Ontology Engineering for Product Development

Henson Graves
Lockheed Martin Aeronautics Company
Fort Worth Texas, USA
henson.graves@lmco.com

Abstract. This analysis is to identify requirements for a Description Logic (DL) to reason about product descriptions and other information related to product development. The DL is intended to be used by software decision support systems to assist in verification of product satisfaction criteria. Decision support requires the capability to express assertions about products and assemble evidence to demonstrate that the assertions are satisfied. Satisfaction criteria involve processing large amounts of data and so present significant scale up challenges for software systems. The analysis indicates need for the expressiveness found in OWL-DL, as well as additional descriptive expressiveness and inference rules. Experience with large scale information systems indicate that reasoning capability can be scaled up to serve large user communities. The use of reasoning capability can have significant cost and savings impact on product development

Keywords: Description Logic, OWL-DL, Product Development, Ontology Engineering.

1 Introduction

Aerospace product development lasts many years and produces millions of data artifacts. Product development programs perform technical assessment to determine the state of product development and to verify that properties of the development and the product are satisfied. Assessment at program milestones takes the form of verifying specific success criteria defined for the milestone. Criteria may express assertions regarding that an air vehicle meets weight and performance requirements, or assertions regarding the vehicle’s qualification to fly. Assessment requires expressing, deriving, and establishing evidence for conclusions. Currently assessment is primarily a manual effort of assembling and combining data to support conclusions. The capability to represent and reason about product requirements, designs, analysis, and test results can have significant reduction in product development lifecycle cost and schedule.

This analysis is intended to identify requirements for a Description Logic (DL) to describe and reason about product information. The DL is to be used to represent requirements, design, analysis, and test information, and express assertions about the objects represented. The assertions include both assumed facts and derived results. The DL is intended to be used by decision support systems that require large amounts of data to establish assertions.

Describing and reasoning about products fits the Description Logic paradigm, as exemplified by OWL DL [4]. Analysis of the expressiveness requirements indicates that class and relation constructions can be used to describe requirements and design constraints. OWL DL is an attractive candidate with its formal semantics and its position within the Semantic Web [5] stack of standards. However, there are issues regarding expressiveness and there is a need to represent and check deductions as a part of the services offered by product decision support. Scaling up decision support systems to use massive amounts of data and provide effective reasoning capability is a significant challenge. The challenge to provide reasoning services is on top of an already significant challenge to provide access, location, and data exchange in large scale distributed data systems. However, there is good evidence that Semantic Web standards enable syntactic interoperability, i.e., the capability to access, exchange, and discover data. Reasoning in support of verification of design criteria can scale as the tasks to be automated consist primarily of matching and filling in specific assertions in evidence trees. An evidence tree is a predefined outline of the information required to establish an assessment criteria.

2 Background

2.1 Decision Support

Establishing that a product with a certain design configuration satisfies a weight requirement is currently primarily a manual effort of assembling and combining evidence to support conclusions. The production of evidence often involves chains of inputting data to program processes and combining results to produce new data. In general product assessment includes checking assertions of design consistency, maturity, producibility, and requirements verification.

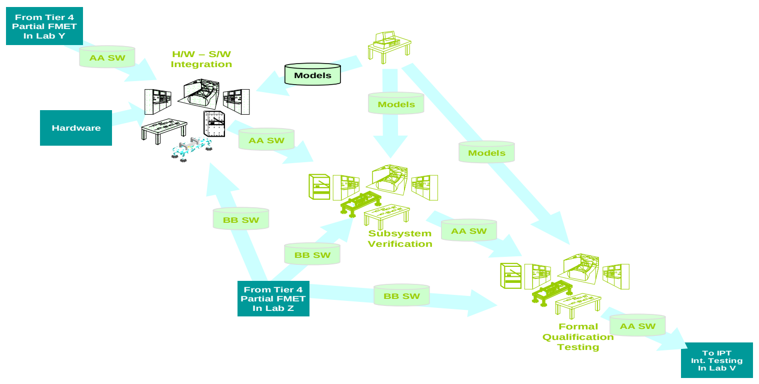
From Tier 4 Partial FMET In Lab Y
AA SW
H/W – S/W Integration
Models
Hardware
Models
AA SW
Models
BB SW
Subsystem Verification
AA SW
BB SW
From Tier 4 Partial FMET In Lab Z
BB SW
Formal Qualification Testing
AA SW
To IPT Int. Testing In Lab V

Fig. 1. The figure illustrates a graph of activities and intermediate data results that lead to a conclusion that flight readiness criteria have been satisfied. Intermediate data is continually updated and so results may depend on obsolete data leading to erroneous conclusions.

