Rule Requirements for Product Ontologies

Authors

Henk Meeter
Integrity GmbH
henry.meeter@integrity.vc

Henry Kurz
Integrity GmbH
henry.kurz@integrity.vc

Introduction

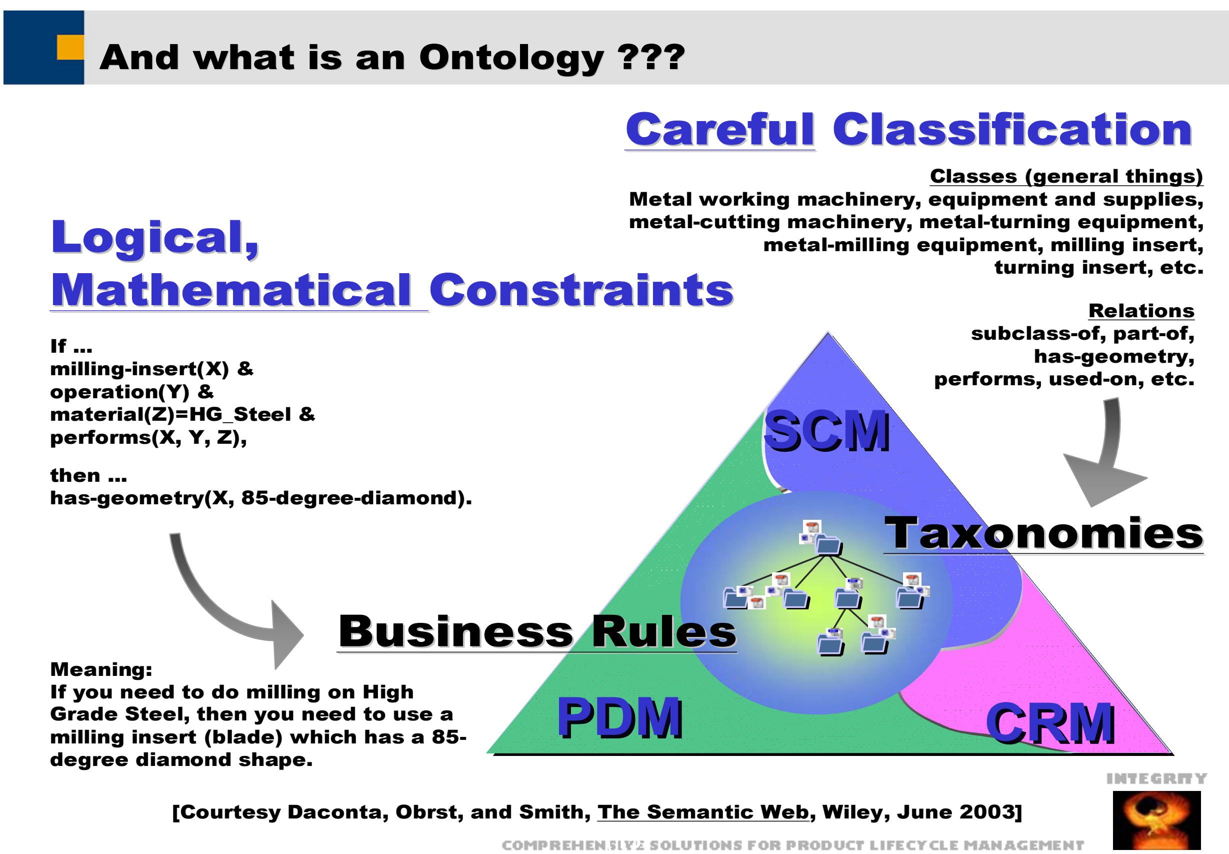
It is our belief that, while the standardization on the various forms of OWL is not only useful but absolutely imperative, there remains nevertheless a significant impediment, not only to its adoption specifically, but also to the adoption and exploitation of the Semantic Web in general.

This is the lack of a similar standardization on Rules.

And while we believe that this is obviously true relative to the goal of achieving a thorough-going Service Orchestration, we would assert that it is at least as true, if not even more so, in the area of product information and logic.

Which after all is the core around which any discussions of Service Orchestration – at least relative to e*Commerce – must necessarily revolve.

Our point can be summarized, we believe, in the following illustration, taken from a presentation delivered to the Semantic Technology Conference in San Francisco on March 10, 2005.



What it shows, at a high-level and elaborated on a relatively concrete example (taken from Daconta, Obrst, and Smith, The Semantic Web, p. 211), is a sample product conceptualization.

And within that we note the centrality, not only of Taxonomies (which business people, especially those in product information-related disciplines, might understand more immediately as "careful classification"), but also of Business Rules.

