CM-OPL: An Ontology Pattern Language for the Configuration Management Task

Ana Carolina Almeida[1], Daniel Schwabe[2], Sérgio Lifschitz[2], Maria Luiza M. Campos[3]

[1]Dept. of Comp. Science – State University of Rio de Janeiro (UERJ)
[2]Dept. of Comp. Science – Pontifical Catholic University of Rio de Janeiro (Puc-Rio)
[3]Dept. of Comp. Science - Federal University of Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brazil

ana.almeida@ime.uerj.br, {dschwabe,sergio}@inf.puc-rio.br,
mluiza@ppgi.ufrj.br

Abstract. *Although most methodologies for ontology development emphasize reuse of existing ontologies, it is still often too complicated for people to understand the available ontologies to minimize redundancies via ontological analysis. In this context, this paper presents an Ontology Pattern Language to facilitate the construction of a well-founded ontology in the configuration management domain. As there are few studies of OPL for this domain, we present an initial version of the Configuration Management OPL (CM-OPL) and how it was used to build a configuration ontology for the vehicles domain.*

1. Introduction

Configuration Management (CM) can be applied to many diverse domains. For example, we may customize a car according to a client's requirements, which can be seen as a CM task for a given car model. Such configuration must typically follow specific rules to take into account the fact that not all options are suitable to every model.

In the computing domain, the CM task appears in many situations. For example, requirements analysts design many artifacts (e.g., UML diagram, requirements artifacts) that need to have version control since they are live documents that are updated as the clients' demands evolve over time. The CM task is presented throughout the software lifecycle too, involving people with multiple views about different products and the management of items of various types. So, it becomes useful to establish a common language to refer and deal with such variety of concepts. Although there are different products in each stage of the configuration cycle, everything converges to a final product (e.g., a vehicle or a piece of software) that needs a common understanding of all those involved.

Ontologies can be used as an inter-lingua to map concepts and services used by different tools [Guizzardi 2007]. Most existing methods for building ontologies suggest reuse as the first step (e.g., Noy et al. 2001). Although there are many ontologies available, it is not always easy to discover, understand and use ontologies developed by others for a particular domain. As a result of this difficulty in reusing ontologies, the concept of Ontology Pattern Languages (OPL) emerged. An OPL [Falbo et al. 2013a] is a network of interconnected domain-related ontology patterns that provides holistic support for solving ontology development problems for a specific domain. There are many efforts related to OPL construction [Falbo et al. 2013b][Falbo et al. 2014][Falbo et al. 2016] to expedite the process of building ontologies. In addition to facilitate reuse, an OPL also defines a reasoning order to be followed in a well-grounded systematic way.

Configuration is a critical and fundamental task, so it would be beneficial to create and make available an OPL that facilitates the development of new ontologies for any product for which configuration management information is crucial to its development, versioning or final characterization.

We call our proposed OPL CM-OPL. Its purpose is to encourage and facilitate the development of CM ontologies or subontologies modules in specific domains. The expected result of this work is that the construction of ontologies involving CM becomes more agile, precise and with fewer ambiguities. We have developed a set of patterns and created CM-OPL based on a well-founded task ontology, called CMTO, that has already been analyzed and extended for two configuration scenarios [Calhau et al. 2012]. These patterns were then used to build an ontology for the car configuration task to exemplify its application.

This paper is organized as follows. Section 2 discusses the need to define an OPL for the CM task and describes related work. Section 3 presents CM-OPL and its design. Section 4 illustrates how CM-OPL was applied to develop a car configuration ontology. Section 5 concludes this paper.

2. Unified Foundational Ontology and Ontology Pattern Language

The Unified Foundational Ontology (UFO) is an upper-level reference ontology [Guizzardi 2005]. It is strongly recommended that any ontology is built with the support of a foundation ontology to remove ambiguity and improve understanding of the defined concepts. We chose to consider a well-founded ontology as a basis to extract patterns, more specifically the already mentioned CMTO.

Ontology Patterns (OPs) are modeling solutions to solve recurrent ontology development problems [Ruy et al. 2017]. OPs can be of different types. In our CM domain, we chose the Domain-Related Ontology Pattern (DROP) type. DROPs are reusable fragments extracted from reference ontologies that should capture the core knowledge related to a domain. Thus, they can be seen as fragments of a core ontology of that domain [Falbo et al. 2013a].

An OPL complements the patterns solution providing explicit guidance to the user. It highlights the recurring problems in the domain and suggests an order to address these problems, recommending one or more patterns to solve them [Falbo et al. 2014].

