Developing a Manufacturing Ontology Language Framework

Kaelan Sinclair

Tim French, Wei Liu
School of Computer Science and Software Engineering

Melinda Hodkiewicz
School of Mechanical Engineering

Abstract

Ontologies are machine readable representations of high level information (Corcho, Fernandez-Lopez & Gomez-Perez 2003). They contain several functionalities, including reasoning and communication standardisation. This has prompted the exploration of ontology development within the manufacturing domain.

A case defined within the manufacturing industry was utilised to aid in tailoring ontology development to manufacturing. It has been found that Subject Matter Experts prove to be the most fruitful source of information for knowledge elicitation in manufacturing, and that manufacturing concepts and rules are more easily represented by "heavyweight" ontology formalisations.

1. Introduction

1.1 Ontology Background

Ontologies are representations of high level information that is machine readable (Corcho, Fernandez-Lopez & Gomez-Perez 2003). This is achieved by utilising a formal language to represent domain knowledge as concepts, terms and rules. Ontologies have had successful implementations within the fields of biology with the Gene Ontology (Ashburner et al. 2000), and image classification with ImageNet (Deng et al. 2009). The successful use of ontologies within these areas has motivated the exploration of the use of ontologies within manufacturing.

Ontologies offer several functionalities that could prove advantageous. One functionality is that ontologies provide a platform supporting standardised communication and translation (Uschold & Gruninger 1996). For example, a packing team and shipping team within a fruit shipping business may refer to filled boxes as "containers" and "cases" respectively. An ontology developed for this business would be able to recognise the equivalence of both terms, and aid in preventing miscommunication between the two teams. Ontologies also provide the ability to perform reasoning and consistency checking over captured information (Feilmayr & Woess 2016). For example, the fruit shipping company determines whether a fruit box is to be shipped internationally or locally based on the grade of the fruit (where the grade is based on the fruit's shelf life). The ontology would be able to determine which boxes are for international shipping or local shipping via reasoning over the rule defining the minimum grade for international shipping, and classifying the fruit boxes respectively.

The Web Ontology Language (OWL) (Motik et al. 2009) is one of the most popular languages for ontology formalisation and is strongly based on the concepts of description logic (DL) (Kroetzsch, Rudolph & Hitzler 2008). DL is a subset of first-order logic, and covers various expressive families of logic (Kroetzsch, Rudolph & Hitzler 2008). The DL families are constructed by the combination of different logical operations which includes, but is not limited to, existential quantification, universal quantification, negation and inverse properties. This allows the ability to construct ontologies with different levels of logical expressiveness. Another important language in ontology development is the Semantic Web Rule Language (SWRL); a rule language created from the combination of OWL DL and Datalog RuleML, which are subsets of OWL and Rule Markup Language respectively (Horrocks et al. 2004).

1.2 Objectives

The primary objective of this project is to construct an ontology development framework tailored towards manufacturing. The development framework will describe the process of ontology formalisation for manufacturing in language that is easy to understand. The framework will consist of tools for recording domain specific knowledge and guidelines to translate this knowledge into a formal ontology.

2. Methodology

2.1 Technology Survey

This step involved a review of different ontology development technologies. These technologies were compared and contrasted based on the features of the technology. This was done to determine which of these technologies were most suitable for the development of ontologies for the scope of the project.

2.2 Knowledge Elicitation / Ontology Prototyping

This step involved determining methods which are effective at eliciting system and organisational knowledge from the client. Various methods of eliciting knowledge were attempted, including:

- A high level analysis of system documentation and databases, through reading and understanding the content.
- Text and pattern mining of documentation.
- Liaising with Subject Matter Experts within the client's organisation.

The elicited knowledge was organised into a hierarchical structure, to represent the initial classifications and relationships between the gathered knowledge. A visualisation was constructed to display the hierarchical structure, to aid the review of the structured knowledge by Subject Matter Experts.

2.3 Ontology Formalisation

This step involved formalising aspects of the knowledge gained in the previous step into different ontology formalisations. This was done to broadly determine the required logical expressiveness for an ontology to represent the system. Three ontology formalisations were constructed in this step:

- A lightweight ontology, restricted to existential quantification and disallowing universal quantification and negation.
- A heavyweight ontology with no restrictions on the use of OWL properties, but with no SWRL rules.
- A heavyweight ontology with no restrictions on the use of OWL properties or SWRL rules.

2.4 Ontology Validation

This step will involve the comparison of the ontology formalisations constructed in the previous step. Based on a system related case, the ontology formalisations will be queried and the responses will be assessed by Subject Matter Experts. The judgements on the ontology formalisation performance along with the potential extra functionality provided by the formalisation will be utilised to determine which is most suitable.

2.5 Ontology Development Framework Construction

Based on the results gathered from the previous steps, an ontology development framework will be constructed. It will detail the technologies and methods that proved most reliable at representing the client's system. The framework will be written using simple language, which is understandable by most people within the industry.

