TOWARDS INDUSTRIAL STRENGTH KNOWLEDGE BASES FOR PRODUCT LIFECYCLE MANAGEMENT

Abstract

The benefits of using semantic technologies for industrial applications are becoming more and more apparent. While their general usefulness is widely acknowledged, the uptake of technologies originally developed for the semantic web is complicated by the mismatch between industrial requirements and design principles of standards such as OWL. In this paper, we argue for the need of a flexible, metamodel-driven approach to address non-functional requirements for the use of knowledge bases in industrial settings and present a lightweight metamodel for ontologies and rules.

Keywords: ontology, mapping rules, product lifecycle management, metamodel.

1 MOTIVATION

Semantic technologies originally developed for knowledge representation and reasoning on the web are gaining attention from industry as an adequate means for solving complex information management tasks. A constantly growing number of companies that offer professional services or tools in this area reflect this trend (Davis 2006). Typical applications of ontologies and rule bases include knowledge and skill management (Staab 2002) as well as web service and business process management (Fensel 2006). Studies (Hahn 2005, Hahn 2007), prototypes (Stegmüller 2003, Küsters 2006), and first successful projects (Syldatke 2007) emphasize the relevance of these technologies in the area of product lifecycle management (PLM), in particular in the automotive sector. As such projects are getting bigger and hence the value of the created ontologies and rules sets, new challenges are emerging for ensuring the successful adoption of these technologies in industry. While theoretical issues like expressiveness and decidability still remain fundamental, non-functional aspects like knowledge base maintainability and security are getting crucial too (cf. Hepp 2007). Providing support for developing and managing industrial strength knowledge bases is therefore a key success factor, especially if we want to avoid the fallacies of early knowledge based systems that often did provide the functionality needed for a given task but failed to integrate smoothly into the general IT infrastructure and policy of companies (cf. Bobrow 1986).

In this work, we analyze requirements for industrial strength knowledge bases with a focus on PLM, one of the major challenges in today's automotive industry. In particular, we argue for the need of a model-driven approach for ensuring interoperability and providing a solid basis for developing and managing large knowledge bases. In the rest of the paper, we present a metamodel for a lightweight and pragmatic integration of ontologies and rules, which can be used as a formal foundation for implementing development and management methodologies. In this context, we first develop an integrated model of ontologies and rules in section 5 and present the corresponding metamodel in section 7. We close with a description of a metamodel-based management environment for industrial strength knowledge bases that is currently under development.

2 APPLICATION OF SEMANTIC WEB TECHNOLOGIES IN PLM

Today, automotive development is characterized by ever changing markets, growing product diversification, increasing functional complexity, sharpening quality demands, shortening development life cycles, and tightening regulatory requirements. Optimizing the development process, which covers all the activities from the planning of technical and organizational aspects to physical tests including the optimization of the assembly processes, is an essential strategic task in order to cope with these challenges (cf. Bullinger, Kiss-Preußinger and Spath 2006).

In the past, many IT systems have been built to support single and focused aspects within the product development process (e.g., car conception, virtual simulation, part tracking), each of them maintaining their own models, structures, and semantics about the products and their parts. Nevertheless, all systems do contribute data and information to the final product, i.e. the actual car to be built for the customer. In order to enable flexible changes and adoptions in such a grown infrastructure – especially from a business point of view, there is a strong need to share and exchange such heterogeneous product models. This aspect is addressed by PLM, which is further defined as (cf. CIMdata 2007):

- a strategic business approach that applies a consistent set of business solutions that support the collaborative creation, management, dissemination, and use of product definition information,
- supporting the extended enterprise (customers, design and supply partners, etc.),
- spanning from concept to end of life of a product or plant,
- and integrating people, processes, business systems, and information.

One way to achieve the model integration is to adopt standards for data exchange like ISO 10303 - also known as STEP (standard for the exchange of product model data). Since a standard can hardly take all particularities of a specific company and its development process into account, there will always be limitations in such a solution, which may result in lacking user acceptance. Furthermore, such an approach that has effects on a great number of existing IT systems would not only be time-consuming but also costly. Alternative solutions that can handle the heterogeneous models without the need to change existing data are therefore strongly demanded.

Technologies used for the semantic web provide promising means for integrating different product models. Ontologies can be used to describe and formalize the existing product models and structures. However, descriptive ontologies are not sufficient to express all semantic relationships between such formal models (e.g., unit conversion, or instance creation, cf. Stuckenschmidt and Uschold 2005). Rules, which are also part of the semantic web technology stack, can provide the required expressiveness, and are thus an important means to overcome with the limitations of ontologies (cf. Jones and Uschold 2005, Angele and Gesmann 2007).

