Hetereogeneous Logics for Semantic Manufacturing

Luis RAMOS
University of Bremen, Cartesium
Enrique-Schmidt Strasse
28359 Bremen, Germany

Abstract. Semantic Manufacturing has grown as an application area of the Semantic Web Framework. This approach enables reasoning on products in the form of ontologies and involves products requirements verification within its scope. However, there are cases where the expressivity of OWL, *a defacto* standard for the Semantic Web, is insufficient for expressing fundamental features of designs. As such features are not expressible in OWL, it is not possible to carry out many verifications of them. OWL is therefore a limited scope language for such applications. In this work we present in detailed uses cases to document the presence of such requirements in engineering. Subsequently, we introduce a heterogeneous architecture to overcome the shortcomings of OWL. This consists of the combination of the Common Algebraic Specification Language (CASL) with OWL within the Heterogeneous Tool Set (HETS). A sample CASL specification to prove certain design requirements is presented in order to demonstrate how such requirements can be modeled and verified trough this heterogeneous framework.

Keywords. Semantic Web, OWL, CASL, HETS, NASA, SPASS, manufacturing

1. Introduction

Ontology and Semantic Web based approaches are receiving increasing interest as a means to support the manufacturing industry in carrying out several activities. Authors mostly suggest that integrating Semantic Web into products descriptions enables semantic querying and reasoning on products models. Thus, flexible manufacturing systems, design decision support [1], development and quality control of digital designs can all be facilitated. To support this, some CAD standards [2], [3] have been exchanged into ontology languages like OWL; so that reasoning over mechanical designs becomes enabled. Such reasoning includes the possibility of comparing a design against a given specification in order to determine if the given design fulfills the standard requirements. As a consequence of this, automatic validation of design can increase the workflow in production processes.

Nevertheless, besides the benefit obtained by this integration, there are some fundamental shortcomings of OWL that have been neglected in most research that has proposed OWL as language for Semantic Manufacturing.

In this paper we aim to present some OWL shortcomings and to describe some use cases where such shortcomings can be identified. Moreover, we propose an approach that can help to overcome the issues introduced here.

The rest of this paper is divided as follows; Section 2 presents two use cases with certain engineering requirements for which the expressivity of OWL is insufficient for a proper modeling. In Section 3, we introduce CASL and HETS as a framework to deal with heterogeneous ontological environments. Section 4 describes a heterogeneous architecture that integrates OWL and CASL for reasoning about products requirements. Section 5 gives a description of an implemented specification; moreover we present some initial results of design requirements validation. Finally, we present a discussion about our results and future work.

2. Expressivity versus Decidability Issue

When dealing with deciding about the use of a given language to develop an ontology, we should firstly decide about expressivity versus decidability requirements, given that it is not possible to have both at the same time. For instance, on the one hand, OWL-DL (Description Logic based) has a limited expressivity level, but supports reasonable decision time out. On the other hand, more expressive logics like First Order Logic (FOL) enable a higher expressivity level, but it is not decidable.

To illustrate this difference we can consider the predicate arity. OWL is a binary predicate language, that is we can express relations between two concepts by an OWL predicate (object property in OWL terminology), or a relation between a concept and a numeric value (data type property in OWL terminology), but the representation of higher arity predicates is limited.

In the following two examples we will demonstrate how this higher arity issue naturally appears in common engineering tasks.

2.1. Sheet Metal Parts (SMP) Fabrication

Sheet metal parts (SMP) are common fabrication elements in industry; they can be found in electronic and automotive appliances. In their manufacturing, raw material specifications and dimensioning has to be considered in design prior to fabrication. That means, designers have to take several restrictions into consideration in order to avoid or reduce the waste of time or material during the manufacturing process.