Decision support for product development requires capability to answer questions (whose answer can not be looked up in a database) and provide evidence in the form of data to justify the truth of the answer. For example, to obtain certification of flight readiness for an air vehicle, a large about of design analysis and test data is required. However, the structure of evidence trees is usually defined as part of development effort planning. The automated assistance needed is primarily to the find and match data with specific nodes of an evidence tree.

2.3 Reasoning

The decision criteria used in product development can be expressed as assertions about products that are assumed to satisfy specific descriptions, e.g., as having a specific component structure. Verification of decision criteria is verification of assertions. Reasoning about product information requires the capability to represent product requirements, design, analysis, and test information in a common language for description with logic to verify conditions for design completeness, maturity, and producibility, as well as, for product qualification and requirements verification. The services include assemblage and evaluation of evidence for assertions.

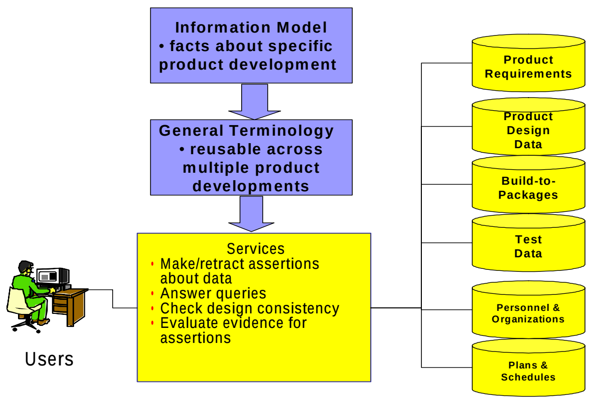
Information Model
• facts about specific product development
General Terminology
• reusable across multiple product developments
Services
Make/retract assertions about data
Answer queries
Check design consistency
Evaluate evidence for assertions
Users
Product Requirements
Product Design Data
Build-to-Packages
Test Data
Personnel & Organizations
Plans & Schedules

Fig. 2. Decision support systems for product development serve large communities of users and utilize large volumes of data. Decision support reasoning services go beyond, but depend on, capability to access, exchange, and discover information (syntactic interoperability). The decision support system in the diagram uses a Description Logic to represent general terminology applicable to product development across an entire domain, as well as, facts specific to an individual product development effort.

Automated support of flight readiness certification, for example, requires verifying that specified test cases have been covered, but not necessarily evaluating the data supporting each specific test result. The data supporting verification is often in a machine readable representation language with well understood semantic conventions. The verification conditions could be expressed in a logical language with well defined inference rules. However, machine processable statements of verification conditions are currently not uniformly used. The information systems used to support product development are naturally evolving from storage and retrieval systems to decision support systems that derive assertions from assumed facts.

3 Analysis

A product development semantic framework fits the Description Logic paradigm. Constructions are needed for individuals, classes, and relationships, and assertions regarding membership and class containment. The facts about a specific product development effort are an ABox. The ABox consists of assertions about the requirements and design description classes. The TBox terminology is the terminology common to a product domain.

3.1 Descriptions

Product requirements and designs can be represented as classes defined in terms of their properties and relationships. To show that design structure can be represented in OWL DL start with a simple statement that any air vehicle design has an airframe component and an avionics component and that the avionics component has a radar component. Structural component hierarchies description are common in UML and SysML diagrams.

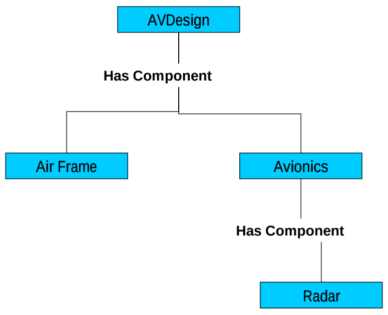

Fig. 3. The component hierarchy is a typical top level description of a component-subcomponent hierarchy can be described as a class defined in terms of a component relation.

OWL DL class constructions can be used to represent structural hierarchies. The *AVDesign* class is specified as the intersection of classes specified by the OWL DL *SomeValuesFrom* construction using a *HasComponent* relation with additional classes for *Air Frame*, *Avionics*, and *Radar*. The approach outlined here is described in [7].