Having a semantically connected network of ontologies that covers all aspects of the development of a product not only allows for sharing and exchanging product data between the used IT systems, but also provides a powerful platform the implement further methodologies for PLM. A subject of an ongoing work is the creation of a business ontology that enables a business oriented point of view on the currently highly IT-driven development process at a German car manufacturer. By connecting such an ontology to the product models, we will be able to analyze the resulting dependencies and thus track the effects of specific decisions in all phases of the product development process. This approach may further provide very valuable information for a more efficient process management. Also, the use of such a business ontology reduces the number of required mappings from n^2, when mapping the models with each other, to n, when only mapping the models to the business ontology.

Apart from semantic integration there are a number of other use cases in the context of PLM, which could also benefit from industrial-strength knowledge bases using ontologies and rules. These include:

- application of ontologies and rules to verify product data during the development process (Syldatke 2007),
- validation and assessment of the product development process based on the product models using metrics (Hahn 2007),

3 REQUIREMENTS FOR KNOWLEDGE BASES

Within this work we focus on the use of ontologies and rules for the semantic integration of product models. We thereby use ontologies to formalize existing data sources and rules to define mappings and translations in order to integrate them with a global business ontology. In the following section we define key requirements for such a knowledge base based on experiences with first prototypes implemented in that context:

- **Reuse of Existing Business Data:** The majority of today’s product data resides in relational database management systems (RDBMS), which provide highly optimized CRUD (create, read, update, and delete) functions on the stored data. Furthermore database schemata already contain valuable semantic information. In order to minimize the efforts and costs for reproducing such existing knowledge, an essential requirement is the direct and transparent reuse of such information within our knowledge base. Furthermore, we can avoid synchronization issues, because we do not need to duplicate any data from RDBMS.
- **Performance and Scalability:** Developing a car is a highly complex and knowledge-intense challenge. Every single step within the development process requires but also generates a great amount of data and information like product definitions, part geometries, or product structures. Addressing aspects within the product development using ontologies and rules therefore requires an infrastructure that is able to deal with great data amounts in terms of performance and scalability.

- **Change Management:** Knowledge is subject to constant changes -- especially in dynamic environments like the automotive development. Appropriate evolution support for knowledge bases is therefore a critical aspect for industrial applications. A major requirement is to ensure a constantly high quality of the developed knowledge bases. Depending on the addressed business use case such models could rapidly grow in their size and thus there is a strong demand for automated methodologies to detect possible logical and modelling errors.
- **Reusability and Interoperability:** As sketched in the last section, an integrated network of product model ontologies could be used as a foundation for addressing additional aspects within the product development process. Therefore, it is important to ensure that such a knowledge base is easily reusable for future applications. Also, there must not be any dependencies to proprietary representation formalisms and thus to a specific vendor. These aspects increase the return on investment (ROI) due to a longer service life and relevance of such a knowledge base provided that it meets all functional requirements.
- **Security:** Today, the development of a car is a highly distributed task. A great number of employees - either within the same company or the corporate group – and external staff from partners and suppliers are involved in the development process. A knowledge base covering the whole development process needs to be protected against security threats. Ensuring authenticity, integrity, confidentiality, availability, and non-repudiation for industrial knowledge bases are therefore essential requirements.
- **Enabling Rule-based Mappings:** Finally, we need to be able to represent various kinds of mappings between the business ontology and the different product models. While simple mappings can be expressed using ontology axioms (e.g., equality), complex ones like translation, conversion, and derivation of data require the use of a rule language. Jones and Uschold (2005) list some common mapping use cases based on the work of Garcia-Solaco et al. (1995), which also apply and are therefore required in the context of PLM.

4 SYSTEM DESIGN GOALS

In the following section we define the major design goals for industrial strength knowledge bases in the context of PLM in the automotive sector, which we derive from the requirements stated in the last section and on which we will focus in our future research:

- **Compatibility with existing Standards:** Standardization has proven to be a key success factor concerning the uptake of technologies in enterprises. Since its invention the Web Ontology Language OWL has experienced a wide spread and has become the most often used knowledge representation language in the world, especially outside academia. For an industrial setting, standardization is important to guarantee interoperability between different systems and also ensures independence of a specific product for handling the knowledge. However, a standard rule language for the semantic web does not exist today, although there are several proposals to develop one (e.g., SWRL, WRL, F-Logic). Currently, the most promising work is provided by the Rule Interchange Format (RIF) working group at the W3C.
- **Lightweight Ontologies and Rules:** As argued in Calvanese et al. (2006), OWL itself is not suited for the use in the context of ontology-based data access and thus for data integration. In fact, the expressiveness of the representation formalism needs to be restricted to guarantee efficient and scalable reasoning w.r.t. data complexity. In order to meet the requirements regarding performance and scalability we initially trade off expressiveness for the ability to handle great amounts of product data. However, we must ensure the extensibility of such a foundation for being able to add more expressive language features in the future. These aspects also apply for the rule language.

