Modeling Structure in Description Logic

Henson Graves, Yvonne Bijan

Algos Associates,
2829 West Cantey Street,
Fort Worth, TX 76109 United States
Henson.graves@hotmail.com

Abstract. An overall goal of the INCOSE MBSE initiative is to provide SysML with a formal semantics and to integrate reasoning services as part of system engineering. UML class diagrams have been encoded as Knowledge Bases (KB) within the Description Logic (DL), ALCQI. The encoding provides a formal semantics for class diagrams which accords with the informal semantics. The encoding applies to SysML which is a profile of UML. The SysML block definition and internal block diagrams are not covered by the class diagram encoding. These diagrams are essential for representing composite structure such as manufactured products and molecular structures. The class diagram encoding is extended to composite structure diagrams in the DL ALCQIb$_{id}$. A composite structure diagram describes structures in terms of part decompositions and connections between objects. A SysML composite structure diagram can be encoded in the language of OWL2, but is not an OWL2 axiom set, as the diagrams contain property equations which violate the regularity ordering constraints for complex property inclusions. Conditions are given for an ALCQIb$_{id}$ KB which are sufficient to encode a SysML composite structure diagram. Further conditions are given for a KB, called a template, which ensure that within an interpretation all realizations of the composite structure have the same graph structure.

Keywords: Description Logic, Ontology, OWL, SysML, UML, structural modeling, molecular chemistry, human anatomy.

1 Introduction

Many engineering tasks involve reasoning on a description (a model in engineering terminology) to determine consistency and to derive implicit knowledge. An overall goal of the INCOSE MBSE initiative [4] is to provide the system engineering modeling language SysML [10], a dialect of UML [11] with a formal semantics and to integrate reasoning services as part of system engineering. SysML lacks a formal logic-based semantics, but has a well-developed informal semantics. For reasoning to give correct results, the formal semantics must be in accord with the informal semantics of SysML. To provide a formal semantics, one may axiomatize SysML directly [7] or encode SysML in a language which has a formal semantics such as OWL2 [12]. The OWL2 semantics is based on the Description Logic (DL)[1], SHOIQ [8].

UML class diagrams have been encoded as a Knowledge Base (KB) within the DL, ALCQI [2], a sublogic of SHOIQ. In this encoding, UML classes are encoded as concepts and UML associations are encoded as roles; to encode the additional information contained in class diagrams, other KB assertions are needed. The result is that an encoding of a UML class diagram is as a KB. The encoding provides a formal semantics for class diagrams which conforms to their informal semantics; the encoding is further validated by comparison of first order logic (FOL) axiomatizations of the UML constructions with the FOL representation of description logic. A consequence of the encoding for integration with reasoning is that a class diagram (class model) corresponds to a knowledge base within a DL. The results of DL consistency checking and derived classes and class inclusions can be reinterpreted within UML.

The DL encoding for UML and its results carry over to SysML. In SysML, blocks are a stereotype of class and SysML uses associations as does UML. SysML is well suited for representing descriptions of composite structure [5]. The SysML diagrams (models) used to represent composite structure are not fully covered by the UML encoding. A composite structure consists of objects and part objects linked by connectors. A structural description is a collection of classes and properties which describe a structure. A structure which satisfies the description is called a realization of the structure. A composite structure is represented in SysML with a Block Definition Diagram (BDD) and an Internal Block Diagram (IBD). A variety of specializations of associations are used to represent part properties and structural connections. Both part properties and connections are binary properties. A BDD describes a part decomposition structure. An IBD is a BDD with connection properties and property equations. An IBD can be used for representing structures which have, for example, multiple objects of the same class, which play different roles in the description. For example, an automobile description may specify four wheels with two front wheels which are driven by the engine and two rear wheels which are not driven. A SysML IBD model of the human heart accords well with the informal semantics and elucidates the distinction between the different kinds of properties (parts and connections) used to describe a heart.

Finding an appropriate Description Logic to represent the class of composite structure diagrams which is sufficiently expressive and for which reasoning is decidable and computationally tractable is challenging. A SysML IBD can be encoded in the language of SROIQ, but in the direct encoding it is not an OWL2 axiom set, as the connection property equations of an IBD violate the regularity ordering constraints on SROIQ axioms [8]. While it is possible to represent these diagrams within SROIQ with a description graph extension [9], there are questions regarding whether the description graphs correctly capture the intended semantics. For a DG extension of OWL2, the FOL suggested semantics for the human heart example in [9] is different from the DL semantics of a SysML IBD which appears to capture the informal semantics faithfully.