Radhakrisnan *et al* [4] define two kinds of manufacturing restrictions, intrafeatures and interfeatures restrictions. Interfeatures restrictions are properties of the features themselves (e.g. diameter of a hole), while intrafeatures restrictions correspond to attributes that relate two or more features (e.g a given value of distance between holes). In this sense, **Figure 2-1**shows a sample SMP indicating two intrafeature restrictions commonly referenced in the technical literature. These are the distance between holes (dbh) and distance from hole to border (dhb). Each one of these restriction involves two ***elements*** of the SMP and relates them by means of a valued number restriction. In Formula 1, an intrafeature restriction is presented, while in Formulas 2 and 3 interfeature restrictions are introduced.

$$has\_Diameter(h_1, d) \tag{1}$$

$$dis\tan ce\_Between\_holes(h_1, h_2, d) \tag{2}$$

$$dis\tan ce\_Hole\_border(h_1, b_2, d) \tag{3}$$

The restriction described in Formula 1 can be represented in OWL and conclusions about the quality of the feature itself can be inferred by reasoning. Restrictions expressed in Formulas 2 and 3 can be indirectly express in OWL [5], but no conclusions about the quality of this design can be obtained from the OWL model.

It is also worth mentioning that the value of these features (**dbh** and **dhb**) shown in **Figure 2-1** also depend on the raw material and other SMP features, for instance its thickness. Moreover, we also avoided mentioning metric units aiming at a simplification of the problem. The example given above is an illustration of a kind of issue involving higher predicate arity in a specific task where OWL expressivity is insufficient to reach the validation of certain mechanical features. However, in the following subsection we introduce a more general use case to demonstrate the general requirement of higher order predicate arity in manufacturing and in engineering in general.

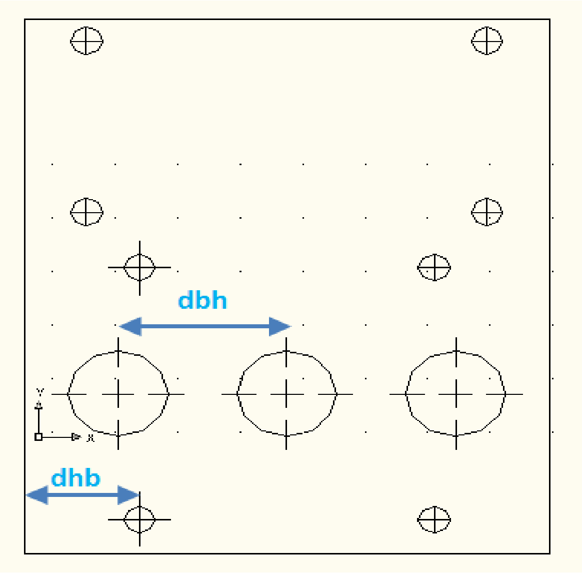


Figure 2-1 Restrictions in Sheet Metal Parts Fabrication

2.2. The Mishap Issue

In this use case we will refer to engineering issues related to Metrology, which is defined as the science of measurement and its application. Measurement is fundamental in all engineering areas, so engineers are required to have wide knowledge of the most commonly used standard measurement systems. While this is a general basic professional requirement, the issue appears when computer based systems have to deal with such engineering information and identify possible input or processing errors, when receiving data from different and multiple sources.

In this vein, computer-based systems should be able to prove units and dimensional homogeneity and should also be able to indicate when a dimensional specification violation or dimensional heterogeneity takes place as well. Such a scenario may unlike to occur, but in September 1999 the National Air and Space Administration (NASA) lost its Mars Climate Observer (MCO) due to the failure of using metric units in the coding of a software file [6]. In other words, while some coding was done using English units (e.g. inches), another part was coded using metric units (e.g. meter), when it was supposed that every dimensioning was in metric units. The result of this inconsistency is in **Figure 2-2**. Here, two trajectories around Mars are shown. The former corresponds to the estimated trajectory which should have been followed by the MCO, and the later corresponds to the one really followed by the MCO. Given that the real orbit point was too low (57 Km of altitude instead of 226 km) the landing could not be properly performed and the Mars Observer was lost.