AVDesign = SomeValuesFrom(hasComponent, Air Frame) INTERSECT (1)
SomeValuesFrom(hasComponent, Avionics) INTERSECT
SomeValuesFrom(SomeValuesFrom (hasComponent, Avionics),Radar)

The *AVDesign* class describes objects that have the three components from the respective classes and that satisfy the respective component relationships. Of course a full description of the design will specify properties such as the form, fit, and function of each component.

Requirements are described similarly. For example, an air vehicle requirement that the weight is less than 33 thousand pounds can be represented in terms of properties on a relation. For the weight property the range of the values are restricted numeric data types and so a value restriction construction is used.

WeightRequirement = SomeValuesFrom (hasWeight (number < 3300)) (2)

There are many kinds of estimates and measurements of weights and there are established procedures to calculate the different weights. For example, parametric weight estimates sum up estimated weights of all of the components where the estimated weights of components are defined by volume multiplied by material mass. Verifying that an individual satisfies a weight requirement requires verifying that the appropriate procedure has been used to calculate the weight, as well as verifying the weight number.

3.2 Assertions

An assertion that a product satisfies a requirement or design configuration can be expressed as an assertion of class membership. For example, the assertion that an individual satisfies a 33000 pound maximum weight requirement is expressed by the membership statement:

a : WeightRequirement (3)

The assertion that a product satisfying the design description also satisfies the weight requirement can be expressed as:

a : AVDesign IMPLIES a : WeightRequirement (4)

To establish that a member of AVDeisgn is a member of AVRequirements requires complex design analysis that can be only partially captured by the Decision Logic.

The precise statement within logic of satisfaction conditions makes a major contribution to verification that assertions are satisfied. In practice accredited procedures are used to establish the results and the conditions required to establish membership results can be expressed by inference rules.

3.3 Inference

Assertions are based on chains of evidence leading back to assertions that are taken to be facts. Each assertion is produced as result of process executions and requires assessment of the evidence that the process execution producing the result is a valid. The concluding assertion depends on the validity of each process execution and its input. Verifying that an individual has some property generally requires that logical rules be used to evaluate the evidence. Inference based on rules used to express validation conditions can be used to check evidence for assertions. For example, an estimated weight requirement can be stated informally as:

a : WeightRequirement IF a : AVDesign AND the sum of the weights of the components is less than 3300 pounds where the weights of the components is calculated as mass multiplied by volume. (5)

The importance of the rules is their ability to state precisely the satisfaction condition of the property. To state inference rules, such as the weight requirement, requires a syntax and constructions for applicative functions, e.g., to describe the weight summation function. Function types can be added to a DL type structure. To use the inference rules to derive or verify assertions about properties such as weight requires logic variables for individuals and classes.

3.4 Description Logic for Product Development

OWL DL is a good starting point for a Product Development Description Logic. Classes and relationships can be used to describe products and their context of use. The formal semantics provides a precise notion of meaning. The representation of classes and relations using XML schemas facilitates the exchange of data between systems. However, DL classes and relations may be insufficiently expressive for product descriptions and their operational contexts. In addition data types are needed for a functional/process calculus (e.g., types for lambda calculus terms). Decidability is not the major criteria for a Description Logic for product development. Rule based logic with real variables is needed, as well as, an explicit proof theory.

4 Scale Up Considerations

This analysis is the direct result of six years of experience in developing and implementing a technical data management system [1]. This system, called the Resource Access System [2], is used by team members from multiple companies and organizations on a large aerospace program; the data repositories are geographically distributed. The architecture of this system uses an ontology stored in a separate repository to organize and reference data residing in multiple repositories. The Resource Access System server uses the ontology to provide access and discovery across multiple repositories.

4.1 Information Interoperability

Information systems to support Product development depend on the capability to access, exchange, and discover information. However, this capability is difficult to achieve because the information space is characterized by:

- Millions of data items produced and revised over long time periods
- Data in multiple formats produced by many application tools
- Large number of independent data repositories, each with its own organization and interface
- Redundant information in different repositories
- Little support for users to access and locate data without their having knowledge of the specific storage repositories
- No standard terminology for the product development domain

The Semantic Web standards enable access, exchange, discovery and offer promise of enabling use of languages with formal semantics in support of decision making. The table below identifies the standards and describes role of each.