The role equations needed to represent the constructions in a SysML IBD are very restricted, even though they do not satisfy the regularity conditions of OWL2. The roles which represent part properties are all atomic with specified domain and range classes which are also atomic. The conditions satisfied by the part properties of a SysML IBD ensure that the part role paths [3] are finite and are unique. For each part

role path, a new atom can be introduced to represent the path. Using these atomic path roles, simple role hierarchy assertions are sufficient to represent the equalities found in an IBD.

The UML class diagram encoding and its extension for conceptual modeling for data integration [3] is used to encode SysML Block Definition and Internal Block Diagrams using the description logic $ALCQIb_{id}$. A part structure for a KB is defined which encodes the essential features of a BDD. Conditions are given on a KB to encode the property equation features of an IBD. Additional meta conditions are given for an IBD KB, called a template, which ensure that within an interpretation, all realizations of the KB have the same graph structure. The conditions for a template are easily checked. A template is illustrated with a SysML model of the water molecule. For a template, results computed from the structure of the KB are valid in any realization. For example the weight of a structure can be computed from the IBD as all realizations will have the same number of parts.

2 The Description Logic $ALCQIb_{id}$

The specific description logic used to encode SysML Internal Block Diagrams is $ALCQIb_{id}$ [3]. $ALCQIb_{id}$ is an expressive DL that extends the basic DL language AL (attributive language) with negation of arbitrary concepts (indicated by the letter C), qualified number restrictions (indicated by the letter Q), inverse of roles (indicated by the letter I), boolean combinations of roles (indicated by the letter b), and identification assertions (indicated by the subscript id). Concepts and roles in $ALCQIb_{id}$ are formed according to the following syntactic rules [3]

$$C, C' \rightarrow A \mid \neg C \mid C \cap C' \mid C \cup C' \mid \forall R.C \mid \exists R.C \mid \geq n\ Q.C \mid \leq n\ Q.C \quad (1)$$

$$R, R' \rightarrow P \mid P^- \mid R \cap R' \mid R \cup R' \mid R \setminus R' \quad (2)$$

where A denotes an *atomic concept*, P an *atomic role*, P^- the *inverse* of an atomic role, C an arbitrary *concept*, and R, R' arbitrary roles. Furthermore, $\neg C$, $C \cap C'$, $C \cup C'$, $\forall R.C$, and $\exists R.C$ denote negation of concepts, concept intersection, concept union, value restriction, and qualified existential quantification on roles, respectively. We then use Q to denote *basic roles*, which are those roles that may occur in expressions of the form $\geq n\ Q.C$ and $\leq n\ Q.C$. A *basic role* can be an atomic role or its inverse, or a role obtained combining basic roles through set theoretic operators, i.e., intersection ("∩"), union ("∪"), and difference("\"). W.l.o.g., we assume difference applied only to atomic roles and their inverses.

Abbreviations are introduced for terms and assertions. *Thing* denotes the top concept which can be defined as $C \cup \neg C$ for a concept C, and *Nothing* the bottom concept. The concept $kQ.C$ is an abbreviation of $\leq$kQ.C ∩ $\geq$kQ.C. The *empty* denotes the role p\p for a role p. The notation *(funct p)* is used for the assertion that p is *functional*, i.e., as an abbreviation for $Thing \leq 1\ p.Thing$. The notation *p:(A,B)* is an abbreviation for the assertions

$$A \leq \forall p.B \text{ and } B \leq \forall p^-.A \quad (3)$$

which capture the property that *A* is the domain and *B* is the range of *p*. For a functional role with *p:(A,B),* we have $A \leq 1p.B$. For a role *p:(A,B)* that is functional, we use the notation *p:(A,B)[1]*. Two roles p and q are *disjoint*

$$p \perp q \text{ IFF } p \cap q = empty \quad (4)$$

A *path* is given by the production rule

$$\pi 1,\ \pi 2 \rightarrow S \mid D? \mid \pi 1.\ \pi 2 \quad (5)$$

where S denotes an atomic role or the inverse of an atomic role, *D* denotes a concept, and $\pi 1.\pi 2$ denotes the composition of paths $\pi 1$ and $\pi 2$. The expression *D?* is called a test role, it denotes the identity relation on instances of the concept *D*. If π is a path, the length of π, denoted length(π), is 0 if π has the form C?, is 1 if π has the form S, and is length(π1) + length(π2) if π has the form $\pi 1.\pi 2$.