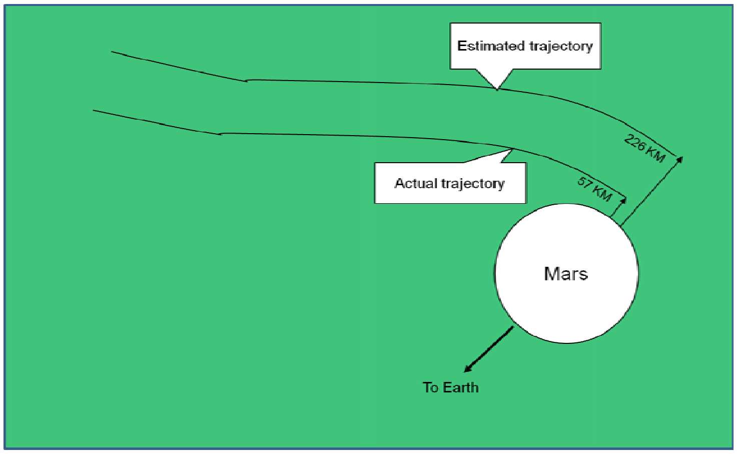


Figure 2-2 Mars Observer Orbit

$$dis\tan ce\_to\_mars\_Km(mco,d)$$ (4)[1]

$$dis\tan ce\_to\_mars(mco,v,u)$$ (5)

A model of the scenario depicted above is formalized in Formulas 4 and 5. On the one hand, Formula 4 illustrates how the distance between the MCO capsule and Mars was originally modeled by NASA. That is, they assumed every distance value (*d*) as a fixed SI value (Km). On the other hand, Formula 5 shows that there were two standard systems of units (*u*) per value (*v*). This last formula implies the coexistence of different

[1] The corresponding units involved in the issue were *lb – sec* instead of *Newton - sec*

standard systems of units, being a more accurate model of the reality. The issue here is at the moment of using OWL because, as indicated in the previous subsection, OWL does not support ternary relations.

Referring to the MCO, the Mishap Investigation Board recommended, among other things, to:

- Verify the consistency of the use of units
- Conduct a software audit for specification compliance.

NASA implemented several policies after this event and the establishment of the international decimal metric system (SI) as their standard system of unit was one of these. Relevant to our research area, NASA developed the Semantic Web for Earth and Environmental Terminology (SWEET) [7] as an upper level ontology for Earth system science; this group of ontologies were implemented in OWL. SWEET aims at providing the different possible representations of a concept among different science communities. It contains several science concepts including partial support of metrological terminology. We want to stress this *partial* support because, as indicated above, NASA recognizes a unique standard, and that is reflected in SWEET, where only the SI standard is presented. Furthermore, NASA also developed the NASA Exploration Information Ontology Model (NExIOM) using OWL as well [8]. NExIOM has a larger and more technical scope than SWEET, and was proposed to support decision traceability, including reasoning and verification of product data (parts and technology) and mission data (segments, events and processes). Ensuring consistency analysis is one of the fundamental goals of NExIOM; in other words shared models and data must be consistent in meaning and value.

However, to really ensure the consistency of engineering data, we must require more than only representing this data with a control vocabulary within a domain. It is also necessary to perform reasoning over the given data set or ontology and find inconsistencies if any. In this vein it has been demonstrated that OWL has limitation for representing and reasoning on parthood relation [9] and procedural knowledge. Consequently, OWL has limitations to fully represent declarative knowledge in engineering project where such requirements are present.

The shortcomings of the SWEET and NExIOM ontologies (OWL) can also be extended to any product representation where modelers have to deal with n-ary relations, units, parthood relations and perhaps process representations. Such product modeling has been a concern for the Semantic Web Community for a long time, as is evidenced in the Product Modeling using Semantic Web Technologies report [10]. In that report only the SI standard system was taken into consideration, the working group recognized that OWL had not the capability to model certain parthood relations and that reasoning with OWL models of design was difficult. In fact nothing could be inferred from some of the described models because of the limitations they listed. Additionally, they claimed that some functional requirements representations were also not available in OWL.

We consider worthy mention before finishing this subsection that, although the use of the SI is a commonality in the approaches of NASA and W3C, for the modeling architecture that we will present in the following Sections, we prefer to follow the approach originally proposed in [11]. That is, we will not fix our model to a given standard system of units, but we follow the conceptualization provided by these authors.

Consequently, the SI and British systems will both be considered as instantiations of such a conceptualization.