From the defined design goals, we conclude that we need a uniform foundation to meet all the given requirements. A model-driven approach is therefore the most suitable solution. Due to missing standards for lightweight ontologies and rules, we need an independent formal foundation to address industrial-related aspects of such knowledge bases. A metamodel allows us to tailor existing representation languages (e.g., OWL) to the specific requirements of the application domain. We can further specify the requirements for a rule language for ontology mappings without the need to commit to any specific formalism. Depending on the used infrastructure we can finally transform the actual models to the required representation formalisms. This ensures a high level of interoperability between the system components. Finally, the metamodel also provides a stable foundation for realizing and supporting the development and management of such knowledge bases.

In the following, we present a metamodel for a practical integration of ontologies and rules that meets the representation and reasoning requirements of industrial applications in the context of PLM.

5 ESTABLISHING AN EXTENSIBLE FOUNDATION

Integrating ontologies and rules is a non-trivial research problem, which is reflected by the number of publications on that topic. A good overview about the current state of research can be found in Antoniou et al. (2005). In this work we aim at addressing development and maintenance issues of these models in an industrial setting rather than providing another solution for combining ontologies and rules. However, we do need a formal foundation to start off with our research in that context. In order to provide future-proof methodologies and solutions, we need to ensure independence to proprietary representation formalisms. In particular, any foundation must obey existing standards and be extensible with respect to future ones.

Description Logics ($\mathcal{DL}$) and Logic Programming (LP) are the two common formalisms for representing ontologies and rules respectively in the semantic web. We therefore propose to use the well-known intersection between $\mathcal{DL}$ and LP -- i.e. Description Logic Programs (DLP, cf. Volz 2004) - as an initial foundation for the investigation of aspects regarding the industrial use of ontologies and rules. In DLP, $\mathcal{DL}$ axioms are transformed into a conjunction of Horn formulae, which could be translated into different LP languages. Rules can easily be added using the target LP language.

Obviously, the expressiveness of DLP is limited. Nevertheless it provides a good, pragmatic starting point, which hardly narrows the adoption of future scientific findings on combining ontologies and rules. Hitzler et al. (2005) discuss some of the controversies concerning the DLP fragment of OWL and argue why it is nevertheless a feasible way to go.

5.1 Lightweight Ontologies

Since we need to ensure a scalable infrastructure, we build on a subset of DLPs that can be processed in plain Datalog, for which sound, complete, and efficient evaluation strategies are well known. De Brujin, Pollers and Fensel (2004) have presented such a subset - i.e. OWL Lite$^-$. Since we need to represent existing data from RDBMS we add the support of primitive and single ranged data types to this subset. The abstract syntax and semantics of data types are already defined in OWL Lite and therefore this extension is trivial. However, this feature needs to be supported by the used Datalog system. We finally remove minCardinality(0) because it has no real use (de Brujin et al., 2005, p. 28). The resulting subset will be called OWL Lite$^{-,p}$. Table 1 gives an overview over the included features and defines the mapping to Datalog as well as to F-Logic. Note that the commercially available OntoBroker supports the LP fragment of F-Logic and primitive datatypes. Therefore it is currently a reasonable target LP system – especially in industrial settings. Another relevant system is the IRIS reasoner (http://iris-reasoner.org).