An $ALCQIb_{id}$ knowledge base (KB) is a pair ‹T, A›, where T is a TBox and A is an ABox. A TBox is a finite set of assertions of the form $C \leq C'$ with C and C' arbitrary concepts, or of the form $R \leq R'$ with arbitrary roles R and R', or an identification constraint. An *identification constraint* is an assertion of the form *(id C* $\pi 1, \ldots, \pi n)$ where C is a concept, $n \geq 1$, and $\pi 1, \ldots, \pi n$ are paths (called the components of the identifier) such that $length(\pi i) \geq 1$ for all $i \in \{1, \ldots, n\}$ and $length(\pi i) = 1$ for at least one $i \in \{1, \ldots, n\}$. Intuitively, an identification constraint asserts that for any two different instances o, o' of C, there is at least one π_i such that o and o' differ in the set of their πi-fillers. An ABox is a finite set of membership assertions of the form *A(a), P(a, b)*, and $a \neq b$, with *A* and *P* respectively an atomic concept and an atomic role occurring in T, and *a, b* constants. The condition for role inclusions is weaker than the standard condition for a KB in $ALCQIb_{id}$.

The semantics of $ALCQIb_{id}$ concepts and roles is given in terms of interpretations, where an interpretation is defined as a correspondence of the KB concepts and roles [3] with classes and properties in a domain for which all of the KB assertions are satisfied. The semantics of a path π is defined in terms of the reverse of the composition of the roles occurring in the path. This device allows us to express well-formed path equations as role assertions within an $ALCQIb_{id}$ KB.

The semantics of an $ALCQIb_{id}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is the set of models of $\mathcal{K}$, i.e., the set of interpretations satisfying all assertions in T and A. As noted in [3], checking whether an assertion holds in every model of a KB, is decidable in deterministic exponential time.

While $ALCQIb_{id}$ works for encoding SysML block diagrams it seems likely that some new variant could be devised which could be tailored more precisely for representing composite structure models.

3 Encoding SysML Block diagrams in a DL

We review the principles of encoding class diagrams established in [2] as applied to SysML block diagrams. We use a simple illustration of a water molecule model to show how role equations naturally occur in a composite structure diagram. For the water example we show that all realizations have the same structure. The next section

will generalize the concept of a KB which abstracts the properties of a composite structure and show that a kind of KB called a template enjoyes the properties that all of its realizations are isomorphic.

The SysML language uses blocks which are classes and associations with several predefined kinds of specializations. The molecular unit of SysML is called a model. A SysML model is a collection of declarations which introduce constants of a signature and specify typing relations. A SysML model may contain subclasses and limited kinds of role assertions and may be composed and presented using multiple visual diagrams. However, all of the diagrams that constitute a model use the same block and property symbols.

A class diagram in UML is a restricted kind of SysML model. The encoding of UML class diagrams [2] carries over to SysML which is a UML profile developed for systems engineering. The Description Logic ALCQI is used to provide the encoding. This encoding accords with the informal semantics of UML. Classes (SysML Blocks) and associations are translated into DL concepts and roles [2]. The translation of a class diagram is as a role assertion. However, SysML models are not covered by the encoding in [2]. In particular, a SysML Block Definition Diagram and an Internal Block Diagram are not covered. An undirected association p is identified with a role p. The diagram of boxes labeled A and B connected by a line becomes the assertion

An *aggregation* property p from A to B with cardinality restriction 1 is represented in DL this becomes $A \leq (\exists p)$. A property *p* with *p:(A,B)* is *mandatory* if $A \leq k\ p.\ B.$ for an atomic functional role p with domain A and range B we use the abbreviation p:(A,B)[1].

