] still remain challenging problems. As the automation systems touch a wide range of problems from industrial devices to optimizers or even managerial supervisory tools, this paper is focused on simulation models, i.e. the software approximations of the behavior of real industrial systems that are useful for the optimization of the real system operation.

Simulation models are able to cover a lot of industrial engineering tasks. They can be used to perform experiments that would be, for example, dangerous on the real plants themselves, such as testing diverse critical scenarios of nuclear power plants [9]. As some real experiments cannot be repeated under the same conditions, such as testing performance of buildings and their control algorithms thanks to the outside weather being every time unique, the models can contribute to optimize operation of such systems, too. Although simulation models can be used in many domains, neither their integration nor their design have been satisfactorily solved.

The goal of this paper is to introduce a significant impact of using ontology-based methods on simulation model design phase. An approach used nowadays and the proposed ontology-based methodology are compared in Fig. 1. Instead of manual and error-prone work we propose to formalize plant and simulation knowledge. Consequently, a semantic engine can be used to assign the simulation model automatically. The labels of the edges denote the kinds of data that flow between the engineering steps. The formalization using ontologies provides powerful support for various sound methods such as ontology querying and reasoning and it is very efficient thanks to heterogeneous data model structures that occur not only on the automation system level, but on the simulation level as well. The exemplary SPARQL query implementing the core functionality of the semantic engine is given in this paper.

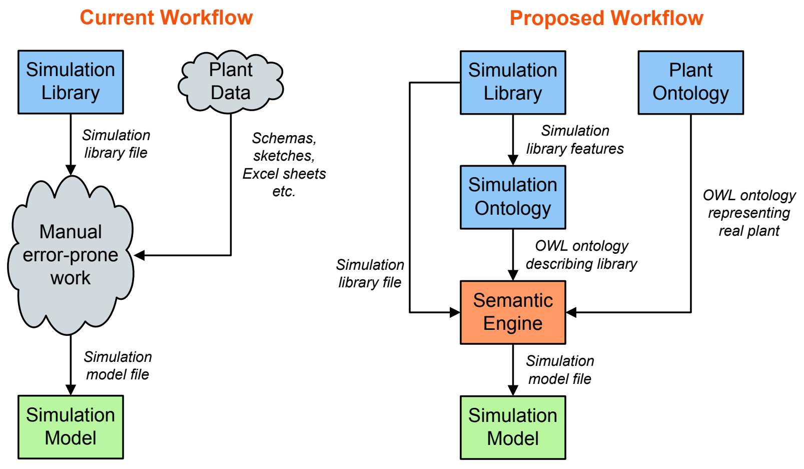

Fig. 1. Comparison of current and proposed workflows.

The usage of the presented methodology guarantees avoiding structural errors in simulation models design. Such kind of errors is hard to detect manually in real industrial cases. Further, as the meaning of signals is expressed, the semantic engine guarantees a compatibility of interacting signals or detects errors in existing models such as different scales of the physical variables. On the other hand, our approach brings a requirement on a consistency of the describing ontologies. The presented methodology is limited for single-input and single-output simulation blocks connected in series, but it can be generalized for an arbitrary kind of blocks.

The remainder of this paper is structured as follows: Section 2 summarizes the related work, the third section characterizes a problem scope and research issues. The fourth section describes our solution to formalize plant and simulation knowledge in ontologies and the fifth section introduces ontology-based methods to support simulation model assigning from the atomic simulation blocks. The methodology is demonstrated on a use-case electrical circuit in the sixth section. The seventh section concludes and proposes further work.

2 Related Work

The term ontology originates from philosophy, where it refers to the theory of existence. In technical or scientific sense it is defined in many ways [1], [2]. One of the most cited definition is by Gruber: *"An ontology is an explicit specification of a conceptualization"* [3]. The core feature of ontologies is the representation and sharing of domain knowledge in a machine-understandable form. Ontologies comprise concepts (also called ontology classes), individuals (i.e. instances of ontology classes), properties and constraints. At a concept (class) level, the relations are usually declared and they are defined at an individual level.