3. Heterogeneity as a matter of fact

To deal with a scenario such as the one described in the previous section, and to fulfill the modeling requirements of engineering, we need at least an ontology language with higher expressivity level. But, when we move from OWL to a more expressive ontology language, we also face the risk of falling into undecidable scenarios. This is a common trade off that has been previously studied and for which frameworks have been proposed [12]. In this vein, if we can precisely divide the scenarios when OWL is expressive enough for our purposes from the ones where OWL is not enough, then we can introduce a formalism to represent our requirements and evaluate its decidability level. The coexistence of different modeling formalisms (languages) for ontology modeling is known as heterogeneity [13] and is the foundation of the architecture that we will introduce below.

3.1. The Common Algebraic Specification Language (CASL)

CASL [14] is an specification language that subsumes many of the constructors most commonly found in specification languages. CASL specifications can be composed at different complexity levels; those are:

a. - *basic* specifications, which allow modeling structures with predicates, subsorting with constrains, first order axioms and datatypes;

b. - structured specifications, which allow translating, reducing or extending basic specifications,

c. - *architectural* specifications, which define how the developed separated specifications have to be composed and their respective interfaces; and finally

d. - *libraries,* for storage and distribution of particular specifications [15].

Another important feature of CASL is that it belongs to a family of languages, rather than being an isolated language; this feature is depicted in **Figure 3-1**. Thus, sublanguages can be developed: for instances, CASL-DL [16] is a sublanguage of CASL restricted to SHOIN [17] (the respective logic of OWL 1.0 [18]). As a consequence, within CASL-DL, an alignment with OWL becomes available. Further, more extensions can be also developed for this language.

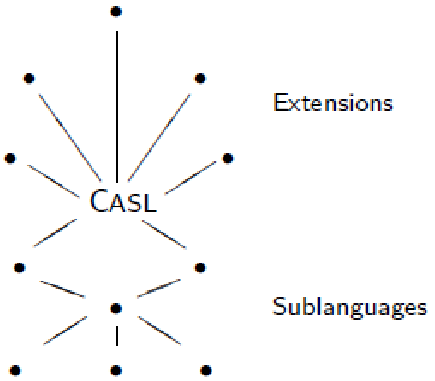


Figure 3-1 CASL Family Framework

CASL also enables proof obligations. That is, any theory previously defined in a specification, can be validated by means of obtaining expected consequences. In the following Subsection we will introduce the standard tool for CASL.

3.2. The Heterogeneous Tool Set

The Heterogeneous Tool Set (HETS)[2] is a software tool for parsing, analyzing, and proving specifications written in different logics. In addition, translations among several logics are also possible, and many provers and reasoners are available [19].
Figure 3-2 shows its architecture. There, under the Logic graph, CASL appears interconnected with its extensions and sublanguages. Here the connection with CASL and OWL-DL by means of the sublanguage CASL-DL is particularly relevant. It is also worth mentioning that CASL has a higher order extension called HasCASL [20], making available higher order logic, however a discussion of this extension is out of the scope of this paper.

HETS, as just mentioned, also allows translations among given logics. In [21] every logic supported by HETS and their corresponding translations are listed. The proof obligations, referred in the previous section, are supported in HETS by means of development graphs. Such graphs show specifications, their hierarchical structure and proof state. Regarding the proof state, after properly specifying the proof obligation, the user can interact with the graph, select a theorem prover and proves the correctness of the model.

In the following section, we retake our sample shape of **Figure 2-1** to show how to integrate our design ontology, originally developed in OWL, with CASL within HETS and validate certain design constrains.

[2] http://www.informatik.uni-bremen.de/agbkb/forschung/formal_methods/CoFI/hets/