This encoding of a SysML model as a KB is illustrated with a SysML water molecule model. The water molecule is represented as a SysML model using two kinds of diagrams, a Block Definition Diagram (BDD) to represent the decomposition structure and an Internal Block Diagram (IBD) to represent relationships among the parts within the structure. The language elements in both diagrams are part of the same SysML model. In Figure 1, the top half shows the decomposition structure for water. The BDD shows that water has three part properties whose range classes are oxygen and hydrogen. The shared Association (open diamond headed arrow) is a part property in SysML. There are two kinds of part properties in SysML. The shared Association property is used because the atoms can be a part of any molecule. The two arrows pointing at hydrogen mean that there are two parts of type hydrogen within water. The diamond arrow pointing to oxygen shows that there is one part of type oxygen within a water molecule. The numbers on the arrows in this diagram represent the cardinality restriction on the number of parts that a water molecule can have. In this case, the numbers are all 1, which says that an individual water molecule has exactly one oxygen and two hydrogen atoms as parts.

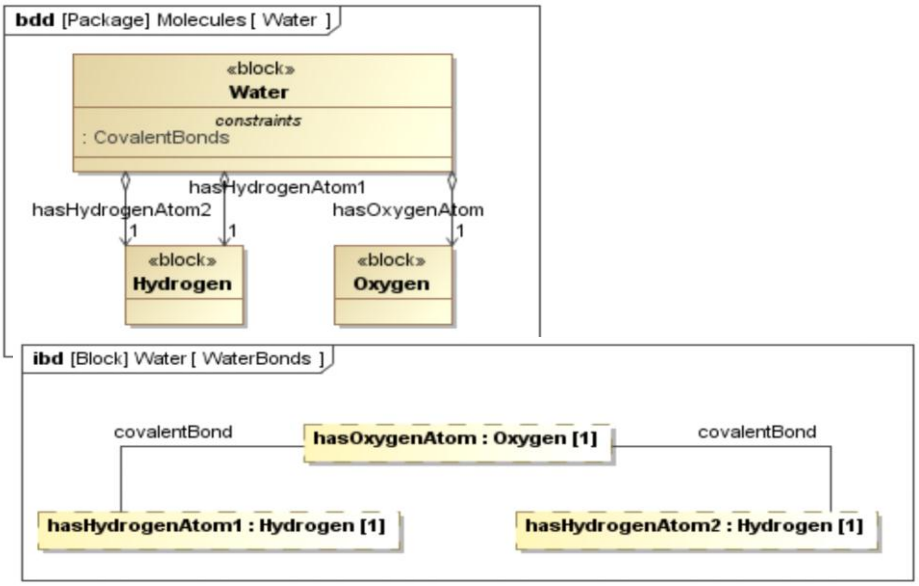

Fig. 1. SysML model for water molecule

The bottom diagram, called an Internal Block Diagram, shows the bonding relationships between the part properties. The arrows from the BDD are represented as rectangles with dashed lines. In this diagram, the rectangles are not blocks; they represent part properties of Water. The rectangle is labeled with the name of the part property and the range type of the property, as well as the cardinality restriction of the properties. The diagram shows the oxygen part has a covalent bond with each of the hydrogen parts. The diagram title box signifies that the IBD is within the scope of water.

The informal semantics of the water model is that any realization of a water molecule has exactly three atoms: two hydrogen atoms and one oxygen atom. Further, we expect that the covalent bonds from oxygen atom are connected to distinct hydrogen atoms. This fact will follow from the declaring the *covalentBond* property to be functional and declaring that the *hasHydrogenAtom1* and *hasHydrogenAtom2* are disjoint. Informally, a realization of a water molecule in this theory is a tree with a root *w1* corresponding to the water molecule and three other nodes, an oxygen atom *o1*, and two hydrogen atoms *h1, h2*. The tree has edges *{ <w1,o1>, <w1,h1>, <w1,h2>, <o1,h1>,<o1,h2> }*. The first three edges correspond to the part properties and the last two correspond to the bond properties.

For a KB used to encode a SysML composite diagram a *realization* of a KB is a collection of ABox assertions of the form objects $C_j(a_i)$ where the C_j are atomic and for any atomic concept *C* in the KB there is at least one ai with $C(a_i)$. Also, for any atomic role p in the KB there is a pair $<a_i,a_j>$ with $p(a_i,a_j)$. There may be multiple a_i with $C(a_i)$. A realization is an internal model of the KB. In general, a realization may not be finite. With the restrictions that will be used on a KB to encode an IBD, one

can construct realizations of the KB by adding individuals. A template KB has finite realizations. Also, a model may contain multiple realizations.