OWL Abstract Syntax	DL Syntax	F-Logic Syntax	Datalog Syntax
restriction(R allValuesFrom(C))*	$\forall R.C$	see partial class definitions	
Class(A partial $C_1 \ldots C_n$)	$A \sqsubseteq C_i$	$\bigwedge^1 A :: C_i$ $\bigwedge^2 x_1 : A \wedge x_1[R_i \twoheadrightarrow x_2] \rightarrow x_2 : C_i$	$\bigwedge^1 A(x) \rightarrow C_i(x)$ $\bigwedge^2 A(x_1) \wedge R(x_1, x_2) \rightarrow C_i(x_2)$
Class(A complete $C_1 \ldots C_n$)	$A \equiv C_1 \sqcap \ldots \sqcap C_n$	$\bigwedge \begin{cases} A :: C_i \\ C_i :: A \end{cases}$	$\bigwedge \begin{cases} A(x) \rightarrow C_i(x) \\ C_i(x) \rightarrow A(x) \end{cases}$
EquivalentClasses($C_1 \ldots C_n$)	$C_1 \equiv \ldots \equiv C_n$	$\bigwedge_{i \neq j} C_i :: C_j$	$\bigwedge_{i \neq j} C_i(x) \rightarrow C_j(x)$
ObjectProperty(R			
super(R_1) ... super(R_n)	$R \sqsubseteq R_i$	$\bigwedge x[R \twoheadrightarrow y] \rightarrow x[R_i \twoheadrightarrow y]$	$\bigwedge R(x,y) \rightarrow R_i(x,y)$
domain(C_1) ... domain(C_n)	$\top \sqsubseteq \forall R^-.C_i$	$\bigwedge x[R \twoheadrightarrow y] \rightarrow x : C_i$	$\bigwedge R(x,y) \rightarrow C_i(x)$
range(C_1) ... range(C_n)	$\top \sqsubseteq \forall R.C_i$	$\bigwedge x[R \twoheadrightarrow y] \rightarrow y : C_i$	$\bigwedge R(x,y) \rightarrow C_i(y)$
[inverseOf(R_0)]	$R \equiv R_0^-$	$\bigwedge \begin{cases} x[R \twoheadrightarrow y] \rightarrow y[R_0 \twoheadrightarrow x] \\ x[R_0 \twoheadrightarrow y] \rightarrow y[R \twoheadrightarrow x] \end{cases}$	$\bigwedge \begin{cases} R(x,y) \rightarrow R_0(x,y) \\ R_0(x,y) \rightarrow R(x,y) \end{cases}$
[Symmetric]	$R \equiv R^-$	$x[R \twoheadrightarrow y] \rightarrow y[R \twoheadrightarrow x]$	$R(x,y) \rightarrow R(y,x)$
[Transitive])	Trans(R)	$x[R \twoheadrightarrow y] \wedge y[R \twoheadrightarrow z] \rightarrow x[R \twoheadrightarrow z]$	$R(x,y) \wedge R(y,z) \rightarrow R(x,z)$
SubPropertyOf(R_1 R_2)	$R_1 \sqsubseteq R_2$	$x[R_1 \twoheadrightarrow y] \rightarrow x[R_2 \twoheadrightarrow y]$	$R_1(x,y) \rightarrow R_2(x,y)$
EquivalentProperties($R_1 \ldots R_n$)	$R_1 \equiv \ldots \equiv R_n$	$\bigwedge_{i \neq j} x[R_i \twoheadrightarrow y] \rightarrow x[R_j \twoheadrightarrow y]$	$\bigwedge_{i \neq j} R_i(x,y) \rightarrow R_j(x,y)$
DatatypeProperty(U			
domain(C_1) ... domain(C_n)	$\top \sqsubseteq \forall U^-.C_i$	$\bigwedge x[U \twoheadrightarrow y] \rightarrow x : C_i$	(not supported)
range(D))**	$\top \sqsubseteq U.D$	$C_i[U \Rightarrow D]$.	(not supported)
Individual(o type(C_1) ... type(C_n)	$o \in C_i$	$\bigwedge o : C_i$	$\bigwedge C_i(o)$
value(R_1 o_1) ... value(R_n o_n)	$\langle o, o_i \rangle \in R_i$	$\bigwedge o[R_i \twoheadrightarrow o_i]$	$\bigwedge R_i(o, o_i)$
value(U_1 v_1) ... value(U_n v_n)	$\langle o, v_i \rangle \in U_i$	$\bigwedge o[U_i \twoheadrightarrow v_i]$	(not supported)

* only allowed in partial class definitions
** only primitive datatypes allowed

1: for named classes
2: for universal class restriction

Table 1. Features included in OWL Lite$^{-,p}$

5.2 Lightweight Rules

As mentioned in section 3, Jones and Uschold (2005) list main mapping cases between ontologies. We consider following ones as most important in the context of PLM:

- Class and property name mapping; class and subclass mapping; instance mapping
- Coded data translation (e.g., Boolean value encoded as '1'/'0' or 'Y'/'N' in an Oracle RDMS); scale and unit translation and data derivation (e.g., convert the engine power from KW to PS)
- Conversion between property with cardinality n, containers and collections (e.g., split an aggregated part to its single parts)

From this list, we conclude that the ability to refer to ontology elements (e.g., classes, properties, individuals) and to perform data translation, conversion, and derivation in a mapping rule are the main requirements for a rule language. For defining an appropriate rule language for ontology mappings we refer to the latest Core Horn Rules definition of the Rule Interchange Format (RIF) working group at the W3C (Boley et al. 2007). This allows us to maintain a significant level of interoperability to other rule languages. Compared to that definition, we disallow following features to reduce the complexity of the resulting rule language:

- Disjunctions: They can be eliminated by applying the tautologies $(h \leftarrow b \vee c) \equiv (h \leftarrow b) \wedge (h \leftarrow c)$ and $\forall x(F \wedge G) \equiv \forall xF \wedge \forall xG$ (cf. Boley and Kiefer 2007).
- Nested terms: They can be split to several non-nested terms.
- Equality and Functions: Both features are not part of Datalog.