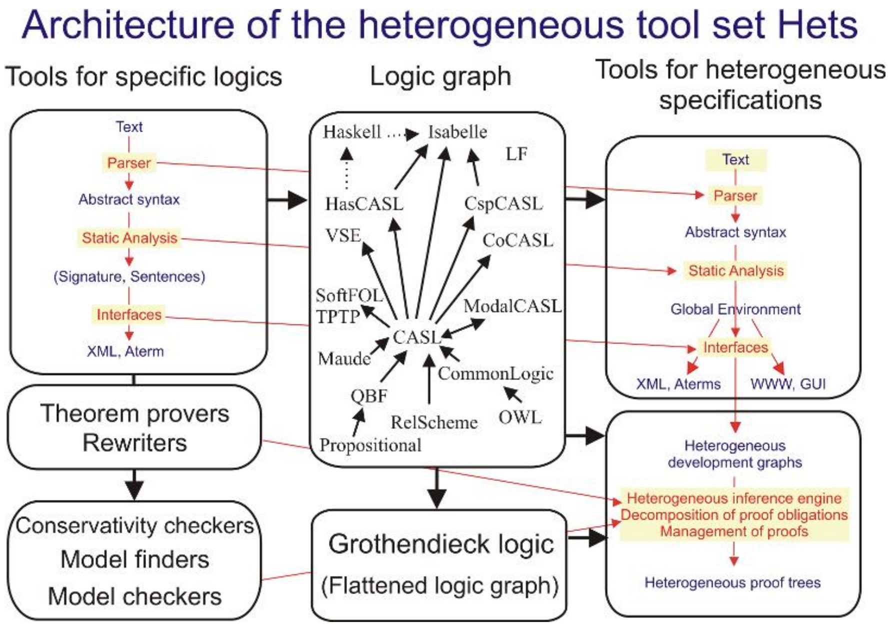


Figure 3-2 Architecture of HETS [21]

4. Heterogeneous Ontology in Manufacturing

In this section we consider that for dealing with use cases such as the ones indicated in Section 2 and to find inconsistencies by means of reasoning, an ontology language like OWL is not expressive enough. Nevertheless, the decidability level of OWL is a valuable feature that can be used in many other cases related to manufacturing, considering that reutilization of OWL ontology is a current goal of Ontological Engineering [22]. Therefore, we propose a heterogonous architecture to bridge the gap between representing and reasoning over manufacturing requirements in the Semantic Web.

Figure 4-1 presents our architecture. It is a modified view of the architecture presented in [10]. The main difference is in the inclusion of a heterogeneous layer. Such a layer allows us to deal with engineering requirements written in different languages. Our architecture contains:

1. A given Product Ontology (PO) which contains products definitions and features that can be represented and from which conclusions about their correctness can also be obtained.

2. Instantiations of this PO, which can be done by including specific designs into it.

3. A heterogeneous bridge for when higher expressivity with proof is available requiring, PO bridges with CASL by means of it.

4. Quantities, Units and Scale are also included as elements of this architecture, because they are a fundamental aspect in product description. There are links to the

Heterogeneous Bridge because we do not have a fixed standard system, but leave open the possibility of assigning the one preferred by the user to the give product.

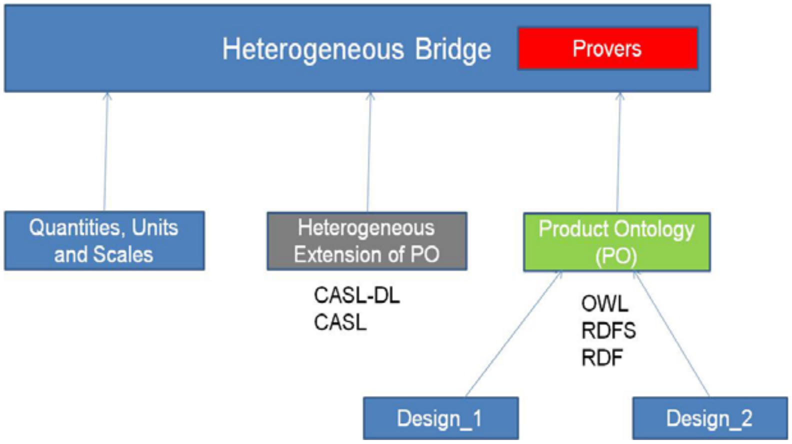


Figure 4-1 Heterogeneous Ontological Architecture for Manufacturing

In the next section, we present the preliminary results of implementing this architecture in order to validate features of mechanical designs.