And these in turn we might elaborate as not only those Business Rules that might be expressed as purely logical constraints, but also, and arguably even more importantly, those which can only be expressed mathematically.

Scope

Product information is in almost all of its forms descriptive rather than algorithmic. Because of this we restrict the scope of our requirements to Rules as declarative constraints, rather than Rules as elements in the build-up of procedural algorithms.

Positions

1.0 The public representation and interchange of Rules

We do not have a position on the public representation and interchange of Rules, at least not to the degree that this "public representation and interchange" is something that needs to occur between humans.

These strange creatures have, after all, been more (or less) adequately describing and interchanging Rules with each other for thousands of years, and it is difficult to imagine improving either on their preferred media of the narrative word, the drawn symbol, or the mathemtical equation (a relatively more recent development).

Nevertheless we do have a position when it comes to the representation and interchange of rules between machines. And while we cannot say that our position is buttressed by thousands of years of more or less (and more often less) successful history, what we can say is that the past several decades have offered up a limited set of possibilities for what machines are most likely to be able to digest.

1.1 Class Atoms

First, we suggest, there need to be Atoms, which represent Classes (or ultimately Individuals of Classes). These are the most elementary pieces of information, of Facts, we can postulate about the world. We can think of them either as being really true Facts about a really true World, or we can think of them simply as Assertions (of Fact) about a World we believe in (true, in other words, only to the extent that we believe them to be true).

In the World of Telecommunications, an Atom might be a Class representing "Base Station." Or it might be a Class representing the Attribute "Maximum Configuration." Or it might even be a Class representing the Value "3X12," where "3X12" itself represents, in shorthand, three "Sectors," with up to twelve "Transceiver Units" per "Sector."

In order to reason about these Atoms effectively, especially when that reasoning might entail distinguishing between multiple (different) Individuals of the same Class simultaneously, we need to be able to bind these Atoms to Variables.

1.2 Property Atoms

Secondly, we suggest, there need to be Atoms representing Properties, or Relationships between Classes. Such Atoms have actually been represented, almost from time out of mind (at least within the IT epoch) as Predicates, which are themselves ultimately functional in character. Like Class Atoms they can be said to be either really true Facts about a really true World, or simply Assertions of Fact about a World we say we believe in.

In the World of Telecommunications, such an Atom might be a Predicate representing the Fact that a "Base Station" has the Attribute "Maximum Configuration." And that might in turn be augmented with an Atom or Predicate representing the Fact that the "Maximum Configuration" Attribute has the Value "3X12".

In natural language we would say, quite straightforwardly, that the "Base Station's Maximum Configuration is 3X12." And if we had, along with our spoken words, an understanding of the surrounding contextual concept of what a "Base Station" is, we would know exactly what we were talking about.

But since we are not talking about natural language, but rather about language that needs to be intepreted by machines, we have to say what we just said far more formally and functionally (with the emphasis, perhaps, more on "functionally" than "formally").

In order for a machine to unambiguously understand that statement, we would at the very least have to state it in the form of functional Predicates, for example as follows:

IsCharacterizedBy(BaseStation, MaximumConfiguration),

HasActualValue(MaximumConfiguration, 3X12)

Again, however, in order to ensure that we are talking precisely and particularly about certain Individual "Base Stations" with "Maximum Configurations" and "3X12's," and especially when we were doing so in a context where in the very same Rule we might be talking about other "Base Stations" too, and other "Maximum Configurations," and other "3X12's," we would need to ensure that we could bind those Atoms to Variables.

1.3 "Other" Predicate Atoms

Thirdly, we suggest, there need to be Atoms representing Predicates for performing two different types of Declarative Rule. They are Predicates for performing Math, and they are Predicates for performing descriptive Operations on Sets (specifically, that is, for Intersections).

Without belaboring the Telecommunications example we've already cited, let's just say that we need the ability to represent statements of the type A = B + C, and we anticipate needing to do so with something very similar to the Predicate Add(A, B, C), which in this case should achieve the very same result.

1.4 "Before" and "After"

Fourthly, we suggest, there needs to be the abiolity to state both Antecedents and Consequents (i.e. the condition on which a certain truth is contingent, and the contingent truth itself, respectively), and we nned to do so in such a way that a particular Antedecent is understood, and unambiguously understood, to be the condition on which the Consequent holds true.

2.0 The format by which Rules should be represented and exchanged

Here we have a Position, and it is very simply. There is an already existing submission to the W3C for "a Semantic Web Rule Language Combining OWL and RuleML" (fittingly shortnamed SWRL), and it is described in a W3C Member Submission dated 21. May, 2004.

