Extending an Ontology Editor for Domain-related Ontology Patterns Reuse: An Application in the Collaboration Domain

André K. Souza[1], Renata Guizzardi[1], Maria Luíza M. Campos[2], Giancarlo Guizzardi[1]

[1]Ontology and Conceptual Modeling Research Group (NEMO)
Department of Computer Science
Federal University of Espírito Santo – Vitória, ES – Brazil
[2]Computer Science Department, Federal University of Rio de Janeiro – Rio de Janeiro, RJ – Brazil

[1]aksovix@gmail.com, [1]rguizzardi@inf.ufes.br,
[2]mluiza@ppgi.ufrj.br, [1]gguizzardi@inf.ufes.br

Abstract. *Developing ontologies from scratch is a hard task, since an ontology is expected to provide a comprehensive and coherent representation of a specific portion of the world. Thus, reuse is highly recommended in ontology development, allowing ontologies to be build based on pre-existing models, leading to better quality results. In this sense, Ontology Patterns (OP) have been appointed as interesting tools to facilitate reuse, and several authors from the Ontology Engineering community have already proposed OPs and mechanisms to apply them. However, automated systems to support their use in practice are still missing. In this paper, we present an ontology editor extension that fills this gap, supporting Domain-related OPs reuse and management. This extension includes a catalogue of Domain related OPs, exemplified in this paper through a set of OPs for the Collaboration domain.*

Resumo*.O desenvolvimento de ontologias a partir do zero é uma tarefa difícil, uma vez que uma ontologia deve fornecer uma representação global e coerente de uma parte específica do mundo. Assim, a reutilização é altamente recomendada em seu desenvolvimento, permitindo que as ontologias sejam construídas com base em modelos pré-existentes, levando a melhores resultados quanto a sua qualidade. Neste sentido, Padrões de Ontologia (PO) são considerados como ferramentas interessantes para facilitar a reutilização, sendo que vários autores da comunidade de engenharia de ontologias já propuseram PO e mecanismos para aplicá-las. No entanto, sistemas automatizados para apoiar a sua utilização na prática ainda são raros. Neste artigo, apresentamos uma extensão de um editor de ontologias que preenche esta lacuna, apoiando o reuso e gerenciamento de POs específicos de domínio. Esta extensão inclui um catálogo de POs específicos de domínio, exemplificada neste trabalho através de um conjunto de POs para o domínio de Colaboração.*

1. Introduction

Ontology engineering has evolved in the last decades with an increasing number of methodologies, tools and applications being proposed and experimented in academia and in industry. However, ontologies' development from scratch is still a hard and complex task, since an ontology is expected to provide a comprehensive and coherent representation of a specific portion of the world. Similar to successful Software Engineering approaches and good practices, reuse has been largely advocated for ontologies to be built based on pre-existing models, speeding up the development process and leading to better

quality results. In this sense, Ontology Patterns (OP) have been appointed as interesting tools to enable reuse.

An Ontology Pattern (OP) describes a particular recurring modeling problem that arises in specific ontology development contexts and presents a well-proven solution for the problem. Different kinds of OPs support ontology engineers on distinct phases of the ontology development process [Falbo et al. 2013]. For instance, a Domain-related Ontology Pattern (DROP) is a kind of Content Ontology Design Pattern that captures a reusable fragment extracted from a reference domain ontology, which may assist in building the conceptual model of a new domain ontology. A Logical Ontology Design Pattern, or an Ontology Coding Pattern, in turn, supports ontology implementation, aiming at solving problems related to reasoning or concerning shortcomings in the expressivity of a specific logical formalism. In this paper, we focus on DROPs, as our work is related to the conceptual modeling phase of ontology engineering.