5. Implementation and Preliminary Results

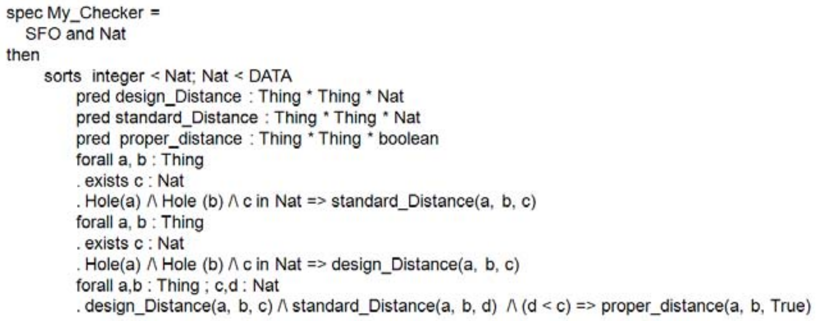
From Subsection 2.1 we consider the SMP fabrication issue in order to deal with it by means of our proposed heterogeneous architecture. Moreover, we take advantage of a previously developed Sheet Feature Ontology (SFO). This SFO representation and the quality checking of SMP were presented by us in [23].

Figure 5-1 shows the CASL code corresponding to the specification My_Checker. In the upper part, SFO is imported with its terminology into My_Checker. Later a group of ternary predicates are defined: `designDistance,` `standardDistance` and `properDistance`. Finally, we define that there will be a proper distance if the design distance is greater than the standard distance.

The proof obligation referred to in Subsection 3.1 was implemented in My_Checker. **Figure 5-2** shows this proof obligation for a set of instances from the SFO ontology. After loading the specification in HETS we obtained the windows showed in **Figure 5-3**. On the left hand side, at the top and bottom the development graph are presented. There, nodes named `Nat, SFO` and `My_Checker` correspond to given specifications, such nodes are shown in green color on HETS. More specifically, `Nat` comes from the library of `Numbers` (CASL), `SFO` comes from the SFO ontology (OWL) and `My_Checker` imports both (`SFO` and `Nat`). The node `My_Checker__E2,` shown in red on HETS, represents proof obligations. On the right hand side of the same figure, at the top view we can see the proving axiom.

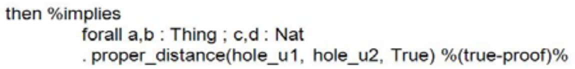
Finally, at the proof window, there is a list of all axioms present in the specification and the available theorem provers.

To finish our task, the theorem prover SPASS [24] was run on the proof node to assure the correctness of our instantiation. **Figure 5-4** depicts the result, which in this opportunity was proved, confirming that our design fulfills our requirements.



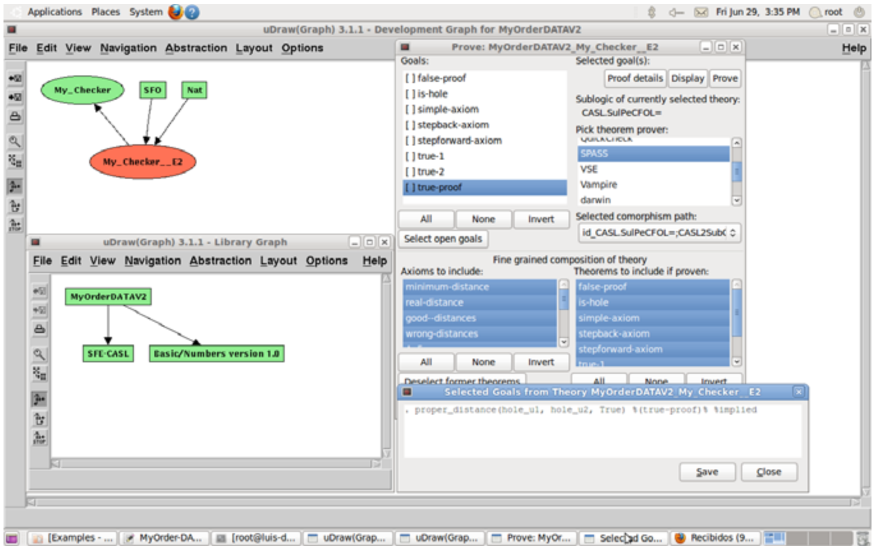
```
spec My_Checker =
   SFO and Nat
then
      sorts  integer < Nat; Nat < DATA
           pred design_Distance : Thing * Thing * Nat
           pred standard_Distance : Thing * Thing * Nat
           pred  proper_distance : Thing * Thing * boolean
           forall a, b : Thing
           . exists c : Nat
           . Hole(a) /\ Hole (b) /\ c in Nat => standard_Distance(a,  b, c)
           forall a, b : Thing
           . exists c : Nat
           . Hole(a) /\ Hole (b) /\ c in Nat => design_Distance(a,  b, c)
           forall a,b : Thing ; c,d : Nat
           . design_Distance(a,  b, c) /\ standard_Distance(a,  b, d)  /\ (d < c) => proper_distance(a,  b, True)
```