What we see at this point is that SWRL not only addresses all the points we have outlines above, but it does so with the significant advantage (over the other "languages" cited in the Rules Workshop "Call for Participation") of being thoroughly – rather than half – baked.

In other words, not only does it offer (yet) another alternative syntax for stating Rules (actually not so alternative, really; nothing more, after all, than RuleML with stated mathematical capabilties), it also binds that syntax to OWL, and thereby to both expressible, and processible, XML.

3.0 The types of application we anticipate (for starters) requiring such representation and interchange

As is demonstrated by the illustration shown in the Introduction, above, the applications on which we are currently most focused are those have to do with Product Information.

These include but are not restricted to CAD tools, Design Advisors, Marketing Analytics, and Product Configurators, as well as Product Catalog Search Engines (arguably a form of Product Configurator).

All require, currently, their own ideosyncratic syntax for information, notwithstanding the fact that the information they deal with is conceptually identical.

All of these applications, because descriptive, would benefit from the ability to interchange between themselves, and other equally descriptive "Product Information Applications," comprehensively articulated ontologies.

And by "comprehensively articulated ontologies" we mean, obviously, not only the half (or less) of the product story represented – or even representable – within taxonomies. We mean ontologies that are the whole of the product story. Taxonomies completed, and not just augmented, with declarative Rules.

In the attached XML you will find an ontology representing a Base Station. This ontology was originally developed for the SAP Product Configurator, before we knew that ontologies existed – at least outside of Philosophy.

Interestingly, the syntax required for the Rules was very "Prolog-ish," to be expected, probably, for an application growing out of the same AI roots as all the other reasoning languages.

The main point, however, is that because the SAP Product Configurator is fully object-oriented and declarative, we were able to "translate" its syntax directly into OWL complemented with SWRL.

At the same presentation to the Semantic Technology Conference in San Francisco from which we tool the illustration above, we demonstrated generating this ontology, completely represented on OWL/SWRL, driectly from a fully graphical modeling (Ontology Editing) environment, transforming it into Prolog Predicates, and running it through a prototypical forward-chaining reasoner we had implemented to prove the point.[1]

It is comprehensive. That is, it shows not only the relatively more trivial Rules by which some Property can be determined to be mathematically equivalent to another Property, or mathematically equivalent to the result of dividing some other Property by yet another Property. It also manages the conditions under which certain Individuals (Base Station Cabinets, for example, or actual Transceiver Units) exist.

And on top of that it demonstrates a concept for the evaluation of Truth Tables (or Cladistics Tables, if we want to keep our terminology apace with the currently trendy borrowing from Biology), using List Intersections based on Anonymous Classes and Table Rows as enumerated Individuals.[2]

---

[1] That Visio file, representing the Base Station and its logic, is also attached. The Visio file, of course, shows nothing more than the shapes and lines which can be inferred to have meaning, by virtue of there somehow having been generated from them a rather full OWL/SWRL XML.

The real meaning, however, is contained in an RDF file – not included – and the RDF file and the Visio file are kept synchronized by an application called SemTalk™, which is offered by a company called Semtation, in Germany.

[2] Although, admittedly, "comprehensive" is just slightly overstated. In reality it is somewhere between 75% and 85% complete, or "three-quarters baked."

What the Rules are all missing of course are two Atoms: the Class Atom for the Class **CharacteristicValue**, and the Property Atom for **IsValueInRangeOf** relationship between the Classes **CharacteristicValue** and **Characteristic**.

This is because we started our development activity with the assumption that when humans write constraints like Table.Height = Tabletop.Thickness + Tableleg.Height, they do not expect to be forced to say something like Table.Height.Value = Tabletop.Thickness.Value + Tableleg.Height.Value.

4.0 Conclusion

As we stated above, in the Introduction, we are convinced that the Semantic Web has enormous potential.

Unfortunately, however, we are also convinced that, before it can reach that potential – and especially when it comes to realizing the idea, alluded to in the Daconta, Obrst, and Smith book (The Semantic Web), of a fabric of ubiquitous inference engines, dedicated knowledge configurators, reasoning over ontologies that themselves contain the logic necessary to process them – it needs to include Rules.

Therefore we believe that it is imperative that the W3C not only standardize, as it did with OWL, in a Semantic Web Rule Language, but that it do so as quickly as possible.

At the moment we are most in favor of SWRL, because it seems to be the one most fully "along."

But it could just as easily be something else, as long as it combines Antecedents and Consequents in single Rules, as long as it provides the ability to declaratively state both mathematical and set operations, and as long as it ties, seamlessly, to OWL.

And it only dawned on us sometime later – at the very last development minute, actually, when there was no longer time to change it for our demonstration – that the OWL/SWRL needs to reflect, not what humans expect to write, but rather, what machines need to understand.