In our approach, OWL DL[3] is used to represent the ontology. It was selected as a optimal compromise between expressive power and possibilities to perform reasoning in future work. We derive required information by SPARQL[4] queries; nowadays a reasoner is not used. For modifying or visualizing ontologies, we use Protégé[5], and the automatic semantic engine was implemented in Java, using ARQ[6], that is a query engine extending Jena Ontology API[7].

The Semantic Web technologies has been already used for simulation modeling [7]. In [4], the Ontology-driven Simulation Tool (ODS) is described. It is based on two ontologies that are mapped: a domain ontology categorizing a knowledge including a problem vocabulary, and a modeling ontology being used for the simulation model description. Our approach is based on a similar idea, but tries to cover the whole automation system. Ontologies are used as a knowledge representation on a middle-ware layer that manages diverse engineering tools or

[3] http://www.w3.org/TR/owl-features/
[4] http://www.w3.org/TR/rdf-sparql-query/
[5] http://protege.stanford.edu/
[6] http://jena.sourceforge.net/ARQ/
[7] http://jena.sourceforge.net/ontology/

simulation models of diverse simulators in a uniform way, in order to overcome one of the most important problems of simulation model design and integration: *"There is no commonly agreed upon format for representing and storing models"* [4]. The usage of ontologies for production automation is described, e.g. in [5], where can be found approaches touching further parts of automation system.

The presented task is an application of semantic integration [6] - i.e. the level of the system integration dealing with the meaning of exchanged data. In our case, the meaning of plant devices, simulation blocks, their interfaces and interchanged signals has to be stored. This request is satisfied in the automation ontology that is described in the following text. The presented approaches and algorithms were tested on simulation models implemented in MATLAB-Simulink[8], but they are not limited for this tool.

3 Problem Scope and Research Issues

As the industrial plants typically contain devices that are repeated several times in their topological structure, one of the first tasks for the simulation expert is to analyze which devices will be simulated in the same way and what parameters are required. For example, a water distribution system involves devices such as piston pumps and rotary pumps, tanks, and pipes. The engineer has to decide, if the rotary pumps and piston pumps will be simulated by the same simulation block or if their behavior is so different that each kind of pump will be simulated separately. The functionality of the plant devices is comprised in so-called generic simulation blocks, i.e. blocks that have to be parameterized to obtain the simulation for the particular device. The parameters are, for example, diameters or lengths, but can also depend on the classification of the blocks as some parameters can emerge to fine-tune the model or simplify its structure. These blocks are comprised in a so-called simulation library, sometimes also called universal simulation library to emphasize the necessity for parametrization.

A simulation model for the particular plant is created by copying generic blocks from the library into a simulation model file, interconnecting them and entering their parameters. A simplified example of the simulation library is depicted on the right side of Fig. 5. This library comprises generic simulation blocks representing an ideal source of voltage, ideal capacitor, and two resistors. Only the dominant inputs and outputs of the blocks are depicted; additional voltage inputs and outputs that are needed for implementation are omitted.

The research work presented in this paper was classified into following research issues.

RI-1: Formalization of industrial plant structure. The plant description is not usually well-structured, even it is not stored in an electronic form in many cases in industrial praxis. The formalization of the knowledge in a machine-understandable way, enabling to use automatic methods for retrieving required information and deriving new facts, improves both design and runtime phases of automation systems.

[8] http://www.mathworks.com/products/simulink/

RI-2: Formalization of universal simulation library features. Although simulation models are computer structures, a lot of pieces of information that was used during analysis and design phases of universal simulation libraries are not stored in a machine-understandable way. In contrast, it is usually given in some documentation and modifying the particular simulation model requires the simulation expert to study the rules for building the simulation model, the description of inputs and outputs, the meaning of signals etc.

RI-3: Industrial plant – simulation mapping. As the simulation models approximate the industrial plants and the generic blocks approximate the industrial devices, for both design and interpretation of simulation models the relationship between real industrial domain and the simulation domain is crucial.

