DOMAIN ONTOLOGY OF MATERIALS IN RECONFIGURABLE MANUFACTURING SYSTEMS

ДОМЕЙН ОНТОЛОГИЯ НА МАТЕРИАЛИТЕ В РЕКОНФИГУРИРУЕМИ ПРОИЗВОДСТВЕНИ СИСТЕМИ

K. Stoyanov[1], D. Gocheva[2], I. Batchkova[2], G. Popov[1]

[1]Technical University of Sofia[1],
[2]University of Chemical Technology and Metallurgy - Sofia[2], Bulgaria
E-mail: kostadin_sto@abv.bg, dani@uctm.edu, idilia@uctm.edu, gepop@tu-sofia.bg

Abstract: This paper presents the developed domain ontology in order to be used as a knowledge base for materials in the field of reconfigurable manufacturing systems (RMS). Web Ontology Language (OWL 2) and Protégé as an editor and knowledge acquisition tool are used. The article explains the process of ontology development by using ontology reasoning and automated instance generation.

1. INTRODUCTION

In order to be competitive on the global market, the manufacturers need to satisfy the fast changing demands of the consumers. This leads to the development of new solutions for the development and manufacturing of the final product. One of the successful solutions is the development of reconfigurable manufacturing systems. These systems offer cost-effective and productive methods for faster responses to varying customer requirements and to ensure productivity growth and competitive responsiveness through automatically and dynamically reconfiguration on machine, control and system level. Reconfiguring provides adding up, removal and modifying of specific work processes, elements of management, software and machine structures. However, these reconfigurable manufacturing systems still have too many problems in respect to the required hardware and software that prevent their mass deployment on the world market. One of the main software problems is the interoperability issue that leads to the use of too many devices and the implementation of different software solutions in order to establish a connection and proper transfer of information between the different machines in the system. The results of this problem are increased manufacturing time and enhanced costs for the final product. Some of the interoperability problems may be solved through the combined use of standards, such as: IEC/ISO 62264, ISO 10303 (STEP) and IEC 61499 based on shared ontology.

In this paper an approach for developing an ontology of materials for reconfigurable manufacturing systems is presented. The approach is based on reasoning as a process for ontology development using Web Ontology Language (OWL 2) [1, 2] and Protégé and also an automated instance generation from relational databases and Excel tables based on RDBToOnto [3]. The purpose of the Materials ontology is to be used as a knowledge base for different materials and their properties. After the introduction a short overview of the applied techniques and tools is proposed in part 2 of the paper. Part 3 represents the developed domain ontology "Materials" where its basic components – classes, properties, data property assertions and instance generation are described. Finally, some conclusions are given.

2. OVERVIEW OF THE APPLIED TECHNIQUES AND TOOLS

Semantic web ontologies provide a common framework that promotes the sharing and reuse of data across different applications. They enrich data with metadata and support the development of software for processing by specifying the meaning of the data and allowing web based systems to exchange knowledge, using the advantages of software for intelligent logical analysis. In the literature reviews [4], [5] and [6] the relationship of semantic data models in connection with the relational model of database systems are analyzed; the common features and differences are systemized. Semantic data models have clear semantics, low connectivity, integration capabilities and interoperability, opportunities for automatic consistency checking, easy expandability and maintenance. Semantic data models are easy to understand and implement, as data queries are very close to natural language.

A. Rationale. Relational databases, object databases and data warehouses are widely acknowledged for storing information, but they have limited or no intelligence. The semantic databases are built on artificial intelligence principles, thus allowing intelligence to be gathered from the information. In the whitepaper [7] five reasons for using semantic data models are indicated. Semantic technologies embrace diversity of data models, enabling to use different data models; they provide uniformity of terms and harmonize data from different sources; semantic data models are created incrementally as information becomes available; they are 'lightweight' and 'flexible' and are able to leave data in its original repository and federate it on demand. Additionally, in [7] three features of semantic data models, which other data models do not possess, are given: semantic models allow to define the data structure which can be modified, when necessary, and also to make the analysis and verification of the logical structure of the model and the data at any time; semantic models allows us to create a layer on top of the native data sources and map them into a standard model.

B. Web Ontology Language OWL2. The Web Ontology Language OWL 2 [1] accepted as a World Wide Web Consortium (W3C) Recommendation from 11 December 2012

is a powerful knowledge representation language; it has been applied successfully for knowledge modelling in many application areas [2]. OWL 2 ontologies can be used along with information written in RDF, and OWL 2 ontologies themselves are primarily exchanged as RDF documents. As a descriptive language, OWL 2 is used to express expert knowledge in a formal way, and as a logical language, it is used to draw conclusions from this knowledge. An OWL2 ontology is a formal description of a domain of interest which consists of the following three different syntactic categories:

- *Entities*, such as classes, properties, and individuals, identified by IRIs;
- *Expressions* represent complex notions in the domain being described;
- *Axioms* are statements that are asserted to be true in the domain being described.