Figure 5-1 Partial View of My_Checker Spec



```
then %implies
           forall a,b : Thing ; c,d : Nat
           . proper_distance(hole_u1,  hole_u2,  True) %(true-proof)%
```

Figure 5-2 Instantiated Proof



Goals:
[ ] false-proof
[ ] is-hole
[ ] simple-axiom
[ ] stepback-axiom
[ ] stepforward-axiom
[ ] true-1
[ ] true-2
[ ] true-proof

Selected goal(s):

Sublogic of currently selected theory:
CASL.SulPeCFOL=

Pick theorem prover:
SPASS
VSE
Vampire
darwin

Selected comorphism path:
id_CASL.SulPeCFOL=;CASL2SubC

Fine grained composition of theory

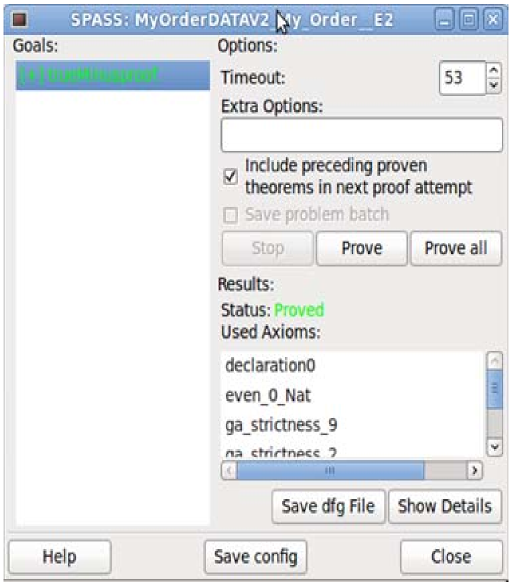
Axioms to include:
minimum-distance
real-distance
good--distances
wrong-distances

Theorems to include if proven:
false-proof
is-hole
simple-axiom
stepback-axiom
stepforward-axiom

Selected Goals from Theory MyOrderDATAV2_My_Checker__E2

```
. proper_distance(hole_u1, hole_u2, True) %(true-proof)% %implied
```

Figure 5-3 HETS views of My_Check



Goals:
[+] [illegible]

Options:
Timeout: 53
Extra Options:
☑ Include preceding proven theorems in next proof attempt
□ Save problem batch

Results:
Status: Proved
Used Axioms:
declaration0
even_0_Nat
ga_strictness_9
ga_strictness_2

Figure 5-4 Proof of Given Sample

6. Discussion and Future Work

The use of OWL and the Semantic Web Framework for engineering applications is an undeniable trend. Much research aims to exploit the advantages of reasoning and the possibility of deriving certain kinds of conclusions from knowledge bases. This characteristic has been proposed to facilitate designs verification and validation. However, we have presented here specific and general use case where the expressivity of OWL is insufficient for properly engineering modeling. Moreover, simply changing OWL for a more expressive language is not an effective decision, given that augmenting expressivity reduces decidability.

Consequently, since maintaining the decidability level of OWL and having a higher expressivity language when possible is desirable, we proposed a heterogeneous architecture for Semantic Manufacturing and product representation on the Web. This approach will help us to overcome the issues described in the use cases presented in Section 2. In our proposed architecture we integrate CASL and OWL by means of HETS. CASL allowed us to express some design restrictions as ternary first order formulas and write proof obligations. Furthermore, in HETS, the so called Development graph of the CASL specification was used to enable access to theorem provers which carried out the necessary proofs.

We have performed some initial experiment to verify simple designs within this architecture and we currently continue carrying further experiments in larger data sets and with more complex designs with the purpose of having a more accurate measure of the effectiveness of our proposed architecture. Moreover, we will work in interfacing our CAD - OWL design checker with HETS - CASL.