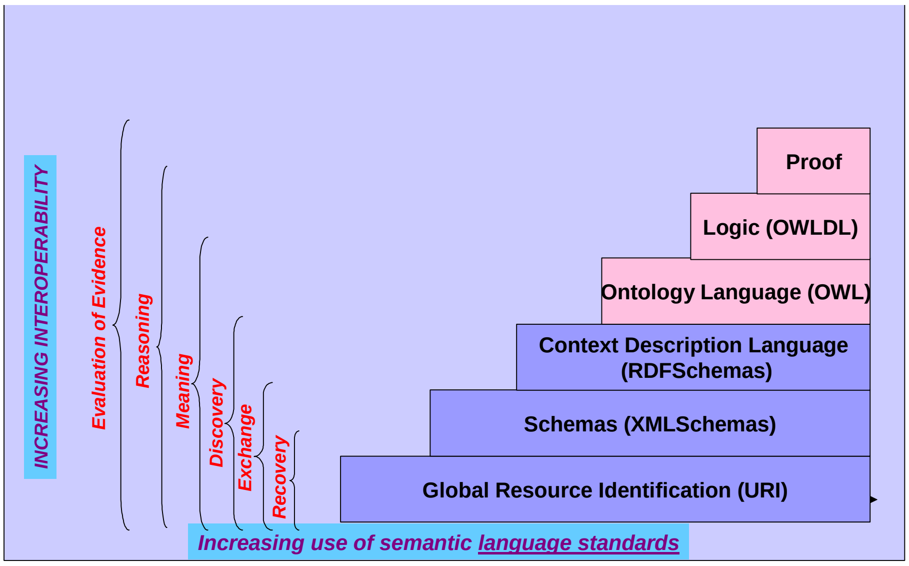
INCREASING INTEROPERABILITY
Evaluation of Evidence
Reasoning
Meaning
Discovery
Exchange
Recovery
Proof
Logic (OWLDL)
Ontology Language (OWL)
Context Description Language (RDFSchemas)
Schemas (XMLSchemas)
Global Resource Identification (URI)
Increasing use of semantic language standards

Fig. 2. A Semantic Web Initiative objective is to achieve semantic interoperability between people and computer systems through a stack of representation standards that apply above network connectivity to achieve semantic interoperability. Increasing use of Semantic Web standards correlate to solving specific aspects of information interoperability. Implementation of the Semantic Web standards provides good evidence that syntactic interoperability (access, exchange, and discovery) can be achieved. However, achieving full semantic interoperability is yet to be achieved.

4.2 Reasoning Scale Up

Based on prior experience with implementing large scale information systems for product development [3] using a reasoning engine with a Knowledge Base is feasible, even with terabytes of data involved. The volume of data is primarily the property values of complex data types, e.g., a geometry file. The ABox will need to accommodate on the order of 2 to 10 million assertions. However, these assertions can be stored in a separate repository which is accessed by the decision support server on behalf of the task of an individual client. The TBox will need to accommodate only hundreds of classes and relations. This architecture has proven to effectively support at least several thousand users.

References

1. Graves, H., Hollenbach, J., Barnhart, M.: *JSF Authoritative Modeling Information System (JAMIS) Architecture (03S-SIW-040)*, Simulation Interoperability Workshop, September 2003.
2. Graves, H., Campbell, R.: *Joint Strike Fighter Program Technical Data Integrity Management*. National Defense Industrial Association (NDIA) Modeling and Simulation Working Group, June 7, 2006.
3. Graves, H., Hollenbach, J.: *Using Technical Data For Program Assessment,* Presentation: National Defense Industrial Association (NDIA) 9th Annual Systems Engineering Conference, October 2006.
4. Heflin, J.: *OWL Web Ontology Language Use Cases and Requirements*, 10 February, 2004.
5. Horrocks, I.: *Ontologies and the Semantic Web.* Needham Lecture. http://www.epsg.org.uk/pub/needham2005/Horrocks_needham2005.pdf, 2005.
6. McGuinness, D.L., van Harmelen, F.: *OWL Web Ontology Language Overview*. http://www.w3.org/TR/owl-features/ (3/29/2005).
7. Price, D. and Bodington, R.: *OWL, Description Logic and Systems Requirements Satisfaction*, 8th NASA-ESA Workshop on Product Data Exchange, 2006.