Conversely, we add built-in predicates for enabling data conversion, translation, and derivation in complex mapping rules. Furthermore, we define following constructs for referring to ontology elements:

- Object Classification: States that an object term X (i.e. individual, object variable, or function) is member of a specific class C.
- ObjectProperty Reference: States that an object property R references an object term Y. The domain of the property is defined by an object term X.
- DatatypeProperty Reference: States that a data type property U references a data term Y (i.e. data variable, data value). The domain of the property is defined by an object term X.

Note that the resulting rule language goes beyond plain Datalog. Nevertheless both, Ontobroker and IRIS Reasoner, do support the required features.

6 AN EXAMPLE FROM THE AUTOMOTIVE SECTOR

Consider two ontologies in the context of PLM – one describes requirements towards specific part classes, whereas the second one provides detailed information about concrete parts (e,g., their availability):

```
Namespace(req = http://www.example.org/requirements#)

Class(req#PartClass partial)

Class(req#Requirement partial)

Class(req#WeigthRequirement partial req#Requirement)

ObjectProperty(req#requires domain(req# PartClass) range(req#Requirement)

DatatypeProperty(req#MaximumWeigth domain(req#WeightRequirement)
range(xsd:decimal))

Namespace(par = http://www.example.org/parts#)

Class(par#PartClass partial)

Class(par#Part partial)

ObjectProperty(par#belongsToPartClass domain(par#Part) range(par#PartClass))

DatatypeProperty(par#hasWeigth domain(par#Part) range(xsd:decimal))
```

Both ontologies share the same information about part classes. We now intend to connect all parts that fulfil the defined requirements for specific part classes. We therefore define following mapping ontology:

```
Namespace(rp = http://www.example.org/requirements_parts#)

EquivalentClasses(req#PartClass par#PartClass)

ObjectProperty(rp#fulfilsRequirement domain(par#Part) range(req#Requirement))
```

Obviously, a part fulfils the weight requirements, if it belongs to the same part class and its weight is less than the defined maximum one. However, both ontologies use different units to define the weight – we assume grams for the requirements ontology and kilograms for the parts one. Such a complex mapping can only be formalized using rules:

```
rp#relevantPart(P,R) <- req#requires (C,R), par#belongsToPartClass(P,C).

rp#fulfilsRequirement(P,R) ← rp#relevantPart(P,R), req#WeigthRequirement(R),
req#MaximumWeigth(R,M), par#hasWeigth(P,W), conversion(kilo, grams, C), multiply(M,
C, M1), less(W, M1).

conversion(kilo, grams, 100). conversion(kilo, pounds, 2). conversion(kilo, ounces, 32).
```

7 AN INTEGRATED METAMODEL

The interdependencies between lightweight ontologies and rules are an essential part of such formal models. An integrated metamodel of both, which explicitly includes such links, is therefore an important foundation with respect to maintaining industrial strength knowledge bases.

7.1 Related Metamodels

In the context of ontologies and rules 3 metamodels -- all based on the Meta-Object Facility (MOF), the Object Management Group (OMG) standard for defining such models -- are relevant:

- Ontology Definition Metamodel (ODM, IBM and Sandpiper Software Inc. 2006): The ODM specification has been submitted by IBM and Sandpiper Inc. in respond to a Request for Proposals (RFP) of the Object Management Group (OMG, Object Management Group 2003) and is currently undergoing the finalization phase within the OMG technology adoption process. The metamodel covers OWL Full and Common logic. An ODM implementation -- the EMF Ontology Definition Metamodel (EODM) - is provided by the Eclipse Modeling Project and is used for example by IBM's Integrated Ontology Development Toolkit (IODT).
- Networked Ontology Model (Haase et al. 2006): This ontology metamodel is being used within the NeOn project. It is restricted to OWL DL because of the computational issues of reasoning with OWL Full. It further provides modules for SWRL rules, ontology mappings, and modular ontologies (cf. Haase et al. 2006, p. 4).
- REWERSE Rule Markup Language (R2ML): R2ML has been developed by the REWERSE Working Group I1. It aims at integrating the Object Constraint Language (OCL), the Rule Markup Language (RuleML), and the Semantic Web Rule Language (SWRL). R2ML supports four different types of rules: derivation, production, integrity, and reaction rules. The vocabulary metamodel is driven by the different rule languages.

The standardization progress of the ODM specification is far advanced. First implementations of the specification and tools are already available. Due to the support of influential players of the branch (e.g., IBM, HP, and AT&T) we expect significant efforts and contributions from these parties. In order to benefit from these results, an integrated metamodel of ontologies and rules should directly reuse relevant parts of the ODM. A main issue however is the complexity of the ODM metamodel, which mainly results from the support of OWL Full. This has been one of the reasons for the development of the Networked Ontology Model. Its design, however, strongly differs from the ODM metamodel. A pragmatic solution for designing a metamodel for lightweight ontologies is therefore to cut down the ODM specification to the required parts. When extending the supported OWL subset in a later step, it is easy to re-add the required constructs from the ODM without the need of any basic design change.