Although OPs are useful, applying them in practice may take a lot of effort since, in general, OP catalogues are composed of several interconnected OPs. In this paper, we present the extension of an ontology editor to automate the use of OPs, describing its use together with a catalogue, specially built to manage a set of Collaboration OPs. The extended editor is named OLED. By extending it, we aim at taking advantage of the support this pre-existing editor has for ontology validation and ontology code generation.

In this paper, we discuss the use of OPs considering the Collaboration context, as they as applicable to a broad range of application areas, such as Groupware, software development, emergency management systems, among other domains. Thus, before presenting the developed editor extension, we present the Collaboration Ontology Patterns that compose the automated catalogue. Additionally, we briefly discuss how these patterns have been extracted from an existing reference ontology, and how they may be applied for ontology development.

The remaining of this paper is organized as follows: Section 2 discusses about Ontology Patterns and present the reference ontology used for the Collaboration OPs extraction; Section 3 describes the Collaboration Ontology Patterns used in the development of the catalogue presented in this paper; Section 4 describes our baseline, i.e. the OLED editor; Section 5 describes the editor's extension, showing how to use an OP catalogue in OLED; Section 6 discusses the viability of the developed editor, by presenting a running example; Section 7 presents some related works; and Section 8 concludes the paper.

2. Theoretic Background

In this section, we put forward some words about Ontology Patterns (section 2.1) and then, we describe the CONTO Ontology (section 2.2), which was the basis for the extraction of the Collaboration Ontology Patterns presented in this paper.

2.1. Ontology Patterns

According to [Buschmann et al. 2007], "an Ontology Pattern (OP) describes a particular recurring modeling problem that arises in specific ontology development contexts and presents a well-proven solution for the problem. The solution is specified by describing the roles of its constituent participants, their responsibilities and relationships, and the ways in which they collaborate". OPs are reusable fragments extracted from ontologies and, "in general, are vehicles for encapsulating knowledge. They are considered one of the most effective means for naming, organizing, and reasoning about design knowledge" [Falbo et al. 2011].

As an illustration, suppose a case of assembling a vehicle. There are several recurring problems to be solved to create the vehicle: how to make the vehicle move, how to make it stop when necessary, how to enable it to illuminate the road in which it is riding, etc. Every time a vehicle is assembled, these problems must be solved. Consequently,

standards have been created to solve them, resulting in the development of different components (e.g. motor, steering wheels, breaks, flashlights) that when assembled together, allow the vehicle to work properly. Analogously, the development of ontologies can be seen as a component-based construction task, where the components are in fact OPs.

By reusing OPs, the ontology engineer can select the patterns that meet his/her needs, extending them for the targeted ontological domain. Experiments have shown that using OPs on ontology development has a direct impact on the quality of the resulting ontology, making them more consistent than the ones built from scratch [Blomqvist, Gangemi, and Presutti 2009].

2.2. CONTO - Collaboration Ontology

The Collaboration Ontology, CONTO follows the 3C Collaboration Model thus being subdivided in three ontologies: Cooperation, Communication and Coordination. Figure 1 shows an UML package diagram representing the three sub-ontologies and the relationship between them.

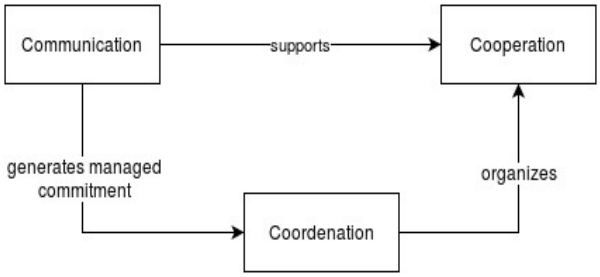

Figure 1. The three sub-ontologies that compose the CONTO ontology

As depicted in Figure 1, communication supports cooperation, since the latter is achieved only by the interaction of the collaborating agents. In addition, the agents' communication create commitments that are managed by coordination mechanisms. Finally, one can say that coordination organizes the cooperation processes [Oliveira 2009].