These three syntactic categories are used to express the logical part of OWL 2 ontologies, which means that they are interpreted under a precisely defined semantics allowing useful inferences to be drawn. *Entities* form the primitive terms of an ontology and constitute its basic elements. *Expressions* are statements that connect the *entities* detailing the description of the domain (complex concepts constructed by rich set of primitives – union, intersection, universal and existential quantification; number restrictions; enumeration of individuals, etc). *Axioms* are statements that are asserted to be true in the domain being described and allow relationships to be established between *expressions*: Subclass Axioms, Equivalent Classes, Disjoint Classes, Subproperties, Equivalent Properties, Disjoint Properties, Inverse Properties, Property Domain, Property Range, Inverse, Functional, Transitive Properties, etc. [1]. *Axioms* about individuals are called assertions. The *ClassAssertion axiom* allows one to state that an individual is an instance of a particular class; the *SameIndividual* assertion allows one to state that several individuals are all equal to each other, while the *DifferentIndividuals* assertion allows the opposite.

C. Reasoners. The ability to infer additional knowledge (deductive *reasoning* [2]) is of great importance for designing and deploying OWL ontologies. A particular kind of deductive reasoning on the *ClassAssertion* axiom, the task of computing the individuals that belong to a given class (or set of classes) is called *instance retrieval*. If the task is to find out whether one particular individual belongs to the given class, it is called *instance checking*. Analogous tasks exist for *SubClassOf* axioms: computing all subclass relationships between a set of classes is called *classification*, and checking a particular subclass relationship is called *subsumption checking* [2]. Very important reasoning task is *consistency checking*, the task of determining whether a class or an ontology is logically consistent or contradictory. Instance retrieval and classification tasks can be solved by using many individual instance and subsumption checks.

The concepts of *soundness* (all computed inferences are really entailed), *completeness* (all entailed inferences are really computed) and *computational complexity* (time and resources needed for a reasoning task) are very important for the choice of suitable reasoner. Lack of *completeness* is sometimes acceptable if it allows for simpler or more efficient implementations, but the lack of *soundness* is usually not desirable [2]. Sound and complete OWL 2 reasoning is of high complexity - double exponential computational complexity - N2ExpTime. The OWL 2 new profiles (OWL 2 EL, OWL 2 QL, OWL 2 RL) restrict the used syntactic categories to improve complexity and practical performance, however with limitation of expressivity. Normally the best balance between language expressivity and reasoning complexity depends on the intended application [8]. In [9] the performance of reasoning in HermiT is compared with that of FaCT++ and Pellet - two other popular and widely used OWL 2 reasoners. HermiT ontology reasoner supports all features of the OWL 2 ontology language, and it correctly performs both object and data property classification/reasoning tasks and is much faster than other reasoners.

D. Tool selection. The most popular free ontology editor is Protégé, which can be used with a variety of OWL reasoners. Popular systems for large parts of OWL 2 DL ($\mathcal{SROIQ}$) include FaCT++, HermiT, and Pellet. Quite often, the RDF/OWL ontologies might be automatically generated from legacy data sources such as spreadsheets, XML files and databases. The process of manually developing from scratch an ontology is difficult, time consuming and error-prone. Several semi-automatic ontology learning methods have been proposed, extracting knowledge from free and semi-structured text, documents, vocabularies and thesauri [10]. RDBToOnto [3] allows automatically generating OWL ontologies from relational databases and Excel tables.

3. DOMAIN ONTOLOGY "MATERIALS"

The developed ontology contains 450 *Entities* (classes, properties and individuals) defined as: 161 classes, 15 data properties, 8 object properties and 266 different individuals. In the ontology 3112 axioms are defined: 159 *SubClassOf* axioms, 127 *EquivalentClasses* and 7 *DisjointClasses* are defined. In addition, for Data property axioms: there are 13 *SubDataPropertyOf* axioms, 13 *FunctionalDataProperty* axioms, 2 *DataPropertyDomain* axioms and 13 *DataPropertyRange* axioms. For individuals defined axioms are: 266 *ClassAssertion* axioms and 2000 *DataPropertyAssertion* axioms. The ontology is created in the free ontology editor Protégé 4.3 and the three of the most popular reasoners – Hermit 1.3.8.413, Pellet 2.2.0 and Fact++ 1.6.3 are used in the process of development. RDBToOnto is used for easy and fast automated instance creation.