Because the R2ML metamodel integrates a number of different rule types and languages, it is highly complex even when limited to derivation rules. Furthermore, the vocabulary part of the metamodel does not comply with the ODM. The SWRL module of the Networked Ontology Model does not cover the required language features the lightweight rule language, whereas the Common Logic metamodel included in the ODM is far too powerful and thus complex. Therefore, we will develop a novel rules metamodel, which is fully integrated with the ontology metamodel.

7.2 Lightweight Ontology Metamodel

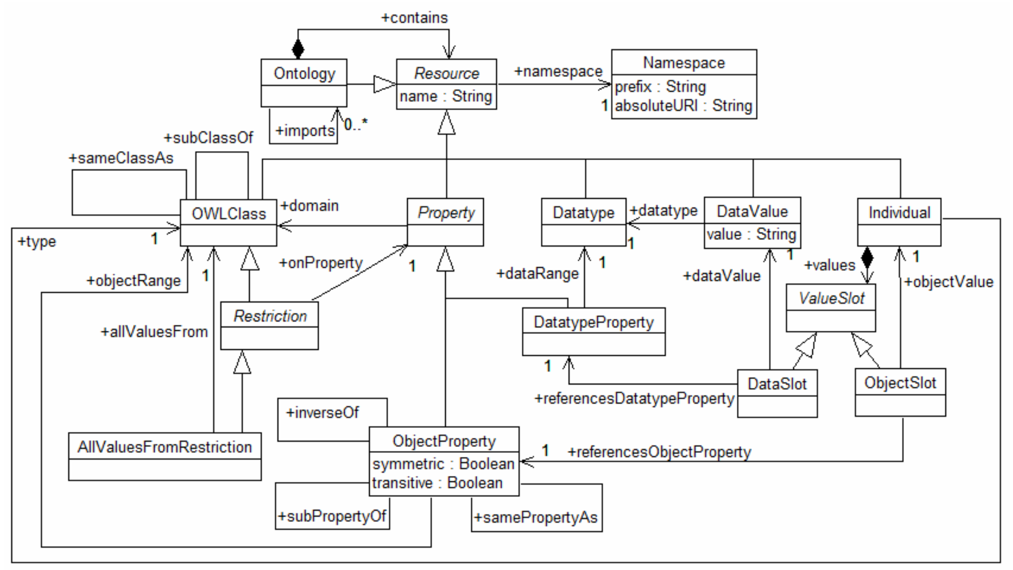

Figure 1. Lightweight Ontology Metamodel

Since OWL DL and Lite (and thus OWL Lite$^{\neg,p}$) require the disjointness of classes, properties, individuals and data values (cf. Schreiber and Dean 2004), we can remove the RDF layer from the ODM. Required associations (e.g., subClassOf relation) that are affected by this step are directly moved to appropriate OWL elements. A class Resource has been introduced as a generic top-level element having a reference to a Namespace. The association samePropertyAs (EquivalentProperty) has been move to ObjectProperty, since it is not available for a DatatypeProperty in OWL Lite$^{\neg,p}$. For providing a more explicit support of individuals and data values we introduce ObjectSlot and DatatypeSlot as proposed in an earlier version of the ODM proposal (IBM and Sandpiper Software Inc. 2005). The support of primitive data types is realized using typed literals -- i.e. a DataValue that references a single Datatype. Minor changes include the naming of some elements. The basic design and structure of the ODM, however, has not been changed for the OWL Lite$^{\neg,p}$ metamodel (see figure 1). Thus, the way of adding additional OWL features to the metamodel in future is already given by the ODM.

7.3 Lightweight Rule Metamodel

The design of the rules metamodel (see figure 2) is not driven by the syntax of the rule language but by the language features. Since disjunctions can be eliminated, we assume that every Atom is connected by a conjunction. We allow the definition of facts (i.e., rules without a condition), queries (i.e. rules without a conclusion), and rules. We separate between terms that are used to represent objects (ObjectTerm) and data values (DataTerm). Concrete instances are contributed by the OWL Lite$^{\neg,p}$ metamodel - i.e. Individual and DataValue. Depending on its type, an Atom consists of several object or data terms in a given order. A builtin must not be used in the consequent of a rule. ObjectClassification and PropertyReference finally enable the interaction of rules with OWL Lite$^{\neg,p}$. Both elements are imported from the ontology metamodel. Thus, properties and classes can only be used in rules, if they have previously been defined in the ontology.