Table 1 presents CONTO's concept dictionary, defining the concepts modeled in each of the sub-ontologies.

3. Collaboration Ontology Patterns

The Collaboration Ontology Patterns have been directly extracted from CONTO. Each of them consists of a set of concepts that together, models a solution to solve a recurrent ontology design problem. In total, we extracted thirty patterns from CONTO, organizing them in a specification document. Figure 2 illustrates how patterns may be detected and extracted from ontologies, by showing two patterns extracted from an excerpt of the Coordination Ontology.

The patterns shown in Figure 2 are: i) the Collaborative Session Resource (CSR) pattern, whose purpose is to identify what resources participate in the Collaborative Session and ii) The Resource Nature (RN) pattern that allows one to query the ontology regarding the shareable and exclusive resources participating in the Collaborative Session. Tables 2 and 3 respectively present part of the specification of these two patterns. We say that the specification is partial because in the specification document, besides the information presented in these tables, for each pattern there is a diagram modeled using OntoUML and an axiomatization formalizing the pattern. Here, we refrain from presenting all axioms and models due to space reasons.

Table 1. CONTO Concept Dictionary

COOPERATION	Agent	Physical Objects that are the collaboration actors.
	Collaborative Role	Social Role that agents play during collaboration
	Collaborative Session	It is an Interaction, i.e. a Complex Action in which at least two Agents are involved for the purpose of collaborating.
	Site	Location where the collaborative session happens. It may be Virtual or Real.
	Collaborative Agreement	Social relator that defines a Collaborative Session.
	Action Contribution	Action which constitute intentional participations performed by the Agents involved in the Collaborative Session.
	Close Commitment	Social Moment established by a Collaborative Role that commits to achieve the Goals of the Collaborative Session, by performing specific Action Universals.
COMUNICATION	Material Contribution	An Action Contribution that physically change the world.
	Communication Act	An Action Contribution that carries out the information that should be exchanged in a session.
	Perception	An Action Contribution that consists in the reception of the information communicated through the Communicative Acts.
	Message	The propositional content of a Communicative Act.
	Idiomatic Language	Language using an idiom for its representation.
	Sender	Agent who sends the message.
	Receiver	Agent who perceives the message.
	Communicative Interaction	Complex Action composed of exactly one Communicative Act and one or more Perceptions.
COORDINATION	Protocol	Normative Description defining the rules that govern the Collaboration Group and, thus, being recognized by this group. The Protocol defines the Collaborative Roles (e.g. leader, participant, etc.) and the Action Contribution Universals presente in the Collaborative Session.
	Collaborative Group	Social Agent composed of the Physical Agents that participate in the Collaborative Session.
	Exclusive Resource	Resource that cannot be used simultaneously.
	Sharable Resource	Resource that can be used simultaneously.

