SimpleOrAggregated

María Poveda Villalón
Ontology Engineering Group (OEG)
Departamento de Inteligencia Artificial. Facultad de Informática, Universidad Politécnica de Madrid (UPM)
Campus de Montegancedo, s/n
28660 Boadilla del Monte, Spain
+34 913363670
mpoveda@delicias.dia.fi.upm.es

Mari Carmen Suárez-Figueroa
Ontology Engineering Group (OEG)
Departamento de Inteligencia Artificial. Facultad de Informática, Universidad Politécnica de Madrid (UPM)
Campus de Montegancedo, s/n
28660 Boadilla del Monte, Spain
+34 913363672
mcsuarez@fi.upm.es

ABSTRACT
In this paper, we describe a content ontology design pattern to represent objects that can be simple or aggregated. The aggregation relation refers to several objects gathered in another object acting as a whole; all these objects should belong to the same concept in the model.

Keywords
Ontology design patterns, mereology, aggregation.

1. INTRODUCTION
Mereological relationships are one of the basic structuring primitives of the universe, and many applications require representations of them (catalogues of parts, fault diagnosis, anatomy, geography, etc.) [3].

We usually have the need of representing objects that are made up of other types of object. In these situations, we can use the part-of [1] pattern to represent transitive mereological relationships. Some examples can be "Brain and heart are parts of the human body" or "Substantia nigra is part of brain". In addition, we can use the componency [1] pattern to distinguish between parts and proper parts in a non transitive fashion. An example of this case can be "The turbine is a proper part of the engine; both are parts of a car. Furthermore, the engine and the battery are proper parts of the car".

However, sometimes we need to represent objects that can be made up of objects that belong to the same concept. In these cases it is also need to distinguish objects into simple or aggregated ones. For this reason, we have created the *SimpleOrAggregated pattern* to represent aggregation relationships, both transitive and non transitive, between objects that belong to the same concept in the model. An example of this situation can be "aggregated service provider is formed by simple or aggregated service providers".

2. PATTERN DESCRIPTION

2.1 Intent
The goal of this pattern is to represent objects that can be simple or aggregated (that is, several objects gathered in another object acting as a whole).

The main difference between the aggregation relation and other mereological relationships (such as part-of or componency) is that the aggregated object and its aggregated members should belong to the same concept.

2.2 Solution Description
As it can be observed in Figure 1 the class "ObjectByCardinality" has been created to classify simple and aggregated objects into its subclasses "SimpleObject" and "AggregatedObject", respectively. These subclasses are disjoint among them.

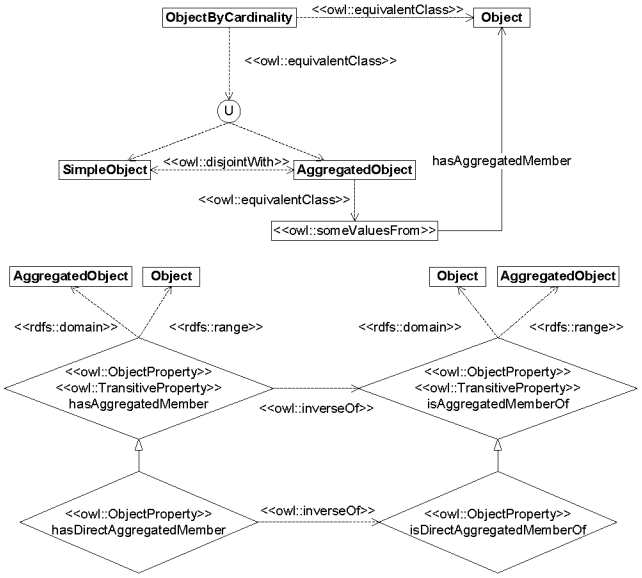

Figure 1. Graphical representation of the SimpleOrAggregated pattern.

The aggregation relationship between objects means that objects of a class can be composed by other objects of the same class. This relationship is represented by the transtive property "hasAggregatedMember" and its inverse property "isAggregatedMemberOf". These properties have as subproperties the non transitive properties "hasDirectAggregatedMember" and its inverse "isDirectAggregatedMemberOf", respectively. By means of this structure of properties, we provide a mechanism (a) to represent transitive aggregation relationships (that is, if A has B as aggregated member and B has C as aggregated member then A has C as aggregated member) and (b) to link each aggregated

member just to the next level (that is, A has B as direct aggregated member).