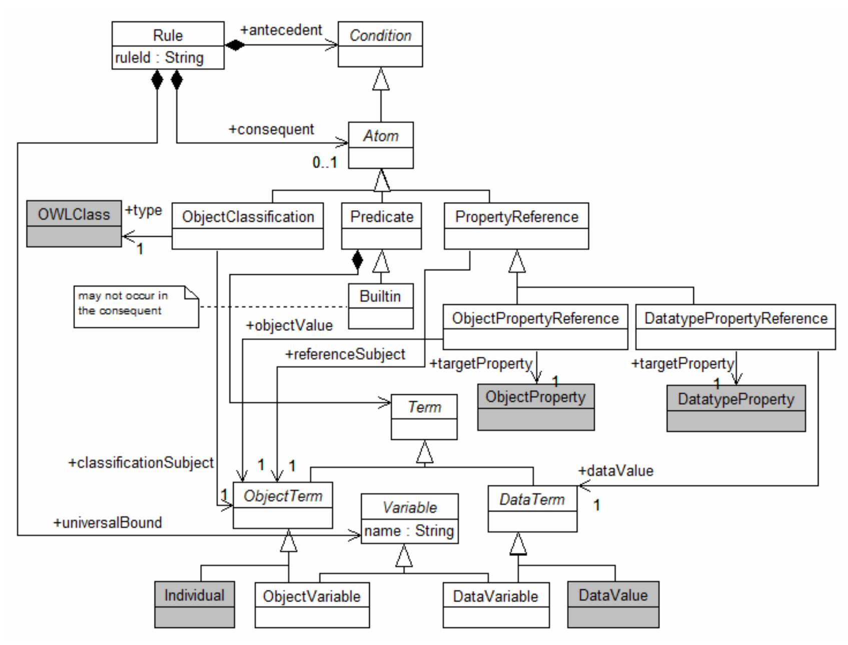

Figure 2. Lightweight Rule Metamodel

7.4 Advantages of a Model-Driven Approach

The metamodel presented above is fairly simple and allows for a good understanding of the underlying concepts. However, it only covers representational aspects of a knowledge base. In order to be useful in industrial practice, there are a number of related issues that have to be taken into account when using a knowledge base. These aspects can now be addressed without the need to commit to a single representation language.

Integration Platform. Using the metamodel we can now combine the use of existing and new development tools -- e.g., tools for OWL editing or visual modelling tools -- to provide a unified development environment. The metamodel also allows for a generic persistence structure. Finally, we can apply either code generation or model transformations to create ontologies and rules for a specific representation language.

Quality Control. We can realize generic methodologies for verification at the metamodel level for ensuring industrial-strength quality of the developed knowledge bases. By taking additional information about the application domain into account during the verification, the accuracy of the methodologies could be improved and thus we may also be able to detect modelling errors.

Security Model. Based on the metamodel, we can realize authorization mechanisms for knowledge bases. We intend to define a formalism to represent security policies, which is tailored to the characteristics of ontologies and rules. In particular, we want to reuse the concepts and relations defined in the ontology to provide a fine-grained access control. However, we will need additional information – i.e. business metadata – for properly defining these security policies. Therefore, we will investigate how this information overlaps with the one required for verification purposes. This could provide guidelines for developing generic business ontologies that can be reused in several applications within a company or even within an industry. In order to ensure a high level of security, we will finally develop verification methodologies for security policies similar to those for quality control.

8 DISCUSSION: TOWARDS MODEL-DRIVEN KNOWLEDGE BASE MANAGEMENT

In this paper we have argued for the need of a model-driven approach to manage knowledge bases for industrial applications. We are currently developing a management framework for industrial knowledge bases based on integrated metamodels of lightweight ontologies and rules. These models act as an interface between the actual knowledge base and the different management methods. In our work we will focus on methods for verification and authorization. The advantages of the approach are the following.

- Formal Foundation: The proposed metamodel provides an initial, formal foundation for addressing various industry-specific needs for knowledge bases – in particular related to quality and security - now. It represents a lightweight subset of ontologies and rules, which is suited for the representation of ontology mappings.
- Language Independence: The solution is not restricted to a single ontology and rule language. In fact it is open and adoptable to future standards and research results in that area. Depending on the used infrastructure we can finally transform concrete models to the required representation language.
- Modularization: The use of a central metamodel allows for building a highly modularized framework for support the development and management of knowledge bases.
- Extensibility and Reusability: Since the metamodel is based on subsets of existing ontology and rule languages, methodologies based on them can easily be extended or reused for supporting particularities of specific formalisms.

Future work includes the implementation of the metamodel and its extensions. While we can rely on existing work on verification for specifying and testing quality criteria, the adequate integration of business aspects and security policies are an open research question.

References

Angele J. and Gesmann M. (2007). The Information Integrator: Using Semantic Technology to provide a single view to distributed data. In Proceedings of BTW 2007.

Antoniou, G. et al. (2005). Combining Rules and Ontologies. A Survey. REWERSE Deliverable I3-D3.