OPLs are still a new topic, but some works have already been published in different application areas. The Service Ontology Pattern Language (S-OPL) provides a network of interconnected ontology modeling patterns covering the core conceptualization of services [Falbo et al. 2016], and it has been applied in a real case study to model an email service in a big Italian company. The Enterprise Ontology Pattern Language (E-OPL) [Falbo et al. 2014] organizes aspects common to several enterprises, and it has been used for building an enterprise ontology on Brazilian Governmental Universities. Also, an Ontology Pattern Language for the Software Process Domain (SP-OPL) was used for creating a domain ontology about the software process [Falbo et al. 2013b]. ISO Software Process OPL (ISP-OPL) is a specialization of SP-OPL focusing on the ISO standards devoted to software processes [Ruy et al. 2015]. Finally, the Measure OPL (M-OPL) addresses the core conceptualization of measurement [Barcellos et al. 2014].

There are presently ontologies for the CM domain, but to the best of our knowledge none of them yet used an OPL approach. Although ISP-OPL has been applied to the software CM domain, it is specific to software. Our proposal tries to generically address any CM task. CM applies technical and administrative procedures for developing, producing and supporting the lifecycle and the evolution of a product [Calhau et al. 2012]. CM helps in the control and organizational changes made to the product throughout its life cycle, preventing significant losses to the project. Calhau et

al. 2012 proposed a well-founded CM Task Ontology as a reference model supporting semantic integration in service and process layers.

3. CM-OPL: A Configuration Management Ontology Pattern Language

CM-OPL includes two parts: a set of ontology patterns and a process describing how to combine them to build new configuration ontologies to be applied to different situations. The CM-OPL patterns are represented in OPL-ML [Quirino et al. 2017], a modeling language for representing Ontology Pattern Languages. As already mentioned, we have extracted the patterns based on the CMTO, a well-founded ontology [Calhau et al., 2012]. The CM-OPL patterns are organized into three groups according to the process presented in [Calhau et al., 2012]: Configuration Identification, Version Control and Change Control.

Figure 1 presents the CM-OPL structural model. This model shows an overview of the initial version of the CM-OPL pattern groups. The CM-OPL process description, the competency questions, the diagrams of basic patterns and the CM-OPL application are presented only for the first pattern group due to space limitations. We developed a complete specification of CM-OPL[1] that can be openly accessed, including the CM-OPL process diagram.

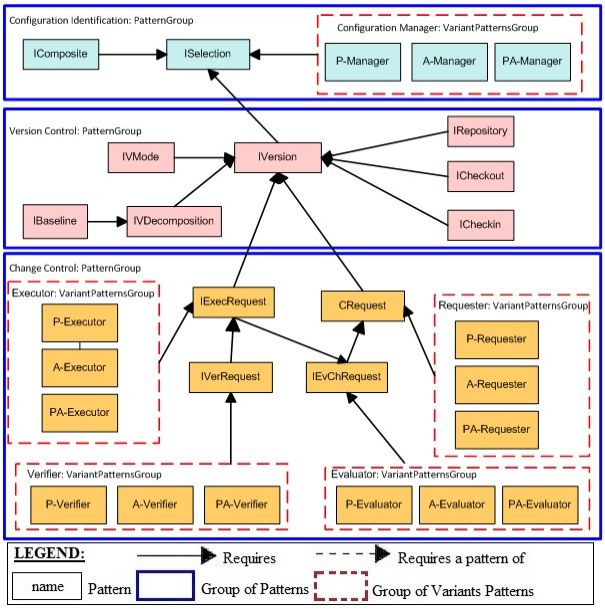

Figure 1 CM-OPL Structural Model

CM-OPL has only one entry point (EP1). The ontology engineer (OE) must start the new ontology by selecting the configuration that s/he needs to do (*ISelection*). Next, s/he decides who will manage the configuration (*Configuration Manager*). Also, it is necessary to define which item that will be configured (*IComposite*). After, the patterns of the version control and the change control groups should be used.

We have used the approach proposed by [Ruy et al. 2017] to derive the DROPs from the CMTO core ontology. The steps are: (i) modularize the core ontology according to the three main activities of CM; (ii) fragment each sub-ontology model into smaller pieces still meaningful for the domain based on competency questions; (iii) review the model fragments and select the DROPs supported by Foundational Ontology Patterns (FOPs[2]); and (iv) pack the DROP with its associated useful information.

[1] available at ftp://ftp.inf.puc-rio.br/pub/docs/techreports/18_06_almeida.pdf
[2] FOPs are reusable fragments derived from foundational ontologies [Falbo et al 2013a].

We present some Competency Questions (CQs) referring to the Configuration Identification (subontology) in Table 1.