A. Classes. There are three main classes in the ontology – *Final*, *Ores* and *Recycled*. The *Final* class contains a collection of classes and individuals that describe materials which are ready to be used in different processes. The *Ores* and *Recycled* classes contain materials that are ores and need additional processing and materials that could be recycled respectively. This paper presents only the *Final* class (Fig.1), as for the other two classes the structure and principles applied are the same. The first level of sub classes of the *Final* class consists of named classes that allocate the different defined classes and individuals into their main categories – *Composites*, *Isolators*, *Liquids*, *Mastics*, *Metal* and *Alloys*. The *Metal* class and the *Alloys* class are the biggest and most complex of all named classes so they will be described in more details. The first level of sub classes for the *Alloys* class consists of named classes that make more precise distribution of the individuals into their respective categories. For example, the sub classes of the *Alloys* class are dividing the individuals into following categories: *AluminiumAlloys*, *SteelAlloys_QuenchingTempering*, *SteelAlloys_Bearings*, *SteelAlloys_Carburizable* and *CopperAlloys*. Each one of these classes contains a collection of defined classes that through the use of different restrictions could enable sorting operations and allocate the individuals according to the specific needs. As shown in Fig.2 the hierarchy of the defined classes is flat before the use of reasoners. When the reasoners are used, the hierarchy is changed and the individuals are allocated according to the applied restrictions (Fig.2). The first subclasses of the *Metals* class are *FerrousMetals* and *NonFerrousMetals*. The *NonFerrousMetals* class contains collection of individuals that represent metals like gold, zinc, titanium and so on, while the *FerrousMetals* class has

additional two levels of sub classes that add additional detail before the individuals are being asserted to the appropriate class. The first sub level consists of the named classes *CastIron* and *Steels*. Each of these classes contains another sub level of named classes that add additional classification before the individuals are being asserted. For example, the *CastIron* class is divided to ductile cast iron, gray cast-iron and soft cast iron, while the *Steels* class is divided to sheets, high quality steels and steels that are of normal quality. All other classes of the ontology are with simpler hierarchy and have only one or zero levels of nested classes before the assertion of the individuals.