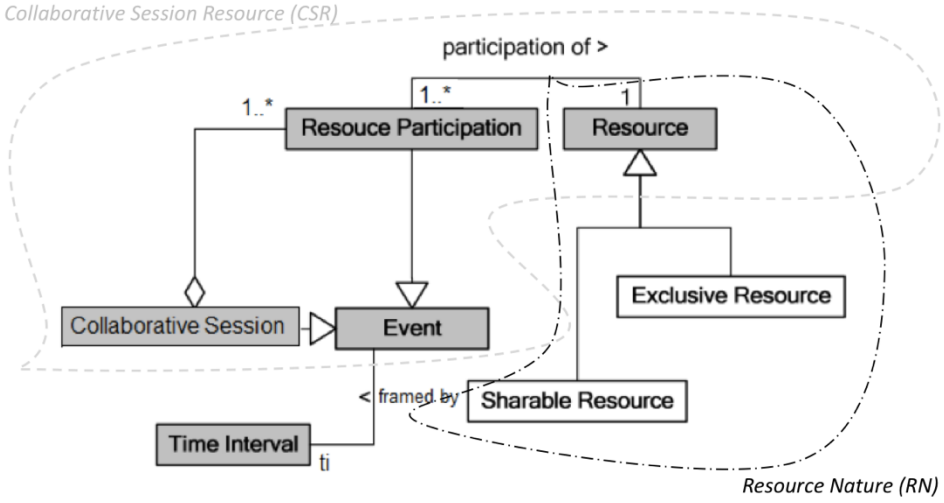

Figure 2. Pattern extraction from a Coordination Ontology excerpt

Table 2. Collaborative Session Resource (CSR) Pattern Partial Specification

Characteristic	***Description***
Name/Achronym	Collaborative Session Resource / CSR
Intent	To represent the participation of resources in a collaborative session, including the mediaton of a resource participation relator.
Rationale	A collaborative session has one or more resource participation, which refers to one resource.
Competency Questions	What are the resources that participate on the collaborative session?

Table 3. Resource Nature (RN) Pattern Partial Specification

Characteristic	***Description***
Name/Achronym	Resource Nature/RN
Intent	To represent resource types in terms of shareability
Rationale	A resource can be either sharable or exclusive
Competency Questions	- What is the nature (in terms of shareability) of a given resource? - Which resources are shareable? - Which resources are exclusive?
Axioms:	A1 – Exclusive Resource (r) → ∀ p, p' participationOf(p,r) ∧ participationOf (p',r) ∧ (p ≠ p') → ¬overlaps (p, p')
	A2 – Sharable Resource (r) → ¬ Exclusive Resource (r)

Selecting a pattern is not so straightforward because, in the ontology, all concepts are connected through relations (see figure 2). And when extracting a pattern, we must be aware of the right granularity, i.e. how many concepts should be part of the pattern. If the pattern is too large, then there may be cases in which some of the concepts will be ignored when the pattern is used on ontology creation. Splitting patterns and ignoring concepts is undesirable, otherwise reuse may be compromised and the resulting ontology may become inconsistent. Thus, during extraction, we were concerned in defining one pattern for each design problem, documenting the pattern's *intent* in the specification document (see tables 2 and 3).

Through pattern-based ontology development, the ontology engineer may create an ontology by extending each of the patterns available in the catalogue [Falbo et al. 2011]. Thus, instead of starting from scratch, the ontology engineer may reuse existing models, which have gone through consistency and correctness verification, also being formally defined. This results in an increase in the ontology quality, when compared with the ones made from scratch. Moreover, if the patterns are extracted from a well-founded ontology, like the ones we present here, the quality of the ontologies that are build over them are even greater.

Figure 3 illustrates how a pattern can be extended for the domain of collaborative editing. The gray elements are from the RN pattern itself, while the white ones are from the modeled domain. In this domain, one important ontological distinction to be made is that of readable versus writable documents. A readable document is a kind of sharable resource, since it may be accessed simultaneously by different agents at the same time, thus participating in different simultaneous participations. Adopting what in collaborative editing is called pessimist lock, the writable document is a kind of exclusive resource, because it cannot be simultaneously altered. Please refer to the two axioms defining Exclusive and Sharable Resource presented on table 3. The participation_of and overlaps predicates in the A1 axiom have the following meaning here: (i) participationOf captures the relation between an object (endurant) and the portion of an event that exclusively depends on that object; (ii) overlaps holds between two events iff their temporal extents share a non-instantaneous temporal interval.

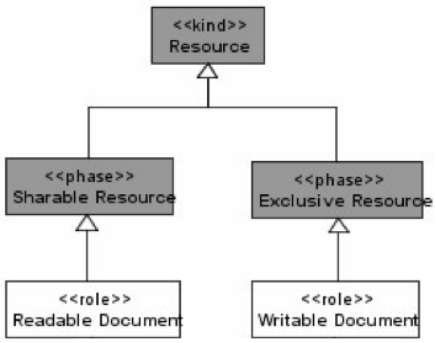

Figure 3. A case extending the RN Pattern

Figure 3 exemplifies one pattern extension only. In order to create the whole ontology on collaborative editing, the ontology engineer must select from the catalogue, the other patterns that are useful for this particular domain, connecting them and extending them one by one, until the whole ontology is created.