RI-4: Semi-automated semantic assigning simulation models. Assigning a simulation model for a particular industrial plant is a time-consuming and error-prone engineering task. For real large-scale systems, the simulation expert has to copy hundreds of blocks and enter thousands of interacting signals according to the real plant structure, as well as setting block names, parameters, and model interfaces.

4 Ontology-based Formalization of Knowledge

This section summarizes our solution dealing with the formalization of plant and simulation pieces of knowledge and emphasizes their relationships. It addresses the research issues RI-1, RI-2, and RI-3.

Our approach is based on strong distinguishing between plant and simulation knowledge. We use a so-called plant ontology (RI-1) to represent the knowledge about the real plant, hence it comprises especially structure of the real industrial plant (i.e. particular devices, their interconnections etc.) and parameters of the plant (e.g. lengths or diameters) The plant ontology does not contain any information about the simulation model. Since we can use more than one simulation models, the knowledge related to available simulation blocks is comprised in the simulation ontology (RI-2). This ontology represents information about the structure of the universal simulation library and its generic blocks. Important knowledge relates to inputs and outputs of the generic blocks. Simulators does not check compatibility of signals, hence there can be set, for example, electric current as voltage input for consequent blocks. For this purpose, so-called signal ontology is defined as well, being a taxonomy of occurring signals. Therefore, the presented approach can be used for checking engineers' solutions as well.

For the integration and semi-automatic designing methods, the relationship of the three ontologies plays an important role. Since each real plant device can be represented by more than one generic simulation block, the crucial issue is to represent explicitly the mapping between real devices and simulation equivalents (RI-3). The mapping is realized by the object property "simulates" that relates plant device and its simulation equivalents. The property "simulates" can be seen in Fig. 2 where a real plant device Resistor is approximated by two generic simulation blocks: "Sim Resistor 1" and "Sim Resistor 2". We assume that to one real device belong one or more simulation blocks. If we enabled a more general situation, when not only each real plant device could be approximated by several simulation blocks, but each simulation block could approximate more than one plant devices as well, this relationship would must be decomposed adding a layer of individuals providing the mapping. Such a kind of decomposition of ontology properties emerges in automation systems in a lot of scopes, typically, as there must be represented port numbers of the devices. It would be useful to define the predicate "is_connected_to" having four parameters, but it is possible neither in description logics nor in OWL DL. The arity of predicates is two by definition, such relationships have to be decomposed, e.g. by involving at least one layer of individuals. Trying to avoid technical and implementation details, we present a simplified situation where only one port is used.

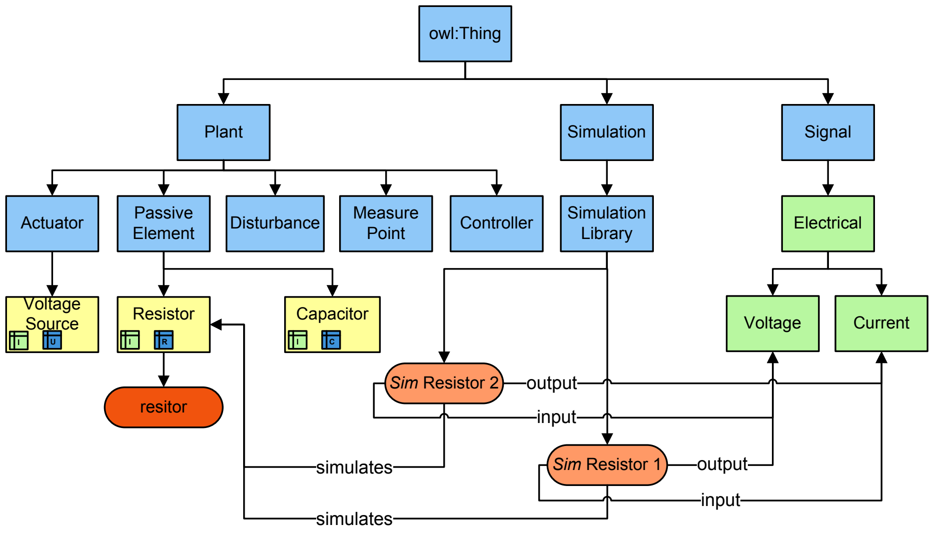

Fig. 2. Automation ontology.