In the first order logic representation of a DL we replace an existential assertion *p:(A,B)[1]* with a Skolem function and use the symbol *p* for the Skolem function as well as the role. We use the notation *a.p* for the value of the Skolem function *p*. This notation allows us to write *p(a, p(a))* as *p(a,a.p)*. More generally for *p1* and *p2* functional atomic roles then from the composition semantics for roles one has *a.(p1.p2) = (a.p1).p2*.

The KB encoding the SysML water molecule model has as atomic roles, Water, Oxygen, and Hydrogen and as atomic roles, *hasOxygen*, *hasHydrogen1, hasHydrogen2, covalentBond1, covalentBond2*. Using the abbreviations the KB contains the assertions:

$$hasOxygen:(Water, Oxygen)[1] \tag{6}$$

$$hasHydrogen1:(Water, Hydrogen)[1] \tag{7}$$

$$hasHydrogen2:(Water, Hydrogen)[1] \tag{8}$$

$$covalentBond1:(Oxygen, Hydrogen)[1] \tag{9}$$

$$covalentBond2:(Oxygen, Hydrogen)[1] \tag{10}$$

the equational role equations

$$hasOxygen.covalentBond1 = hasHydrogen1 \tag{11}$$

$$hasOxygen.covalentBond2 = hasHydrogen2 \tag{12}$$

and the disjointness assertions

$$Oxygen \perp Hydrogen \tag{13}$$

$$hasHydrogen1 \perp hasHydrogen2 \tag{14}$$

$$covalentBond1 \perp covalentBond2 \tag{15}$$

It is easy to show that any realization of the water KB has the same structure. For any *ABox w with w.Water*, we iterate the constructions to obtain the set

{w, w.hasOxygen, w.hasHydrogen1, w.hasHydrogen,
w. hasOxygen.covalentBond1, w.hasOxygen.covalentBond2 }.

However,

$$w.\ hasHydrogenAtom1 \neq w.\ hasHydrogenAtom2 \tag{16}$$

by axiom (12). By axiom (9)

$$w.\ hasOxygen.covalentBond2 = w.\ hasOxygen.covalentBond2 \tag{17}$$

The structure only contains the root and part instances. We can verify that the role instance relations hold. This KB implies that any realization of Water has the expected component parts with the expected connections between them. The next task is to identify the properties of part roles which enable these arguments to be generalized.

4 Block Definition Diagrams and Internal Block Diagrams

The KBs which are used to encode SysML BDDs and IBDs are described below. An abstract Block Definition Diagram (ABDD) is a KB with a subset of the KB signature $p_i : i \in \{1,...n\}$ called part roles. Part roles satisfy the constraints that each p_i is declared with a domain and range type with a numeric multiplicity, the domain and range concepts of the part roles are in the KB signature and are atomic. There may be multiple part properties with the same domain and range types. For the following discussion, we restrict part properties to be functional. We use the abbreviation *p:Part(A,B)* for a part role *p* with *p(A,B).* The concepts that occur as the domain or range of a part role are called *Part* concepts. A part concept which is not the range concept of any part property is a root. We assume that the part class has a root and that it is unique. A *part path* is a well-formed composition of part properties *p1*. *p2* ….*pn* were the *range(pi) = domain (pi+1)* for all i. A part path is called acyclic if domain(pi) ≠ range(pi) for any pi in the path.

(P0) *Root(A)*, for some atomic class *A* and the class is unique.

(P1) *If p is a part path then p is acyclic*

(P2) *If p:Part(A,B) and p2:Part(C,B) then p1* $\perp$ *p2*

(P3) *PartClass(A) IFF Root(A)* or *p:Part(B,A)* and *PartClass(B)*

(P0) identifies a concept as the root. (P1) ensures that part paths do not contain multiple occurrences of a part properly and so have finite length. (P2) states that any two part roles with the same range are disjoint. The (P3) implies that all part concepts are connected the root by a part path. The meta-properties (P0) through (P3) are easily in a KB.