Finally, the class "AggregatedObject" has been defined as equivalent to those things that have some values for the property "hasAggregatedMember". This modelling allows the automatic classification of aggregated objects in this class when a reasoner is applied.

2.3 Consequences

This content pattern allows designers to represent both simple individuals of a given concept (that is, an individual that is made up of itself) and aggregated individuals of a given concept (that is, an individual that is made up of several individuals of the same concept). In summary, this pattern allows to represent both simple objects and aggregated objects and their members.

In addition, this pattern can be used to detect the following contradictory situation by means of applying a reasoner: 'to instantiate the relationship "hasAggregatedMember" for an Object that belongs to "SimpleObject"'. This situation represents a consistency error and it is detected when a resoner is applied due to the following modelling decisions included in the pattern: (a) "AggregatedObject" class represents the "hasAggregatedMember" domain and (b) "AggregatedObject" is disjoint with "SimpleObject".

3. PATTERN USAGE EXAMPLE

This pattern has been applied to different domains such as service providers and context sources during the mIO! ontology network[1] development.

As an example, we show in Figure 2 the application of the SimpleOrAggegrated pattern to represent that a service provider can be classified as simple or aggregated. Each service provider can be also classified with respect to the type of service it provides (e.g. cultural, entertainment, food, health, etc.).

4. Related work

The origin of this pattern is the modelling of service providers and context sources into the mIO! ontology network [2] within the Spanish project mIO![2]. The pattern has been also applied to computing and storage resources modelling in the Metascheduler ontology[3] in the context of the Spanish project *España Virtual*[4].

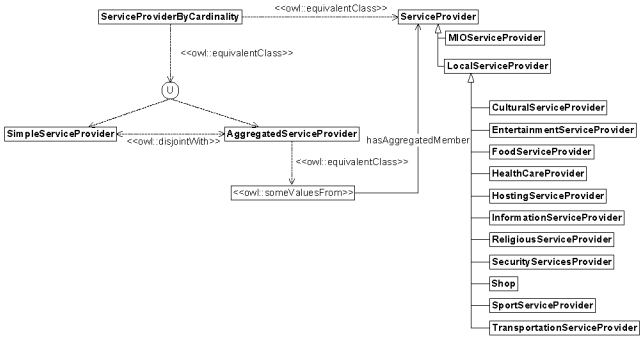

Figure 2. SimpleOrAggregated pattern applied to service providers.

5. Summary and Outlook

The *SimpleOrAggregated pattern* provides a mechanism to classify objects as simple or aggregated objects depending on whether they are an aggregation of some objects. This classification is compatible with another possible classification of objects.

6. ACKNOWLEDGMENTS

This work has been partially supported by the Spanish project *mIO!* (CENIT-2008-1019).

7. REFERENCES

[1] Presutti, V., Gangemi, A., David S., Aguado de Cea, G., Suárez-Figueroa, M.C., Montiel-Ponsoda, E., Poveda, M. *NeOn D2.5.1: A Library of Ontology Design Patterns: reusable solutions for collaborative design of networked ontologies.* NeOn project. http://www.neon-project.org. 2008.

[2] Poveda, M., Suárez-Figueroa, M.C., García-Castro, R., Gómez-Pérez, A. *A Context Ontology for Mobile Environments*. Proceedings of CIAO 2010. Lisbon, Portugal. 11 October 2010.

[3] Suárez-Figueroa, M.C., Brockmans, S., Gangemi, A., Gómez-Pérez, A., Lehmann, J., Lewen, H., Presutti, V., Sabou, M.. *NeOn D5.1.1: NeOn Modelling Components*. NeOn project. http://www.neon-project.org. March 2007.

[1] http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/ontologies/82-mio-ontologies

[2] http://www.cenitmio.es/

[3] http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/ontologies/85-metascheduler-ontologies

[4] http://www.españavirtual.org/