The plant, simulation and signal ontologies are depicted in Fig. 2, where these ontologies with their mappings are shown in a compact form, briefly called "automation ontology". The rounded blocks represent ontology individuals, the rectangles are ontology classes. The upper part of the figure, filled in blue color, is common for all feasible automation systems, the other concepts are domain-specific. On the left side of the figure, there is depicted the plant ontology, classifying its devices into "Actuator" that comprises devices being active from a control point of view, "Passive Element" that cannot be controlled, "Disturbance" that are uncontrollable conditions and phenomena, "Measure point" being in domain of property "measures", and "Controller" performing control algorithms. The further level of the plant ontology is dependent on the type of the industrial plant, in Fig. 2 a simplified electrical circuit was used and its three devices are depicted. Note that real plant classes declare properties to express the interconnection of the devices, denoted in the figure by the tag located on the left side of the classes rectangles and properties comprising parameters of the devices, denoted by the middle tags. Interconnections in the real plant domain are realized via ontology property called "power_bond", emphasizing that the interconnection transfers mainly energy. To reduce the size of the figure, only one real plant device is shown: a "resistor". The resistors are simulated by two generic simulation blocks, denoted by ontology property "simulates", that differ in their input and output signals. In a real case, the structure of both plant devices and generic simulation blocks is much more wider, as well as the signal taxonomy that involves only two electrical signals in this exemplary case. In addition, further ontologies are investigated to cover the whole automation system, such as "SCADA ontology" or "location ontology", exceeding this paper.

5 Automatical Ontology-Based Selection of Blocks

Formalization of knowledge, as it has been described in the previous section, enables to use automatic approaches supporting many engineering tasks efficiently. In this chapter, we propose an ontology-based approach to assign the simulation model according to the real industrial system structure, that addresses the research issue RI-4. The entry for our algorithm are the plant ontology, comprising the real plant structure and the simulation ontology, storing features of available generic simulation blocks. The goal of the automatical semantic engine is to find for each real plant device (i.e. individual of the plant ontology) an equivalent generic simulation block with respect on compatible interfaces of blocks interconnected in series. Although the engine assembles the structure of simulation model automatically, the simulation parameters are entered manually, hence the engine is referred as semi-automated.

To find appropriate simulation equivalents for each real plant device, there must be selected such blocks that simulate the particular devices according to the object property "simulates". By this selection, we obtain a set of candidates for each device of the plant topology. The next step is to select the appropriate blocks out of the candidates sets. This selection is driven by a rule being based on the signal-oriented point of view: *For each signal, an output interface of its producer must be equal to the input interface of its consumer.* Although this rule seems simple at first, it provides powerful support for the simulation model design. These ideas were expressed in a SPARQL query, being depicted in Fig. 3, whereas the results for the exemplary project are shown in Fig. 6.

The SPARQL query depicted in Fig. 3 is motivated by the following steps:

1. Find all ontology individuals that represent the real plant devices. These individuals are recognized by a relation "is_connected_to".
2. For each real plant device, find its simulation equivalents, i.e. generic simulation blocks that "simulates" it.

```
PREFIX ontology: <c:/Implementation#>
SELECT ?plant_device ?generic_block
WHERE {
 ?plant_device          ontology:is_connected_to ?plant_producer.
 ?plant_device          rdf:type                 ?plant_class.
 ?generic_block         ontology:simulates       ?plant_class.
 ?plant_producer        rdf:type                 ?plant_producer_class.
 ?producer_generic_block ontology:simulates      ?plant_producer_class.
 OPTIONAL {
 ?generic_block         ontology:input           ?interface1.
 ?producer_generic_block ontology:output         ?interface2.
 FILTER (?interface1 != ?interface2)
 } FILTER (!bound(?interface1))
}
```

Fig. 3. The SPARQL query to select appropriate generic simulation blocks from a universal simulation library.