For an ABDD, the directed graph whose nodes are the root together with the expressions *p:A*, for a part property *p* with *A = Range(p)* and whose edges are the part roles is a tree. As there may be multiple parts with the same range, class labeling the class with the part role using the expressions of the form *p:A* makes the nodes distinct. Each part concept is reachable by a part path *p1…pn* where *pn = p* from the root by (P4). For any two property paths *p1…pn* and *q1…qk* which terminate at the node *p:A*, then the domain of *pn* and *qk* are equal. By (P2), the part roles *pn* and *qk* are disjoint and so the two paths cannot be equal. Thus, the part property path is unique. The ABBD is used to encode a SysML BDD. The conditions used to define an ABDD are enforced in a BDD.

An Abstract IBD (AIBD) is an ABDD together with a finite set of function role (connections) whose domain and range are part concepts, and a set of path equations of the form

$$p1...pn = q1...qk.c \tag{18}$$

where the *pi* and *qj* are part paths and *c* is in *{c1,..,ck}*. The connection roles encode the arrows between the boxes of an IBD. Conversely an AIBD can be displayed as a graph. A new atomic role p.c is introduced for any part path followed by a connection role. Note that p1…pn:(A,B) where domain(p1) = A and range(pn) = B. We extend this notation to p1….pn.c for a connection role c.

A path π of atomic roles $p_1,\ldots p_n$ is *well-formed* if range(p_i) = domain(p_{i+1}) for all $i \in \{1, \ldots, n\}$. For each well-formed path, π, of atomic roles, we introduce a new role atom π. For any path of length 1, the new role is identified with the atomic role that formed the path. As there is a finite number of part paths, we define a new atomic role for each part path *p1….pn*. Recall that when the atoms in a path are functional, we write *a.p* for the unique individual *b* with *p(a,b)*. This notation simplifies the application of a path *p* applied to *a*. We also use the notation *p:Path(A,B)* for a path with domain *A* and range *B*.

While the DL ALCQIb$_{id}$ does not permit composition directly in the role inclusion assertions, we simulate composition with the atomic path roles. For any connection equation p1…pn = q1…qk.c in the IBD, we replace it with role inclusion assertion $q1...qk.c \leq p1...pn$. However, for any a,b , q.c(a,b) implies *p(a,b)*. However, if *a.A* then as *p* is functional, *a.p = a.c.b*. Thus, *q.c = p*. So the inequalities in an AIBD are actually equalities. A *template* is a AIBD where

(P5) *p1:Part(A,B)* and *p2:Part(A,C)* then $p \perp q$ or p = q

The template axiom says that no parts can be reused in a part decomposition. This statement can be made precise in the first order logic theory of the KB.

To prove properties about the models of an AIBD KB we use the full first order representation of the DL. In the theory generated by the KB existential assertions of the form p:(A,B)[1] are replaced by a first order Skolem function. Properties that hold in this theory will hold in any model of the KB. Note that the part path roles become functions. Thus, the notation *a.p1..pn* is meaningful and we have the associativity law a.(p.q) = (a.p).q.

Definition. For a template KB with root *A*, *a:A, and t:B* for a part class *B,* let

$$Partof(a,t) \text{ IFF } t = a \text{ or } a.p1....pn \tag{19}$$

for a part path.

Lemma. For a template with root *A*, and *a:A:* If *t:B* for a part class *B* and *t* is a part of a, then the part decomposition is unique.

If *t* is a part of *a,* then *t* has a decomposition of the form *t = a.p1….pk* for some *p1,…,pk*. If *t = a.p* and *t = a.q*, for two part properties, then by the template property *p* disjoint q or they are equal, and so *a.p = a.q*. The argument is repeated for the

successive individuals *(a.p1).p2…pn*. With the first order logic of the KB extended with an abstraction construction which allows terms of the form

$$\{ t : P(t)\} \quad (\mathbf{20})$$

constructed from a predicate where the predicate is restricted to equalities with Boolean connectives, then we can define the notion of a realization of a template within the extended theory of the axiom set. The axioms for the abstraction construction include

$$P(a) \text{ IFF } a:\{ b : P(b)\} \quad (21)$$

with usual rules for variables. Using the extended logic, a realization of a template is an abstraction type $G = \{t : Partof(a,t) \}$ for *a:Root*. From the axioms for the part property declarations, a realization of a root instance has a unique part decomposition. A tree structure can be defined with the individuals in *G* as the nodes and $<t,t.p>$ for a part property. Connection edges can be added similarly. A graph isomorphism can be inductively defined between any two realizations.