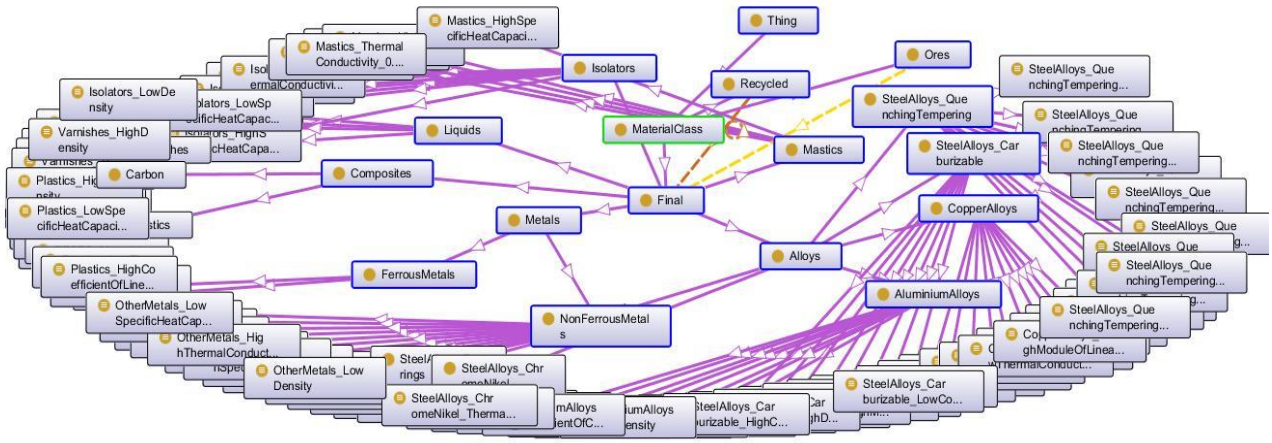

Fig.1: Final class hierarchy

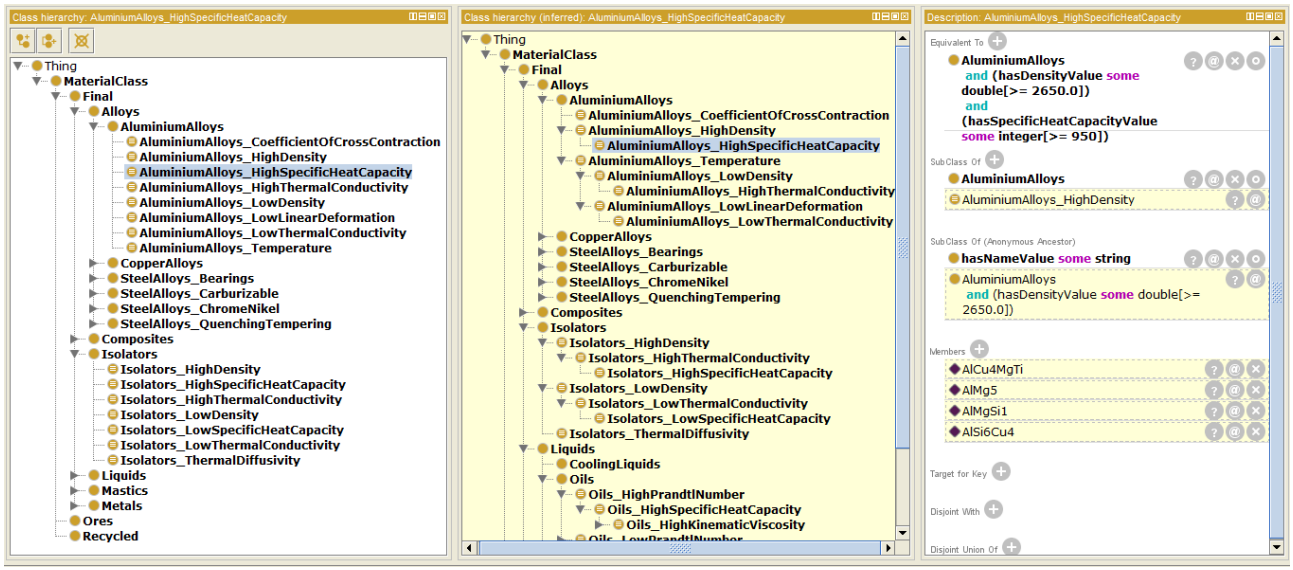

Fig.2: Asserted and inferred class hierarchies

B. Properties. There are two types of properties – object properties and data type properties. The object properties are used to describe the relations between the different individuals. There are eight object properties in the ontology divided into two major groups. The first group contains properties that are used to describe the relations between the different individuals of the Material ontology and it is named *innerProperties*. The object properties of this group are *participateInComposition*, *isRecycledFrom* and *isExtractedFrom*. The first property is used to describe the relations between the individuals of the *Ore* class and those from the *Final* class. For example, the individuals that describe different kinds of iron ores could be linked to individuals that represent different steels and cast irons. The second object property - *isRecycledFrom* - is used to link the individuals from the *Final* class to the individuals of the *Recycled* class. For example, golden chips could be recycled from the processing of golden plates. The third inner property *isExtractedFrom* doesn't have a particular range or domain, but it is used to the extent of the whole ontology. This property is used to describe the relations between individuals that represent materials that could be extracted from other materials. The second group of object properties is used to describe the relations between the different individuals of the Material ontology and all entities of the other three ontologies. The group has been named *outerProperties* and contains the next three object properties – *materialIsUsedInProcess*, *materialIsUsedInEquipment* and *materialIsUsedInPart*. The first one is used in the domain ontology for manufacturing processes and technological operations by linking the materials to the appropriate processes. The property *materialIsUsedInEquipment* is used in the domain ontology for the equipment in the manufacturing systems. This property links the individuals, who represent different technological equipment to the materials, they are made of. The third object property (*materialIsUsedInPart*) is used in the last of the four domain ontologies to link the appropriate material to the desired part.

The data properties are used to describe the characteristics of the different individuals. There are 15 data properties in the ontology. One data property is used for the names of the materials and the other 14 are sub properties of the property *MaterialPropertyValues* and are used to describe different thermal properties of the appropriate individual such as maximum or minimum working temperature, viscosity (for liquids and gases), specific heat capacity, Prandtl number, module of linear deformation and so on.

