EasyChair Preprint

№ 12383

Criterion of the Logical Completeness for Ontologies

Olegs Verhodubs

March 4, 2024

Criterion of the Logical Completeness for Ontologies

Olegs Verhodubs
oleg.verhodub@inbox.lv

Abstract. The number of ontologies is increasing day by day. Each ontology describes its own subject area, but this does not mean that there is only one ontology for each subject area. On the contrary, due to the decentralization of development centers, there may be several ontologies for the same subject area. In this situation, we need tools for assessing the quality of ontologies. Such a tool may consist of several criteria by which the ontology can be assessed. This paper examines one such criterion, namely the criterion of the logical completeness of the ontology. This criterion namely "logical completeness" differs from the criterion "completeness" for ontologies, described in the papers earlier. The main difference is the inconsistency in ontologies, which can be identified logically.

Keywords: Ontology Completeness, Ontology Quality, Semantic Web

I. Introduction

The concept of completeness has been known for a long time, and concerns a variety of areas of human activity. If for the world of programming and algorithms completeness (the completeness of the algorithm) is extremely important, then for some areas of real life this cannot be said so categorically. For example, an interrupted but not completed relationship between people can be an incentive to create a work of art with a claim to greatness. Unplanned events often disrupt the completion of something. It is even more surprising when some people deliberately do not complete something, establishing incompleteness a principle that they subsequently strictly followed. As you know, the great Leonardo da Vinci did this, never completely finishing any of his works (mainly paintings). In his view, incompleteness is the main quality of Life, therefore, in order to be a living product, it must be incomplete.

There is one interesting effect (which can easily be considered a law) associated with incompleteness. It lies in the fact that any incompleteness strives to become complete. That is, figuratively speaking, any incompleteness is a certain potential, potential energy that strives to turn into kinetic energy. The physical analogy can be replaced by a colloquial one: the void will definitely be filled. This does not mean that you need to urgently run and complete Leonardo's paintings. The potential of incompleteness in Leonardo is aimed at achieving completeness, by sucking out feelings, emotions and sensations from the consumer of his paintings, becoming one with the consumer and thus coming to life. Examples of the operation of this law can be found almost everywhere, including in biology, for example, when one organ partially takes over the functions of another. In general, there is nothing new here, because it has been known for a very long time that "Natura abhorret vacuum". This is this very law, expressed extremely briefly.

Scientifically interesting that the effect or law mentioned above permeates not only natural, but artificial objects, that is, objects created by human. And even ontology, which is not only a completely artificial, but also a virtual formation, obeys this law. Ontology is one of the elements of the Semantic Web concept, which reflects a certain part of the surrounding reality and divides this reality into concepts and relationships between these concepts. The quality of ontologies can vary and has traditionally been determined by how complete the ontology is within the domain. That is, does the ontology contain all the concepts and all the connections between them in a given subject area? The problem is that, without being an expert in a certain subject area, and without other ontologies for comparison, we cannot say how fully a given ontology reflects the subject area, namely, whether all the concepts and connections between them are present in the ontology. However, it turns out that not everything is so pessimistic and it is possible to analyze the quality of ontology, armed only with everyday logic and common sense. The structure of the ontology may suggest that it is clearly missing something. Alternatively, we can paraphrase by saying that ontology has hidden elements. Some of the hidden elements of the ontology have already been mentioned earlier, but this was done in the context of generating rules from the ontology for an expert system [1]. Hidden or implied elements (classes, relationships, attributes) of ontology are those elements that are not described in the ontology, but which are implied based on logic. Hidden elements of ontology negatively affect its quality, because they represent the emptiness that strives to be filled. It is good if hidden or implied elements of the ontology will be added to it in the process of the evolutionary development of the ontology. And vice versa, it is not very good if the lack of some elements of the ontology will be felt during the use of the ontology and it will be compensated with the help of some third-party capabilities (for example, by creating adjacent ontologies, etc.). This paper discusses various types of hidden elements of the ontology, and also develops a special criterion by which the quality of the ontology is determined, i.e. how many such hidden elements the ontology has.

This paper is organized into several sections. The first one describes types of hidden elements in ontology. The next one is deicated to the formula for calculation the criterion of the ontology completeness. This paper concludes with conclusions.