3. Results and Discussion

3.1 Technology Survey

Technology	OWL 2 support	Open source	Free	GUI	Ontology Visualisation	Plug-in reasoners	SWRL support
Protégé	Yes	Yes	Yes	Yes	Yes	Yes	Yes
NeON toolkit	Partial	Yes	Yes	Yes	Yes	Yes	No
Fluent editor	Yes	No	Not for commercial use	Yes	Yes	No	Yes

Table 1 Properties and functionalities of shortlisted technologies.

During this stage, nineteen ontology development technologies were researched to determine the suitability of the technology to the project. Of these technologies, a shortlist of three was determined. This shortlist includes: Protégé, the NeOn toolkit and the Fluent editor. A sample of the properties and functionalities of these technologies are represented in Table 2. These properties and functionalities were utilised to determine the technology's suitability to the project. As Table 1 displays, only Protégé satisfies all of the conditions. The NeON toolkit only partially satisfies OWL 2 support, and fails to have SWRL support. This limits the effectiveness of the NeON toolkit to formalise ontologies with the expressiveness required in this project. The inability for the Fluent editor to have plug-in reasoners prevents the editor from providing correct inferences over the ontology formalisations utilised within this project. Additionally, the Fluent editor is neither open source nor freely available for commercial use.

Given that Protégé was the only ontology editor to satisfy the requirements, it was selected for ontology formalisation.

3.2 Knowledge Elicitation / Ontology Prototyping

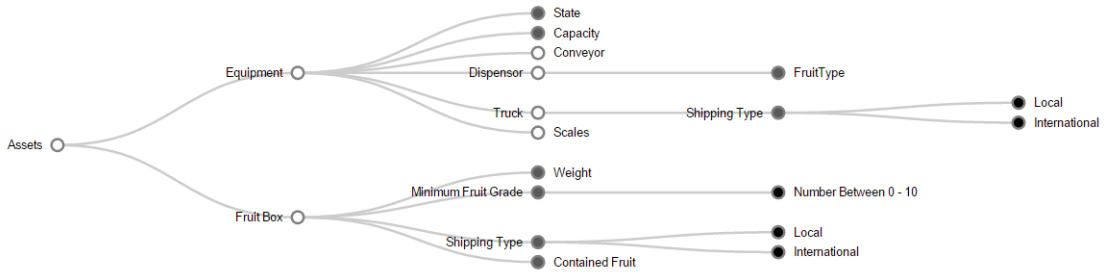

Figure 1 Knowledge store tree. White nodes are classes, grey nodes are properties and black nodes are possible property values.

3.2.1 High Level Analysis of System Documentation and Databases

The gathering of system knowledge through high level analysis of system documentation and databases was partially effective. The knowledge gained through this process was useful to create an initial store of knowledge about the client's systems. While capable of starting the process of knowledge elicitation, several caveats existed. The first of these was limited access to documentation and databases. This limited the ability to find the information and knowledge about much of the system required to be formalised. The second of the caveats is the convoluted nature of the documents. This meant that a significant amount of time was taken to piece together fragments of information gathered from the documentation and databases in order to represent the knowledge within the knowledge store.

3.2.2 Text and Pattern Mining of System Documentation and Databases

The use of text and pattern mining to elicit knowledge from the system was abandoned early on in the project. Much of this was due to the inability to find any significant patterns or rules within the documentation and databases, with the same caveats mentioned in the previous section providing some explanation as to the lack of significant results.

3.2.3 Subject Matter Experts

The use of subject matter experts proved to be the method that produced the most accurate and greatest amount of system knowledge. The process involved the use of the initial knowledge store mentioned in Section 3.2.1. This knowledge store was placed into a visualisation and utilised as a discussion facilitator for the system. This allowed the subject matter expert to highlight mistakes within the store, and to explore the areas within the store to a greater degree of accuracy. This process was repeated, with the knowledge store being updated, and then checked with the subject matter expert. This repetition led to a knowledge store that became significantly more detailed and accurate than from only utilising high level information from documentation and databases.

3.2.4 Construction of Knowledge Store and Visualisation

The process of converting the gathered knowledge into the knowledge store involved determining whether to represent elements of the gathered knowledge as a class, a property of

a class, or as a value for a property. The classes were also organised into a hierarchical structure. This hierarchy was based on the relationship of the concept represented by the class, as well as the associated properties, and recognising when a class inherits properties from another. For example, the concept of equipment would be a class, and the concept of a conveyor would also be a class. A conveyor is a piece of equipment, and inherits properties inherent to being an equipment. This defines the class conveyor as a subclass of equipment. The knowledge store was also split into two forms. One representing static information without time stamps, and is mainly utilised to define the properties of classes. The second is dynamic, that is information associated with a specific point in time, such as information associated with an instance of a class. A visualisation of the knowledge store was created utilising HTML, JavaScript with D3, and JSON, which can be seen in Figure 1. The visualisation was designed to allow for a visual representation of the classes, properties and potential property values. This visualisation was a valuable resource in facilitating discussion with subject matter experts when refining system knowledge previously elicited.