Theorem. For a template, the instances of the type $G = \{t : Partof(a,t)\}$ for *a:Root* are the nodes of a tree with root *a*. The edges $<t, t.p>$ where *range(t) = domain(p)*. The correspondence defined by mapping *a* to the root and a node of the form *t.p* to *p:Range(p)* corresponds the nodes of the parts structure with the nodes of the BDD together with the mapping of an edge $<t,t.p>$ to the edge *p* in the BDD defines an isomorphism of the parts structure with the BDD. Any parts tree has the same number of parts.

The axioms given do not prohibit a structure from sharing individuals with another structure. This property can be added with the axiom:

(P6) *p:Part(A,B)* implies $p.p^*=id(A)$

An interpretation of an axiomatic SysML theory is a mapping of the individuals, pairs, classes, and properties, and other types which preserves the sort structure, the logical axioms, and the declarations. In particular, classes are mapped to subclasses of the mapping of Thing and individuals are instances of Thing. In any valid interpretation of the theory of a model, the unique decomposition will hold.

5 Conclusion

The encoding of a UML class diagram as an ALCQI KB gives an encoding of Class diagrams into OWL2. This encoding is extended to an encoding of a SysML Internal Block Diagram as a KB within the OWL2 language, but not as an OWL2 KB. Each atomic property in the Block Diagram is an atomic role and the well-formed property paths are finite. The encoding correctly captures the part decomposition which ensures that the models are tree like. SysML model development tools enforce the axioms that define a part structure. Conversely, the correspondence between the block

diagrams and the DL language constructions provides a graphical syntax for DL. SysML dos not have individuals, i.e., ABoxes. The Encoding makes clear how individuals can be added to SysML. With the encoding all derivations of inconsistency and concept inclusions can be exported back into SysML.

It is easily to check whether the additional axioms for a template are present. These axioms correspond to manufacturing assumptions that ensure implementations of a design have the same parts and connection structure. For example, water is a subclass of molecules which have an oxygen part. However, Oxygen is not a subclass of the things bonded to hydrogen, only the oxygen molecules which are parts of water have this property. The use of a template KB enables the development of SysML models for which all realizations are isomorphic. This is very useful as computations of the model hold for all of its realizations. It seems likely that graph defined for an abstract IBD, is a Description Graph in the sense of [9].

The axioms given for the water model provide only structural information and are incomplete in terms of constraints on the bonds needed to determine a 3D visualization of a water molecule and do not address the dynamic behavior of water such as how it changes when it freezes. Much more complete axiomatic models of water can be given which address these properties. These SysML models require further extensions to DL to be addressed.

References

1. Baader, F., D Calvanese, DL McGuinness, D Nardi.: The description logic handbook, Cambridge University Press (2007)
2. Berardi, D., Calvanese, D., and De Giacomoa, G., Reasoning on UML class diagrams, Artificial Intelligence Volume 168, Issues 1-2, (2005)
3. Calvanese, D., De Giacomo, G., Lembo, D., Conceptual modeling for data integration, (2009)
4. Friedenthal, S., Greigo, R., and Sampson, M., INCOSE MBSE Roadmap, in "INCOSE Model Based Systems Engineering (MBSE) Workshop Outbrief", INCOSE International Workshop, Albuquerque, NM, (2008)
5. Graves, H.: Representing Product Designs Using a Description Graph Extension to OWL 2. OWLED (2008)
6. Graves, H.: Integrating SysML and OWL, OWLED (2009).
7. Graves, H.: Ontological Foundations for SysML, IC – MBSE 3rd International Conference on Model-Based Systems (2011)
8. Horrocks, I., Kutz, O., and Sattler, U.: "The Even More Irresistible SROIQ," in Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning, pp. 57-67, American Association of Artificial Intelligence Press, (2006)
9. Motik, B., Cuenca Grau B., Sattler, U.: Structured objects in OWL: Representation and reasoning, Proceeding of the 17th international conference on World Wide Web (2008)
10. OMG Systems Modeling Language (OMG SysML™), V1.2 (2010)
11. OMG Systems Modeling Language (OMG UML) V.2 (2010)
12. OWL 2 Web Ontology Language W3C, September (2009)