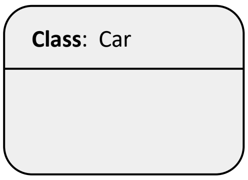
II. Basis for Ontology Completeness

Ontology is a way to display some subject area by dividing it into categories such as classes, properties, relationships and individuals. The generally accepted language for encoding ontology is OWL (Web Ontology Language) [2], and it will be used in the paper in those places where examples will be given.

Ontology completeness is one of the criteria for assessing the quality of the ontology. The field of ontology quality assessment is not new and it contains a lot of criteria for this [3]. Following the source, there are criteria and dimensions for assessing the quality of the ontology, where criteria are grouped in such a way that several criteria form one dimension [3]. Such a two-level organization of indicators should be very successful, because it obeys the advice of Stafford Beer, when he describes the anatomy of Cybersyn, namely the system for management of economic [4]. The main rule here is this: do not use technical criteria for people, because people do not operate with such categories. In this logic, dimensions are more convenient for people, while metrics are more convenient for computers [3].

Completeness dimension consists of four metrics (metrics can also be called as criteria): number of isolated elements, missing domain or range in properties, class coverage, relation coverage [3]. All these metrics reflect what is actually present or not present in the ontology. However, this is not enough. In contrast to what is said in the ontology, it is very important to evaluate what is not said in it. This is similar to how in ordinary speech there is an opportunity to understate something, thus making the desired emphasis [5]. The new metric is needed in order to eliminate the situation when formally the ontology does not contain errors and fully corresponds to the Completeness Dimension, but it is not logically complete. It is not a matter of names, but this metric can be called Logical Completeness. The Logical Completeness metric should be included in Completeness dimension. Now let us look at what this new metric consists of.

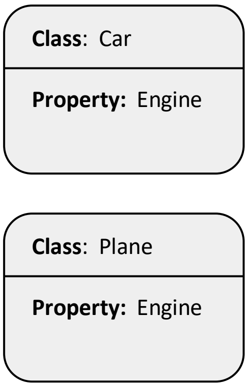
The first complaint is made against an ontology that has classes without attributes (DataProperty in the OWL syntax). If a class is declared, then something is separated from the environment, and if this something has become possible to be separated from the environment, then there are some signs (attributes, properties) by which this has become possible. For example, class “Car” is logically incomplete (Fig.1.):




```
<owl:Class rdf:ID="Car">

</owl:Class>
```

Fig.1. Logically incomplete class “Car” (scheme and code).

The second complaint is made against an ontology that has two or several classes with an identical set of properties. In this case there is no way to distinguish one class from another, because they have the same properties. For example, the classes “Car” and “Plane” are logically incomplete because they have they have the same property “Engine” (Fig.2):

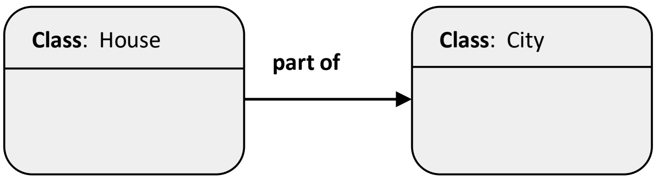



```
<owl:Class rdf:ID="Car">
  <owl:DatatypeProperty rdf:ID="Engine">
    <rdfs:domain rdf:resource="#Car "/>
    <rdfs:range rdf:resource="xs:string"/>
  </owl:DatatypeProperty>
</owl:Class>
```

```
<owl:Class rdf:ID="Plane">
  <owl:DatatypeProperty rdf:ID="Engine">
    <rdfs:domain rdf:resource="#Plane"/>
    <rdfs:range rdf:resource="xs:string"/>
  </owl:DatatypeProperty>
</owl:Class>
```

Fig.2. Logically incomplete classes “Car” and “Plane”(scheme and code).