3. For each device, find a producer of the signal being consumed by the device.
4. For each simulation block, find its input interface and an output interface of its input signal producer.
5. Select such simulation blocks, for which the interfaces are equal.

Note that the current version requires the equality of interfaces. We plan to generalize this approach in the further work in order to match types that can be transformed automatically, e.g., by conversion of units or scales.

Results of the SPARQL query depicted in Fig. 3 can be classified as follows:

1. *For each plant device exists exactly one simulation equivalent.*
   This solution is the most desirable; there exists exactly one simulation model of the industrial plant that is feasible and can be generated automatically.
2. *More than one solutions exist.*
   This class of solutions enables to create several simulation models for the industrial plants. Although all of them can be generated automatically, the engineer has to decide which of them is the most suitable for the task.
3. *A solution satisfying all conditions does not exist.*
   This result proves that the universal simulation library has to be modified.

6 Use-case: Simplified Electrical Circuit

To demonstrate the presented approach, a simplified electrical circuit was selected as a use-case. It consists of three devices: an ideal source of voltage, an ideal resistor, and an ideal capacitor. Although the methods are oriented on large-scale industrial systems, such a simple case enables humans to check the results and furthermore, it can be a sub-part of some complex cyber-physical system, i.e. a system comprising segments of diverse engineering disciplines. Fig. 4

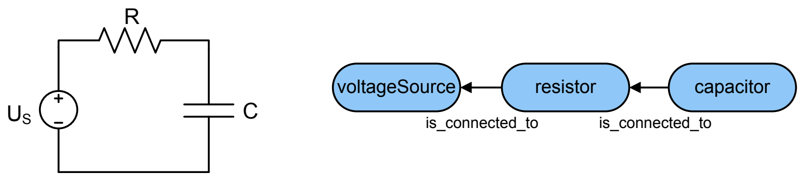

Fig. 4. Use-case electrical circuit: Schema and its representation with plant ontology individuals.

depicts the schema of the circuit on the left side and the adequate plant ontology individuals on the right side.

The output value of the ideal source of voltage is the constant voltage independent on the output current. The output of the ideal capacitor is voltage and the input is electric current. Although a capacitor could be also modeled inversely, having voltage as its input and electric current as the output, this model would not be causal, because the mathematical description would involve a derivation, hence this approximation is not considered. The simulation model of a resistor is much more interesting, because this element has no restriction, how to be simulated. The universal simulation library involves two resistors. *Resistor 1* has electric current as its input, voltage as output and R as a tunable parameter, having the transfer function $u = Ri$; whereas *Resistor 2* has voltage as input and electric current as output, i.e. its behavior is given by $i = u/R$.

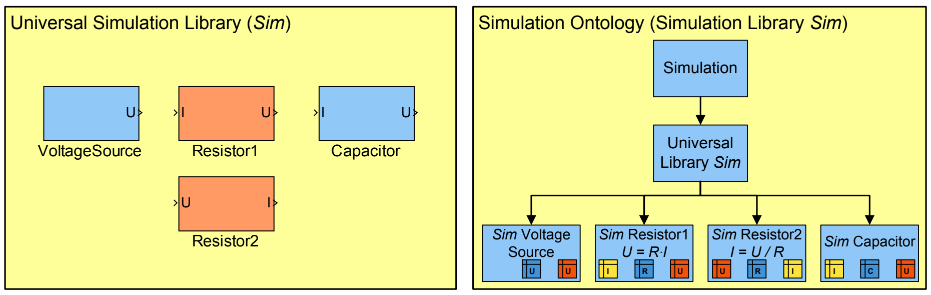

Fig. 5. Simplified simulation library with the dominant interfaces and representation of its semantics with simulation ontology.