3.3 Ontology Formalisation

This section introduces the initial results for the "lightweight" formalisation, and "heavyweight without SWRL rules" formalisation.

For the "lightweight" formalisation, significant care was required in formalising the required aspects from the knowledge store due to the limited logical expressiveness available compared to "heavyweight" formalisations. This was primarily due to the inability to use universal quantification, such as formalising the concept that a fruit box instance "only" contains bananas. The "lightweight" formalisation could use existential quantification, for example, encoding that a fruit box instance contains "some" apples. However, this meant that concept negation was unavailable to the "lightweight" formalisation, as a negated existential statement is equivalent to a universal statement. For example, the statement a fruit box instance does "not" contain "some" apples is a universally quantified statement. This has made representing significant amounts of the knowledge store impossible within the "lightweight" formalisation. Another restriction was the inability to reason over data properties; typed data, such as strings and integers, that are associated to an instance. However, it was found that these properties could be represented as object properties; other instances associated to an instance. For example, instead of representing the processing rate of a piece of equipment as an integer, instead create an instance of the processing rate number and relate this instance to the instance of the piece of equipment. This solution carries inherent limitations, including the inability to perform basic data type comparisons, along with the potential of the convolution of instances within the ontology formalisation.

The "heavyweight without SWRL rules" formalisation achieves greater efficacy in representing the knowledge store formally due to the smaller number of restrictions. As such, the "heavyweight" formalisation had access to universal quantification, existential quantification, concept negation and the ability to reason over data properties. This allowed the ability to formalise more complex aspects of the knowledge store. One aspect that was formalised was the localised classification of equipment based on initial information, and an inherited global classification based on the local classifications of surrounding equipment. Initial attempts involved utilising a single hierarchy of classes to classify the equipment. This approach led to problems surrounding disjoint classes. To perform the initial localised classification, the classes in the classification hierarchy were required to be disjoint. However, with these classes disjoint, it prevented the global classification from applying to a piece of equipment if the global class is disjoint from the class the equipment belongs to locally. This

issue was addressed through utilising two classification hierarchies: one for the initial local classification, where the classes are disjoint; and a second, where the classes are not disjoint, which allows for reclassification.

4. Conclusions and Future Work

Ontologies are a formal representation of high level information that is machine readable. This technology has several beneficial functionalities surrounding communication and logical reasoning. The aim of this project was to construct an ontology development framework that is tailored towards manufacturing.

An exploration into several ontology development steps have been undertaken in relation to the manufacturing industry. For this project it has been found that Protégé is an appropriate technology, and that subject matter experts are a rich source of system knowledge. It has also been found that significant care is required when determining how to formalise the system with different logical expressiveness.

Future work within this project includes completing the formalisation process, completing the ontology validation step, and constructing the ontology development framework. Future work beyond this project includes further refinement of ontology development with more system knowledge, and in depth examinations into text and pattern mining for knowledge elicitation.

6. References

Ashburner, M, Ball, CA, Blake, JA, Botstein, D, Butler, H, Cherry, JM, Davis, AP, Dolinski, K, Dwight, SS, et al. (2000) Gene Ontology: tool for the unification of biology, *Nature Genetics*, **25** (1) pp. 25-29.

Corcho, O, Fernandez-Lopez, M & Gomez-Perez, A (2003) Methodologies, tools and languages for building ontologies. Where is their meeting point?, *Data & Knowledge Engineering*, **46** (1) pp. 41-64.

Deng, J, Dong, W, Socher, R, Li, L-J, Li, K & Li, F-F (2009) ImageNet: A Large-Scale Hierarchical Image Database, in *Cvpr: 2009 Ieee Conference on Computer Vision and Pattern Recognition, Vols 1-4*, pp. 248-255.

Feilmayr, C & Woess, W (2016) An analysis of ontologies and their success factors for application to business, *Data & Knowledge Engineering*, **101** pp. 1-23.

Horrocks, I, Patel-Schneider, PF, Boley, H, Tabet, S, Grosof, B & Dean, M (2004) SWRL: A semantic web rule language combining OWL and RuleML, *W3C Member submission*, **21** p. 79.

Kroetzsch, M, Rudolph, S & Hitzler, P (2008) Description Logic Rules, in *Ecai 2008, Proceedings*, **178**, eds M Ghallab, CD Spyropoulos, N Fakotakis & N Avouris, pp. 80-84.

Motik, B, Grau, BC, Horrocks, I, Wu, Z, Fokoue, A & Lutz, C (2009) Owl 2 web ontology language: Profiles, *W3C recommendation*, **27** p. 61.

Uschold, M & Gruninger, M (1996) Ontologies: Principles, methods and applications, *Knowledge Engineering Review*, **11** (2) pp. 93-136.