4. Baseline: The OntoUML lightweight editor (OLED)

The OntoUML lightweight editor (OLED) is a model-based environment to support Ontology Engineering in OntoUML, in particular the task of formalization, verification, validation and implementation. OLED was designed to aggregate all the aforementioned technologies developed for OntoUML in the long-term research project conducted by the Ontology and Conceptual Modeling Research Group (NEMO) [Guerson et al. 2015]. Figure 4 shows OLED main window.

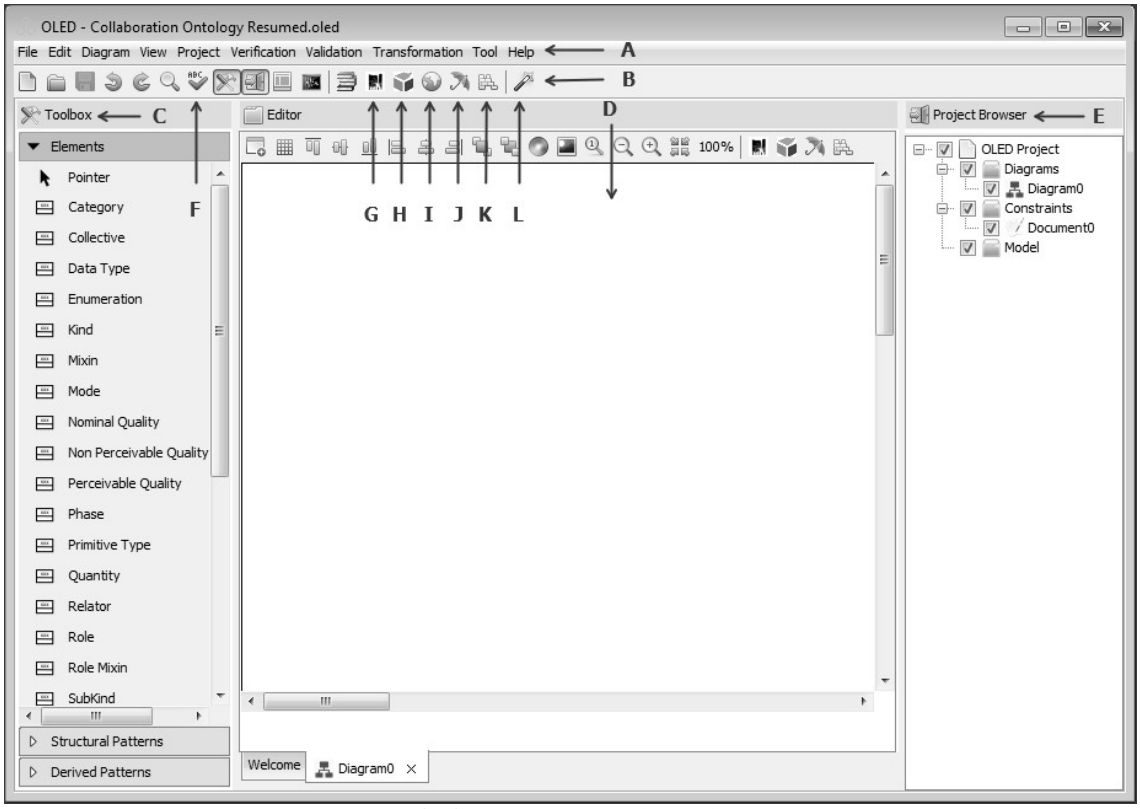

Figure 4. OLED clear project window.