C. DataPropertyAssertion. An example is made with the named class *AluminiumAlloys*. This class contains 8 defined classes and all individuals that represent aluminum alloys. As shown in Fig.2, when the ontology has been inferred, the classes are rearranged and there are some differences in the hierarchy. The first defined class in Fig.2 - *AluminumAlloys_CoefficientOfCrossContraction* – has the existential restriction (1). This means that this class contains all aluminum alloys that have value for the coefficient of cross contraction exactly 0,34.

```
AluminiumAlloys                                                    (1)
  and (hasCoefficientOfCrossContractionValue value 0.34)
```

The next class that is presented on Fig.2 is *AluminumAlloys_HighDensity* and has the restriction (2). The class *AluminiumAlloys_HighSpecificHeatCapacity* has the existential restriction (3). The first class collects all aluminum alloys that have density value bigger or equal to 2650 [kg/m^3], and the second class add to that restriction another one that states that each individual has to have density value bigger or equal to 2650 [kg/m^3], and specific heat capacity value bigger or equal to 950 [J/kgK] in order to be a member of that class. So when the ontology is inferred the *AluminiumAlloys_HighSpecificHeatCapacity* class is becoming a sub class of the *AluminumAlloys_HighDensity* class. If another restriction is added to the existing one, and this new set of restrictions is then applied to another class then the new class is going to be sub class of the *AluminiumAlloys_HighSpecificHeatCapacity* and the individuals will have to pass additional screening in order to be members of the new class. Following this method, the number of individuals that are members of a particular class could be reduced to one out of thousands. So when the ontology is inferred the deepest nested class is related to the steel alloys for quenching and tempering and it is the eight hierarchical levels beneath the *Final* class.

```
AluminiumAlloys                                                    (2)
  and (hasDensityValue some xsd:double[>= "2650.0"^^xsd:double])

AluminiumAlloys                                                    (3)
  and (hasDensityValue some xsd:double[>= "2650.0"^^xsd:double])
  and (hasSpecificHeatCapacityValue some xsd:integer[>= 950])
```

D. Instance Generation. Some of the instances (individuals) in the developed ontology are created through the use of the RDBToOnto software, which allows the creation of ontologies from relational databases. As input data, Excel tables are used, which contain information about the values of the data properties. It should be noted that, despite the exceptional usefulness of the software, there are situations in which the values of the types double and integer are not recognized by the used ontology editors as such.

In the process of ontology design, the three most popular open-source reasoners are used: FaCT++, HermiT, and Pellet. Experiments performed with different reasoners show that for the same reasoning task, the reasoning time consumed is comparable and the classification results were the same. All three reasoners discover inconsistency, due to the lack of coincidence in the datatypes obtained in the process of generating instances.

4. CONCLUSIONS

The developed domain ontology, discussed in this paper aims to represent knowledge about materials used in various ways and for different purposes in the domain of machine tools and systems. The ontology is one of the four ontologies (Materials, Equipment, Parts and Processes), that are being developed to be integrated with the meta-ontology, based on the IEC/ISO 62264 standard in order to work together. The main purpose of the development of the above mentioned ontologies is to capture the basics of the manufacturing processes in order to solve different reconfiguring tasks. One very important task solved in this direction is the generation and online reconfiguration of control programs for reconfigurable manufacturing systems based on the IEC 61499 standard and using specially developed software. Fundamental merit to this has the approach for solving the interoperability issue in reconfigurable manufacturing systems. One other benefit of the suggested domain ontology is that it could be used as a standalone knowledge base for materials and may be continuously extended with new materials, properties and data.

REFERENCES

1. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition) W3C Recommendation 11 December 2012, https://www.w3.org/TR/owl2-syntax/
2. Krötzsch M. (2012), OWL 2 Profiles: An Introduction to Lightweight Ontology Languages, Proc. 8th Reasoning Web Summer School, LNCS 7487, Springer, pp. 112–183.
3. Cerbah F. (2008), Learning highly structured semantic repositories from relational databases: the RDBToOnto tool, In Proceedings of the 5th European Semantic Web Conference, Spain.
4. Motik B., Horrocks I., Sattler U. (2009). Bridging the gap between OWL and relational databases. Web Semantics: Science, Services and Agents on the World Wide Web, 7(2), pp.74-89.
5. Horrocks I. (2008). Ontologies and databases. Presentation at the Semantic Days.
6. Uschold M., Arts S. (2011). Ontologies and Database Schema: What's the Difference? In Semantic Technology Conference, San Francisco, CA.
7. Funnell J., Dickson D., Chacon J. (2012), 5 reasons to use semantic modelling & bring your plant data together, Honeywell Process Solutions, whitepaper, February, Honeywell International Inc.
8. Krötzsch M., Simančík, F., Horrocks, I., Description Logics. In IEEE Intelligent Systems, volume 29:1, pp.12–19. IEEE 2014.
9. Glimm B., Horrocks I., Motik B, Stoilos G., WangZ., HermiT: An OWL 2 Reasoner, Journal of Automated Reasoning, October 2014, Volume 53, Issue 3, pp 245-269
10. Spanos D. E., Stavrou P., Mitrou N. (2012), Bringing relational databases into the semantic web: A survey. Semantic Web, 3(2):169-209.