Bobrow, D. G. and Mittal, S. and Stefik, M. J. (1986). Expert Systems: Perils and Promise. Communications of the ACM, 29 (9), 880-894.

Boley, H. and Kifer, M. (2007). RIF Core Design. http://www.w3.org/TR/rif-core.

Boley, H. et al. (2007). Rule Interchange on the Web. Reasoning Web, Lecture Notes in Computer Science, 4636, Springer, 269-309.

de Bruijn, J. and Polleres, A. and Fensel, D. (2004). OWL Lite$^-$. WSML Deliverable D20 v0.1.

de Bruijn, J. et al. (2005). OWL$^-$. WSML Deliverable D20.1 v0.2.

Bullinger H.-J. and Kiss-Preußinger E. and Spath D., Eds. (2003). Automobilentwicklung in Deutschland – wie sicher in die Zukunft. Frauenhofer IRB Verlag.

Calvanese, D. et al. (2006). Linking Data to Ontologies: The Description Logic DL-Lite$_{A.}$ In Proceedings of the 2006 International Workshop on OWL: Experiences and directions (OWLED 2006).

CIMdata (2007). Product Lifecycle Management (PLM) Definition. http://www.cimdata.com/PLM/plm.html.

Davis, M. (2006). Semantic Wave 2006 - Executive Guide to Billion Dollar Markets. Project10X Special Report.

Fensel, D. et al. (2006). Enabling Semantic Web Services: The Web Service Modeling Ontology. Springer.

Formatiert: Deutsch (Deutschland)

Garcia-Solaco, M. and Saltor, F. and Castellanos, M. (1995). Semantic Heterogeneity in Multidatabase Systems. Object-oriented Multidatabase Systems: A Solution for advanced Applications, 129-202.

Haase, P. et al. (2006). Networked Ontology Model. NEON Deliverable D1.1.1.

Hahn, A. (2005). Integration verteilter Produktmodelle durch Semantic-Web-Technologien. Wirtschaftsinformatik, 47, 278-284.

Hahn, A. et al. (2007). Using Ontologies to Model and Understand Product Development. IBIS - Interoperability in Business Information Systems, 5, 21-39.

Hepp, M. (2007). Ontologies: State of the Art, Business Potential, and Grand Challenges. Ontology Management: Semantic Web, Semantic Web Services, and Business Applications, Springer, 3-22.

Hitzler, P. et al. (2005). DLP is not so bad after all. In Proceedings of the OWLED*05 Workshop on OWL: Experiences and Directions, CEUR Workshop Proceedings, 188.

IBM and Sandpiper Software Inc. (2006). Ontology Definition Metamodel. Sixth Revised Submission to OMG/RFP ad/2003-03-40.

IBM and Sandpiper Software Inc. (2005). Ontology Definition Metamodel . Sixth Revised Submission to OMG/RFP ad/2003-03-40.

Jones, D. H. and Uschold, M. (2005). Role of Rules and Rule Interchange in Semantic Integration & Interoperability. In Proceedings of the W3C Workshop on Rule Languages for Interoperability, 27-28 April 2005, Washington, DC, USA.

Küsters, K. (2006). Teilautomatisierte Überführung der Produktstruktur in eine lastfallabhängige Modellstruktur mittels Ontologien. Diplomarbeit Rheinisch-Westfälische Technische Hochschule Aachen.

de Laborda, C. P. and Conrad, S. (2005). Relational.OWL - A Data and Schema Representation Format Based on OWL. In Proccedings of Conceptual Modelling 2005, Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005), Newcastle, Australia, January 30 - February 3, 89--96.

Object Management Group (2003). Ontology Definition Metamodel - Request For Proposal. http://www.omg.org/docs/ad/03-03-40.pdf.

Schreiber, G. and Dean, M. (2004). OWL Web Ontology Language – Reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210/.

Staab, S. (2002). Wissensmanagement mit Ontologien und Metadaten Habilitation Thesis, University of Karlsruhe.

Stegmüller, H. (2003). Zeit fürs Wesentliche - Audi erprobt semantische Technologien. Digital Engineering Magazin, 4, 44-45.

Stuckenschmidt, H. and Uschold, M. (2005). Representation of Semantic Mappings. In Proceedings of Semantic Interoperability and Integration, Dagstuhl Seminar Proceedings, 04391.

Syldatke, T. et al. (2007). How Ontologies and Rules Help to Advance Automobile Development. Advances in Rule Interchange and Applications. International Symposium, RuleML 2007, Orlando, Florida, October 25-26, 2007, Proceedings, Invited Paper, 1-6.

Volz, R. (2004). Web Ontology Reasoning with Logic Databases. PhD Thesis, University of Karlsruhe.

Formatiert: Englisch (Großbritannien)