On Figure 4, at the top of OLED interface, indicated by label *A*, there are the main menu options like File, Edit, Diagram, View, Project, etc., along with their submenus. Underneath the main menu, there is the toolbar with shortcuts for the most used functions, indicated by label *B*. Vertically, in the left side, we can see: (i) the toolbox, with OntoUML elements; (ii) the library of modeling patterns; (iii) the library of derived modeling

patterns (indicated by label *C*. Labeled as *D*, in the center, there is an empty area, where an OntoUML model may be created and edited. Labeled as *E*, we can see the Project Browser, that shows the project structure, disposed in a tree view, with its subdivisions: Diagrams, Constraints and Model.

On the toolbar labeled as *B*, there are some shortcut buttons, linking to the other functionalities available in OLED (labelled accordingly): *F* - semantic verification; *G* - model simulation using an Alloy Analyzer, that allows the ontology engineer to examine instances of the edited model to verify the consistency of his ontology; *H* - ontology code generation using OWL/SWRL; *I* - transformation of the ontology into a document in the SBVR verbalization language; *J* - ontology Anti-Patterns verification, which allows the ontology engineer to correct modeling mistakes in the ontology; *K* - ontology evaluation, regarding the transitivity of meronymic and part-whole relations; and, finally, *L* - ontology evaluation regarding completeness.

5. OLED Extension with a Catalogue of Domain-related Ontology Patterns

In order to support the management and use of DROPs, we extended the OLED editor with a new module that includes a catalogue and functionalities for OPs reuse and edition. Due to space limitations, we have not included in this paper a detailed description of the catalogue creation. In the current solution, the DROPs catalogue was conceived and implemented as an OLED project in which each potential pattern corresponds to a diagram in this project. Potential because in this case, the DROP is not supposed to be interpreted as a genuine pattern in OLED, skipping some of the usual features for this type of construct in the tool.

A DROPs catalogue can be loaded and become available on the toolbox located on the left side of the OLED interface. In Figure 5, we can see the Collaboration Ontology Patterns Catalogue, with available OPs listed on the left side (marked with label *A*) in the figure). Placing the mouse pointer over an OP, we can see a balloon with some pattern information (indicated by the label *C*), in the same figure), such as name, intent, rationale, among others. To use an OP, we must select the corresponding one (label *B*), by clicking on it. In this particular case, we choose the RN OP from the Collaboration Patterns catalogue. The pattern window will pop-up, so that the pattern can be configured for reuse (as will be described latter in this section). Once configured and executed, the OP configuration window will close and the OP will be included in the current open diagram and in the Project Browser, indicated by the labels *D* and *E* respectively.

Figure 6 shows the OP configuration window for the RN ontology pattern. On the left part of the window, there is a list of the classes included in the RN OP. Any of the classes present in the OP can be selected for reuse, by checking the corresponding checkbox in the "Pick up from model?" column indicated by label *B*. Then, a new window will be open, showing a list of the classes available in the diagram, to be chosen and used. Once we have made the selection, it will return to the current window. Label *C* points to the button associated to the command for performing the OP creation, according to the specified settings (reusing or not existing categories). This window also shows a preview of the OP being configured, as indicated by label *D*.