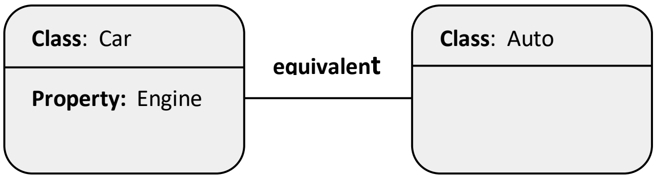
The third complaint is made against an ontology that has classes with only one subclass. If we divide something, then the result of division cannot be less than two. There cannot be only one part in the whole, there are at least two. For example, the class “City”, where the class “House” is part of the class “City” are logically incomplete, because the “City” class has the only one subclass that is the only one part (Fig.3):




```
<owl:Class rdf:ID="House">
<rdfs:subClassOf df:resource="#City"/>
</owl:Class>
```

Fig.3. Logically incomplete class “City” because of only one “part-of” relation (scheme and code).

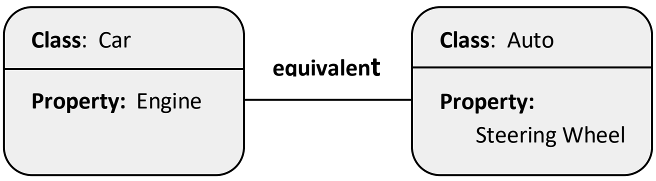
The fourth complaint is made against an ontology that has two equivalent classes, one of which has some property, but another does not have. If classes are equivalent, then properties in one class belongs to another class, too. Often, properties are only described in one class because of saving computer memory, however visually this creates the appearance of logical incompleteness. Here we need an agreement among ontology developers on how to describe such a case: either specify a property in one class and save memory, or specify the same property in both equivalent classes and thereby increase clarity and processing speed. For example, there are two equivalent “Car” and “Auto” classes and the property “Engine” is only described in the class “Car”; thus, the class “Auto” looks logically incomplete (Fig.4):




```
<owl:Class rdf:ID="Auto">
  <owl:equivalentClass>
    owl:Class rdf:ID="Car">
      <owl:DatatypeProperty rdf:ID="Engine">
      </owl:DatatypeProperty>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

Fig.4. The appearance of the “Auto” class logical incompleteness (scheme and code).

The fifth complaint is similar to the previous case, but here two equivalent classes have two different properties. In this case, it is not clear whether properties with different names in different classes are synonyms, or whether each of the equivalent classes is missing one property and this is a logical incompleteness. For example, there are two equivalent “Car” and “Auto” classes, where the class “Car” has the “Engine” property and the class “Auto” has the property “Steering wheel” (Fig. 5):

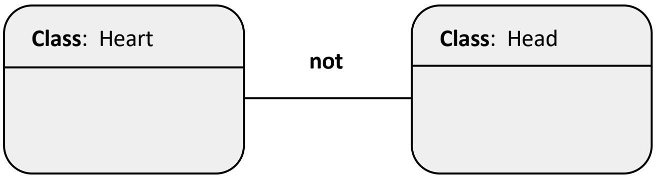



```
<owl:Class rdf:ID="Auto">
  <owl:DatatypeProperty rdf:ID="Engine">
  </owl:DatatypeProperty>
  <owl:equivalentClass>
    owl:Class rdf:ID="Car">
      <owl:DatatypeProperty rdf:ID="Engine">
      </owl:DatatypeProperty>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

Fig.5. Logically incomplete classes “Car” and “Auto”(scheme and code).

It is not clear whether the “Engine” property is equivalent to the “Steering Wheel” property, or the “Car” class should also contain the “Steering Wheel” property and the class “Auto” should also contain the property “Engine” (Fig.5). So, the classes “Car” and “Auto” are logically incomplete classes. It should also not be argued that “Engine” is not “Steering Wheel” based on names, because it is possible that an engine can be in steering wheel inside (as engine may be in the wheel in e-bike).

The sixth complaint is made against an ontology that has two classes without properties in each of them, where one class does not belong to another, i.e. there is a logical “not” relation between these classes. A logical “not” relation is a construct “complementOf” in the OWL language. The meaning is the following: if one class is not another class, then something has to differ them. Here something is some property or properties. For example, there are two classes “Heart” and “Head” with “not” relation between them (Fig. 6):

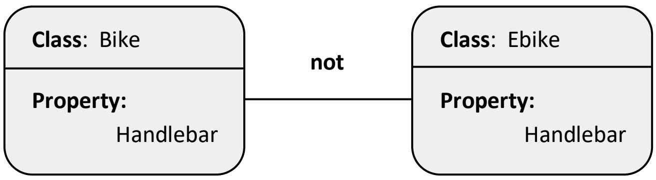



```
<owl:Class rdf:ID="Head"/>

<owl:Class rdf:ID="Heart">

    <owl:complementOf rdf:resource="#Head"/>

</owl:Class>
```

Fig.6. Logically incomplete classes “Heart” and “Head”(scheme and code).