References

[1] Keqin Wang and Shurong Tong, “An Ontology of Manufacturing Knowledge for Design Decision Support,” in *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM ’08. 4th International Conference on*, 2008, pp. 1–5.

[2] S. Krima, B. Barbau, X. Fiorentini, R. Sudarsan, and R. Sriram, “OntoSTEP: OWL-DL Ontology for STEP.” National Institute of Standard and Technology.

[3] L. Ramos, “Ontological CAD Data Interoperability Framework,” presented at the SEMAPRO 2010, Florence, Italy, 2010.

[4] R. Radhakrishnan, A. Amsalu, M. Kamran, and B. O. Nnaji, “Design rule checker for sheet metal components using medial axis transformation and geometric reasoning,” *Journal of Manufacturing Systems*, vol. 15, no. 3, pp. 179–189, 1996.

[5] W3C, “Defining N-ary Relations on the Semantic Web: Use With Individuals.” 21-Jul-2004.

[6] NASA, “Mars Climate Observer. Mishap Investigation Board.” 10-Nov-1999.

[7] R. Raskin, “Guide to SWEET Ontologies.” 2011.

[8] NASA, “NASA EXPLORATION INFORMATION ONTOLOGY MODEL (NEXIOM) PRIMER AND VISION.” 16-Feb-2005.

[9] N. Krdžavac and D. Gašević, “Reasoning with part–part relations in a description logic,” *Knowledge-Based Systems*, vol. 24, no. 2, pp. 347–353, Mar. 2011.

[10] W3C, “Product Modelling using Semantic Web Technologies,” *Product Modelling using Semantic Web Technologies*. [Online]. Available:

http://www.w3.org/2005/Incubator/w3pm/XGR-w3pm-20091008/. [Accessed: 22-Sep-2010].

[11] T. R. Gruber and G. R. Olsen, “An Ontology for Engineering Mathematics,” presented at the Fourth International Conference on Principles of Knowledge Representation and Reasoning, Bonn, Germany, 1994.

[12] O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev, “E-connections of abstract description systems,” *Artificial Intelligence*, vol. 156, no. 1, pp. 1–73, Jun. 2004.

[13] J. Bateman, A. Castro, I. Normann, O. Pera, L. Garcia, and J.-M. Villaveces, “OASIS common hyper-ontological framework (COF).” 30-Jan-2009.

[14] E. Astesiano, M. Bidoit, H. Kirchner, B. Krieg-Bruckner, P. D. Mosses, D. Sannella, and A. Tarlecki, *CASL: The Common Algebraic Specification Language*. Springer-Verlag, 2001.

[15] *CASL Reference Manual*. .

[16] K. Lüttich, “Specification of Ontologies in CASL,” in *Formal Ontology in Information Systems – Proceedings of the Third International Conference (FOIS-2004), volume 114 of Frontiers in Artificial Intelligence and Applications*, 2004, pp. 140–150.

[17] F. Baader, *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, 2003.

[18] I. Herman, “Web Ontology Language OWL / W3C Semantic Web Activity,” *Web Ontology Language (OWL)*. [Online]. Available: http://www.w3.org/2004/OWL/. [Accessed: 21-Sep-2010].

[19] T. Mossakowski, C. Maeder, and K. Lüttich, “The Heterogeneous Tool Set, Hets,” *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 4424/2007, pp. 519–522, 2007.

[20] L. Schröder and T. Mossakowski, *HASCASL: Integrated Higher-Order Specification and Program Development*. .

[21] T. Mosskowski, C. Maeder, and M. Codescu, “HETS User Guide.” 26-Apr-2012.

[22] I. Normann and O. Kutz, “Ontology Reuse and Explration via Interactive Graph Manipulation,” presented at the SERES 2010, Shanghai, China.

[23] L. Ramos, A. García, and J. Bateman, “Ontology-Based Features Recognition and Design Rules Checker System,” presented at the OCAS 2011, as part of ISWC 2011, Bonn, Germany, October, 23th.

[24] C. Weidenbach, “Chapter 27 - Combining Superposition, Sorts and Splitting,” in *Handbook of Automated Reasoning*, Amsterdam: North-Holland, 2001, pp. 1965–2013.