The semantic information about available simulation blocks, their parameters, and interfaces are comprised in the simulation ontology, as depicted in Fig. 5. Nowadays, it is a simulation specialist who resolves which of the simulation blocks should be used. In such an extremely simplified example, it is very fast and easy to recognize that *Resistor 2* must be used for the simulation of this electric circuit, but in real large-scale systems, such decisions can be very time consuming, errors can occur and particularly extensions, proving the feasibility of a simulation model based on available simulation blocks are very valuable. The SPARQL query was used and its results are depicted in Fig. 6.

plant_device	generic_block
voltageSource	sim_voltage_source
resistor	sim_resistor2
capacitor	sim_capacitor

Fig. 6. Result of the SPARQL query. For each plant device the appropriate generic simulation block is selected.

7 Conclusion and Future Work

The paper deals with using ontologies and ontology querying to improve the design phase of simulation models as one of the important tools for optimization of automation systems and industrial plants operations. The presented approach is based on Semantic Web technologies providing efficient formalization of knowledge related to heterogeneous data-models in ontologies and enabling the usage of ontology-based methods, such as ontology querying or reasoning. Knowledge related to a structure of the real plant is represented in the plant ontology, knowledge about available simulation blocks is stored in the simulation ontology, and the taxonomy of interacting signals is formalized in the signal ontology. For various engineering tasks, a mapping between these ontologies plays an important role. The plant and simulation ontologies are mapped via the object property "simulates" that defines which real plant device is simulated by particular simulation blocks. The mapping between simulation and signal ontologies is realized by object properties "input" and "output" that define the interfaces of the blocks.

The proposed method is based on the selection of appropriate simulation blocks via the SPARQL query given in the paper. The query identifies a set of candidate generic simulation blocks that simulate each plant device. Consequently, the appropriate simulation block is selected out of the candidate set by matching the type of the input signal with the signal type of its producer. The explained method is demonstrated using the practical example dealing with a simplified electrical circuit. In this example, two generic simulation blocks for a resistor are available in the universal simulation library and it is demonstrated that the query finds the correct solution.

The ontology-based representation of the knowledge about industrial plants and the overall automation project provides flexible data representation, enables to use sound methods, such as ontology querying or reasoning, and can notably support both system design and maintenance. The early project stages dealing with the ontology structure design and definition could seem quite complex and time consuming, but the reusability and semantic representation enhances standardization and lower time and costs for maintenance or redesign. The main limitation of the proposed method is the current implementation for single-input and single-output blocks that are connected in series.

Future work will focus on the practical application of this approach in order to estimate its computational complexity for real large-scale systems. We plan further generalization to support multi-input and multi-output interfaces as well as the parallel interconnections of the blocks. We also plan to support identification of missing blocks and generation of their interfaces automatically. As it has been explained in the text, the automatic conversion between similar interfaces remains another future work issue.

8 Acknowledgments

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

References

1. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. Second printing edn. Springer-Verlag, London (2004)
2. Obitko, M., Mařík, V.: Ontologies for multi-agent systems in manufacturing domain. In: Proc. of 13th International Workshop on Database and Expert Systems Applications, 597 – 602 (2002)
3. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition 5(2) (1993)
4. Silver, G., Hassan, O.H., Miller, J.: From domain ontologies to modeling ontologies to executable simulation models. In: Proc. of the 2007 Winter Simulation Conference. 1108 – 1117 (2007)
5. Lastra, J.L.M., Delamer, I.: Ontologies for production automation. In: Dillon, T., Chang, E., Meersman, R., Sycara, K., eds.: Advances in Web Semantics I. Volume 4891 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg 276 – 289 (2009)
6. Moser, T., Mordinyi, R., Sunindyo, W., Biffl, S.: Canadian Semantic Web: Technologies and Applications, chapter Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constraints. Springer (2010)
7. Miller, J.A., Baramidze, G.: Simulation and the semantic Web. In: Proc. of the 2005 Winter Simulation Conference (2005)
8. Moser, T., Biffl, S.: Semantic tool interoperability for engineering manufacturing systems. In: Proc. of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA) (2010)
9. Cheng, S.-Y., Shen, B.-C., Peng, M.-J., Tian, Z.-F., Zhao, Q., Xue, R.-J., Gong, C. Research on coordinated control in nuclear power plant. International Conference on Machine Learning and Cybernetics, 2009, vol.6, pp.3622-3627, (2009)