The classes “Heart” and “Head” are logically incomplete because there is a “not” relation between these classes but it is not clear what is the difference between them considering that these classes do not have any properties (Fig.6).

The seventh complaint is made against an ontology that has two classes with the same property and there is a “not” relation between these classes. In this case, it is not clear what the difference is between the classes if they have the same structure. For example, there are “Bike” and “Ebike” classes, each of which has the same property “Handlebar” (Fig.7):

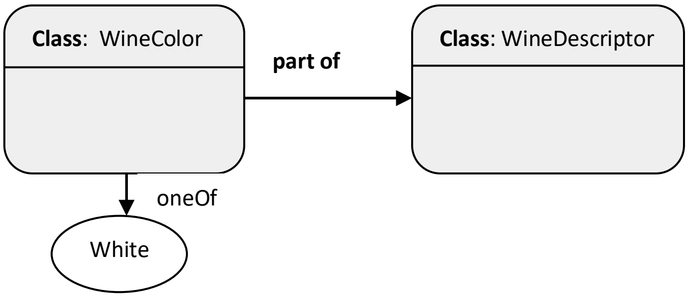



```
<owl:Class rdf:ID="Bike">
    <owl:DatatypeProperty rdf:ID="Handlebar">
    </owl:DatatypeProperty>
</owl:Class>
<owl:Class rdf:ID="Ebike">
    <owl:DatatypeProperty rdf:ID="Handlebar">
    </owl:DatatypeProperty>
    <owl:complementOf rdf:resource="#Bike"/>
</owl:Class>
```

Fig.7. Logically incomplete classes “Bike” and “Ebike” (scheme and code).

The classes “Bike” and “Ebike” are logically incomplete because these classes have the same property “Handlebar”, but there is a “not” relation between them. Thus, it is not possibility to differ these classes. What is the difference between classes if there is a “not” relationship, but a property that the classes have the same?

The eight complaint is made against an ontology that includes a construct “oneOf”, but has only one element in its composition, although “oneOf” implies several elements in its composition. Let us take a changed example from Web Ontology Language Guide [2]. Unlike the example in Web Ontology Language Guide [2], where class WineColor memebers are the individuals White, Rose, and Red, in our example there is a class WineColor whose member is the individual White [2]:




```
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
  </owl:oneOf>
</owl:Class>
```

Fig.8. Logically incomplete classes “WineColor” (scheme and code)

The class ”WineColor” is logically incomplete, because it contains “oneOf” construct with only one element “White”, but “oneOf” construct implies (logically) several elements inside (Fig. 8).

By removing all the mentioned complaints, the ontology becomes more logical and complete. It is necessary to have an indicator in order to evaluate the ontology in accordance with the mentioned cases of illogicality, or in other words of logical incompleteness.

III. Criterion calculation formula

So, in the previous section the theoretical foundations of the criterion of logical completeness for ontology were considered. The criterion of logical completeness is part of completenes dimension that is developed earlier, and it consists of eight subcriterions that are described as complaints to an ontology. It is intended to consider the criterion of logical completeness of an ontology as a certain property (or characteristic) to which the ontology can correspond in whole or in part. In accordance with this paradigm, the criterion for the logical completeness of an ontology is equal to 100% if none of the types of complaints about the ontology described earlier are found in the ontology. In turn, the criterion for the logical incompleteness of the ontology is equal to 0% if all types of complaints about the ontology described earlier are found in the ontology for each corresponding ontology construct. The ontology is partially logically complete if the value of the logical completeness criterion is in the range between 0 and 100%. This is illustrated in table I:

TABLE I. Gradation of compliance with the criterion of logical completeness.