Table 1 Competency Questions only for Configuration Identification subontology

CQ01: Which items should have their configuration managed?	*CQ03: How is a configuration item decomposed?*
CQ02: Who is the Configuration Manager that selects each configuration item?	*CQ04: Who can play the role of configuration manager?*

In the first group, the OE should address problems related to Configuration Item Definition (Figure 2). The first pattern *ISelection* defines the selection of a configuration item, that is an Item, considering the relator Configuration Selection which relates Configuration Manager and Configuration Item. This pattern answers *CQ01 and CQ02*. The stereotype of the Configuration Manager class is given by the pattern selected from the Configuration Manager sub-group. For instance, if *A-Manager* pattern is selected, then Configuration Manager is a <<role>> corresponding to Agent Configuration Manager; if the *PA-Manager* pattern is selected, then Configuration Manager is a <<rolemixin>>. The next pattern refers to the configuration item decomposition. A configuration item can be atomic or composite (*IComposite*). In this case, a composite configuration item has more than one configuration item, and it is classified as rolemixin which represents an anti-rigid and externally dependent non-sortal [Guizzardi 2005]. This pattern answers the *CQ03* and applies Category Pattern FOP – variant 1 with Mixin expression [Ruy et al 2017].

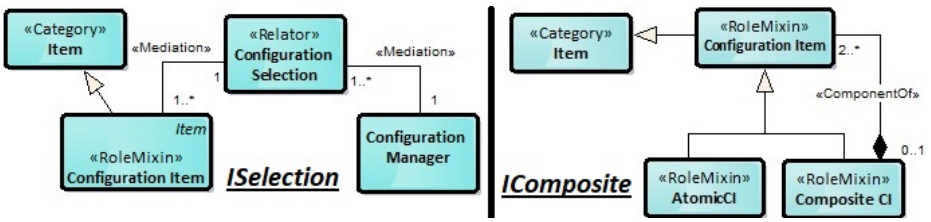

Figure 2 ISelection and IComposite Patterns

We extended the original elements of the CMTO to allow people, computational agents or both at the same time to assume all roles. Industries are increasingly automated, allowing configurations to be made by machines/agents or people. Roles can be Configuration Manager, Verifier, Requester, Evaluator, and Executor. The patterns contemplate all these roles. In Figure 3, we show an example pattern for the Configuration Manager. Thus, the Configuration Manager may be a person (pattern *P-Manager*), or it may be automated by an agent (pattern *A-Manager*), or it may be carried out by both (pattern *PA-Manager*). This last pattern applies the Rolemixin Pattern FOP – Variant 2 [Ruy et al 2017], where we define a RoleMixin as a partition of two or more Roles, each of which is connected to a Kind via a Sortal Expression. The patterns in Figure 3 correspond to the *CQ04*. Analogously, a similar set of patterns and competency questions exist for the other types of roles.

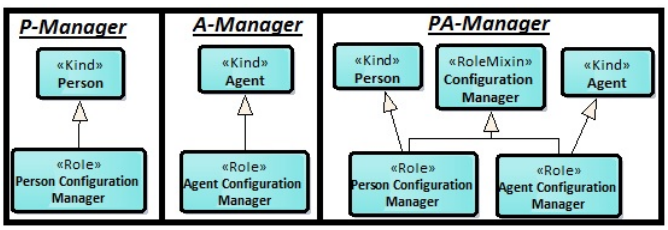

Figure 3 Typical resources structure used by CM-OPL

4. Applying CM-OPL to Vehicle Configuration Management

CM-OPL could be applied to many domains. In this work, we present a simple application example of the CM-OPL in the configuration of a vehicle.

[Ruy et al. 2017] describe two main ways of reusing ontology patterns: by analogy and by extension. We have chosen reuse by extension. In this case, the DROPs are incorporated in the vehicle configuration ontology being developed, using specialization of the original concepts and relations. In our example of reuse presented in Figure 4, we put our extensions in a grayscale.

We could implement the main concepts in the following way: imagine that when a Client wants to buy a vehicle, which has climatronic A/C unit as an optional item. We can think of the following competency questions of the specific domain for the first group (Configuration Identification):

CQ01: What configuration I need to do and who will select the configuration item?

CQ02: Who will manage the car accessory installation?

CQ03: How many parts the configuration item has?

Using CM-OPL starting from the entry point (EP1), the OE begins with the Configuration Identification group by selecting the configuration that s/he needs to do (*ISelection – CQ01*). The particular configuration is to install an A/C in the car. So, s/he needs to specialize *Item* with ***Car***. Next, s/he decides who will manage the configuration (*Configuration Manager*). Answering the *CQ02*, in a simple scenario, we determine that only people carry out all the activities. So, the OE needs to choose the *P-Manager* pattern and to specialize ***Supervisor of Agency*** (the is the person responsible for managing the A/C installation) from Person Configuration Manager. Also, as CM-OPL describes, it is necessary to define the Configuration Item decomposition using the *IComposite* pattern (*CQ03*). The OE analyzes and identifies that A/C is a ***Car Accessory*** as a Composite CI because it is composed of *Air Filter*, for example. Also, the ***Car with Configuration Managed*** is a composite CI too because the Car has many other parts and accessories, but it is not essential to enumerate them here.

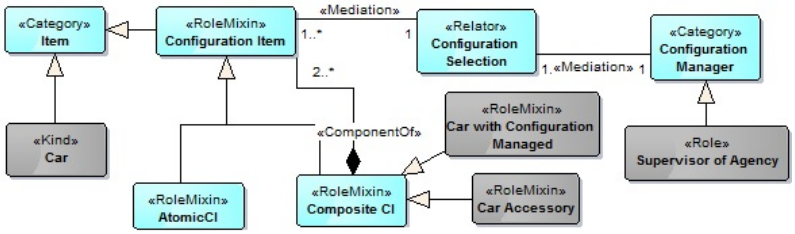

Figure 4 Fragment of Vehicle Configuration Management Ontology

5. Conclusion

OPL shows promising to facilitate the reuse of ontologies by providing a path towards (re)use of pre-defined patterns. Such ontologies may be used to improve correctness by adding models that are more precisely specified based on foundation ontologies. The CM task is commonly present, in some sort, in many computing areas as Calhau and colleagues [2012] have shown. Nevertheless, using CM-OPL, allows developing robust ontologies to characterize various CM tasks in different domains. As a proof of concept of the utility the CM-OPL, we applied it to generate part of vehicle CM ontology.

The development of CM-OPL contributes to building increasingly more complete new ontologies for the CM task. The process defined in CM-OPL guides the OE to consider a diversity of modeling situations, some of which s/he may not have anticipated. The development of new ontologies involving configuration tasks becomes

faster and more error-prone due to the reuse of already tested model fragments and the guidance provided by ordered pattern application.

As future work, we plan to explore CM-OPL in other areas. We are currently applying the same CM-OPL to derive a configuration ontology for the database tuning scenario, a much more complex setting, where the patterns and associated process will be further explored. We also plan to develop a software tool to automate the OPL process of building new ontologies that require a configuration task. Finally, we can identify new groups of patterns to contemplate, for example, activities present in the configuration planning and auditing phases.

6. References

Barcellos, M. P., Falbo, R. A., Frauches, V. G. V. (2014) "Towards a measurement ontology pattern language." In Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering, Rio de Janeiro, RJ, Brazil.

Calhau, R. F., Falbo, R. A. (2012) "A Configuration Management Task Ontology for Semantic Integration", In: Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC'12), pp. 348-353, ACM, New York, NY, USA.

Falbo, R. A., Guizzardi, G., Gangemi, A., Presutti, V. (2013a) "Ontology patterns: clarifying concepts and terminology". In: Proceedings of the 4th Workshop on Ontology and Semantic Web Patterns (Sidney, Australia).

Falbo, R. A., Barcellos, M. P., Nardi, J. C., Guizzardi, G. (2013b) "Organizing Ontology Design Patterns as Ontology Pattern Languages". In: The Semantic Web: Semantics and Big Data, vol. 7882, Lecture Notes in Computer Science, pp. 61-75.

Falbo, R. A., Ruy, F. B., Guizzardi, G., Barcellos, M. P., Almeida, J. P. A. (2014) "Towards an Enterprise Ontology Pattern Language". In: 29th ACM Symposium On Applied Computing (ACM SAC 2014), Gyeongju, Korea.

Falbo, R., Quirino, G. K., Barcellos, M. P., Guizzardi, G. (2016) "An Ontology Pattern Language for Service Modeling". In: 31st ACM Symposium on Applied Computing (ACM SAC 2016), Pisa, Italy.

Guizzardi, G. (2005) "Ontological Foundations for Structural Conceptual Models". In: Universal Press, The Netherlands.

Guizzardi, G. (2007) "On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models". In: Proceedings of the 2007 conference on Databases and Information Systems, pp. 18-39, Amsterdam, The Netherlands.

Noy, N. F., Mcguinness, D. L. (2001) "Ontology Development 101: A Guide to Creating Your First Ontology". In: Knowledge Systems Laboratory, pp: 1-5, Stanford University.

Quirino, G. K. S., Barcellos, M. P., Falbo, R. (2017) "OPL-ML: A Modeling Language for Representing Ontology Pattern Languages". In: Lecture Notes in Computer Science, November 2017, pp. 187-201.

Ruy, F. B., Falbo, R. A., Barcellos, M.P., Guizzardi, G., Quirino, G.K.S. (2015) "An ISO-based Software Process Ontology Pattern Language and its Application for Harmonizing Standards". ACM SIGAPP Applied Computing Review, 15(2):27--40.

Ruy, F. B., Guizzardi, G., Falbo, R. A., Reginato, C. C., Santos, V. A. (2017) "From reference ontologies to ontology patterns and back". In: Journal Data & Knowledge Engineering, v. 109, issue C, pp. 41-69, Elsevier Science Publishers, The Netherlands.