Nr	Value of the criterion	Description
1	0 %	completely logically incomplete
2	100 %	completely logically complete
3	55 %	partially logically complete
4	82%	partially logically complete

The criterion of logical completeness is denoted as LC and it is to the sum of all subcriteria divided by eight that is (1):

$$\text{LC} = \frac{WA+IP+OS+EC1+EC2+CC+CCP+OOO}{8}, \text{ where} \quad (1)$$

WA is a subcriterion that shows in percentage how many classes without properties are in ontology (2):

$$\text{WA} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (2)$$

n is a number of all classes and m is a number of classes without properties,

IP is a subcriterion that shows in percentage how many classes with identical list of properties are in the ontology (3):

$$\text{IP} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (3)$$

n is a number of all classes with properties and m is a number of classes with identical list of properties,

OS is a subcriterion that shows in percentage how many classes with only one subclass are in the ontology (4):

$$\text{OS} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (4)$$

n is a number of all classes with subclasses and m is a number of classes with only one subclass,

EC1 is a subcriterion that shows in percentage how many equivalent classes are in the ontology, one of which has some property, but another does not have (5):

$$\text{EC1} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (5)$$

n is a number of all equivalent classes and m is a number of equivalent classes are in the ontology, one of which has some property, but another does not have,

EC2 is a subcriterion that shows in percentage how many equivalent classes with two different properties are in the ontology (6):

$$\text{EC2} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (6)$$

n is a number of all equivalent classes and m is a number of equivalent classes with two different properties,

CC is a subcriterion that shows in percentage how many classes without properties is in each of them, where one class does not belong to another, i.e. there is a logical “not” relation between these classes (7):

$$\text{CC} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (7)$$

n is a number of all classes with the relation “not” and m is a number of classes with the relation “not” without properties,

CCP is a subcriterion that shows in percentage how many classes with the same property and there is a “not” relation between these classes are in the ontology (8):

$$\text{CCP} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (8)$$

n is a number of all classes with the relation “not” and m is a number of classes with the same properties,

OOO is a subcriterion that shows in percentage how many “oneOf” constructs are in the ontology, but these constructs have only one element in its composition (9):

$$\text{OOO} = \frac{\sum_{i=1}^{n} m}{n} * 100\%, \text{ where} \quad (9)$$

n is a number of all “oneOf” constructs and m is a number of “oneOf” constructs that have only one element in its composition.

The best value of this criterion that can exist is 100%, but it is clear that the value 100% is the ideal case. Nevertheless, the higher the value of this criterion, the closer this value is to 100%, the more logically complete the ontology is.

IV. Conclusion

A criterion for evaluating the ontology completeness has been described in this paper. This criterion shows how complete the ontology is from a logical point of view. The eight typical cases, representing the ontology logical incompleteness, are listed. Based on these eight cases or complaints against the ontology, as they were mentioned in the paper, a criterion for logical completeness for ontologies was developed. The higher the value of this criterion is, the more logically complete the ontology is. The highest value of this criterion is 100%.

In general, low values of this criterion are not critical to the use of the ontology. This only indicates that the ontology is not sufficiently developed, and this insufficient development can negatively affect the operation of this ontology in specific applications, for example, when generating rules from the ontology and using them in expert systems, or in other cases. In any case, a high level of logical incompleteness of the ontology indicates the presence of a negative potential that may emerge over time, i.e. appear at the most inopportune moment. To stop the future problem, this criterion was developed. Particularly low values of this criterion force the ontology to be modified or reworked.

The place of the developed criterion in the complex of other previously developed criteria is not important. You can use this criterion separately, or within the framework of other criteria, for example, in the previously developed “completeness” dimension. The main thing is to maintain representativeness, that is, that this criterion still clearly indicates the problems of ontology from the point of view of logical completeness. This criterion would be useful to implement in tools designed for developing and evaluating ontologies, but could also be integrated into some applications that work with ontologies.

Acknowledgments

This work like most previous works has been supported by my family and my friends.

References

[1] Verhodubs O., Ontology as a source for rule generation, 2014

[2] OWL tutorial. Available online: https://www.w3.org/TR/owl-guide/

[3] Silvio Mc Gurk, Charlie Abela, and Jeremy Debattista, Towards Ontology Quality Assessment, 2017

[4] Stafford Beer, Brain of the Firm, 1972

[5] François Jullien, Putj k celi: v obhod ili naprjamik?, 2001 (in Russian)