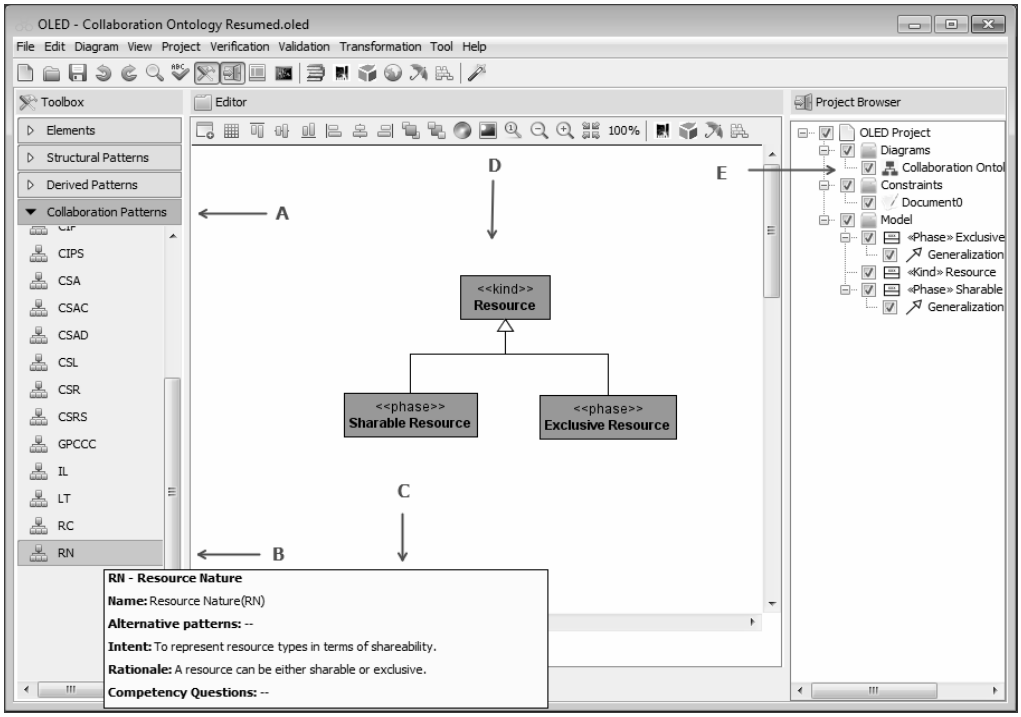

Figure 5. Selecting the RN OP from the OLED Collaboration Ontology Patterns Catalogue

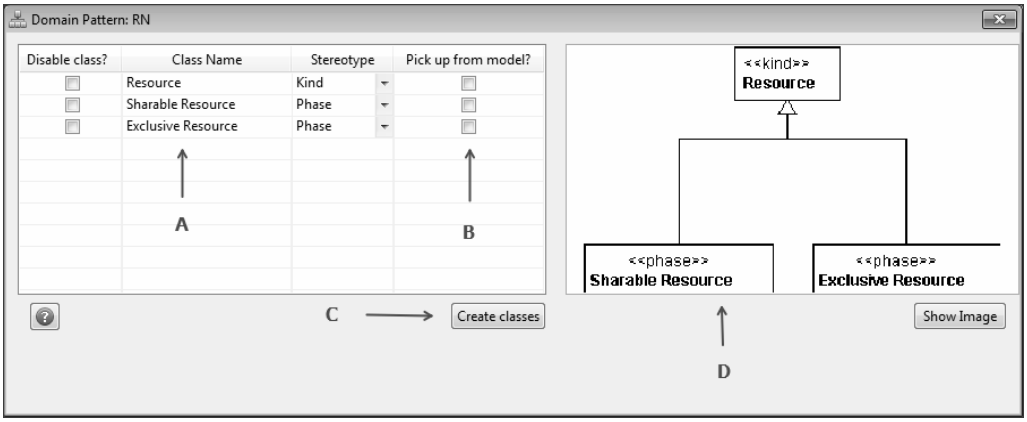

Figure 6. Configuring Ontology Pattern window for the RN OP

After the OP has been added to the diagram, it may be extended, as shown in Figure 7. In the left part of this figure, indicated by label *A*, we can see the available OntoUML categories under the "Elements" item in the toolbox. Two elements of the Role category have been selected (as indicated by label *B*) to define two new concepts in the diagram. To select the elements in the toolbox, we simply click with the left mouse button on the element and then click on the diagram. We have to do the same thing with the generalization relationships we want to add to extend the RN OP. The Roles and the Generalization relations added to the diagram are respectively indicated by labels *D* and *E*.

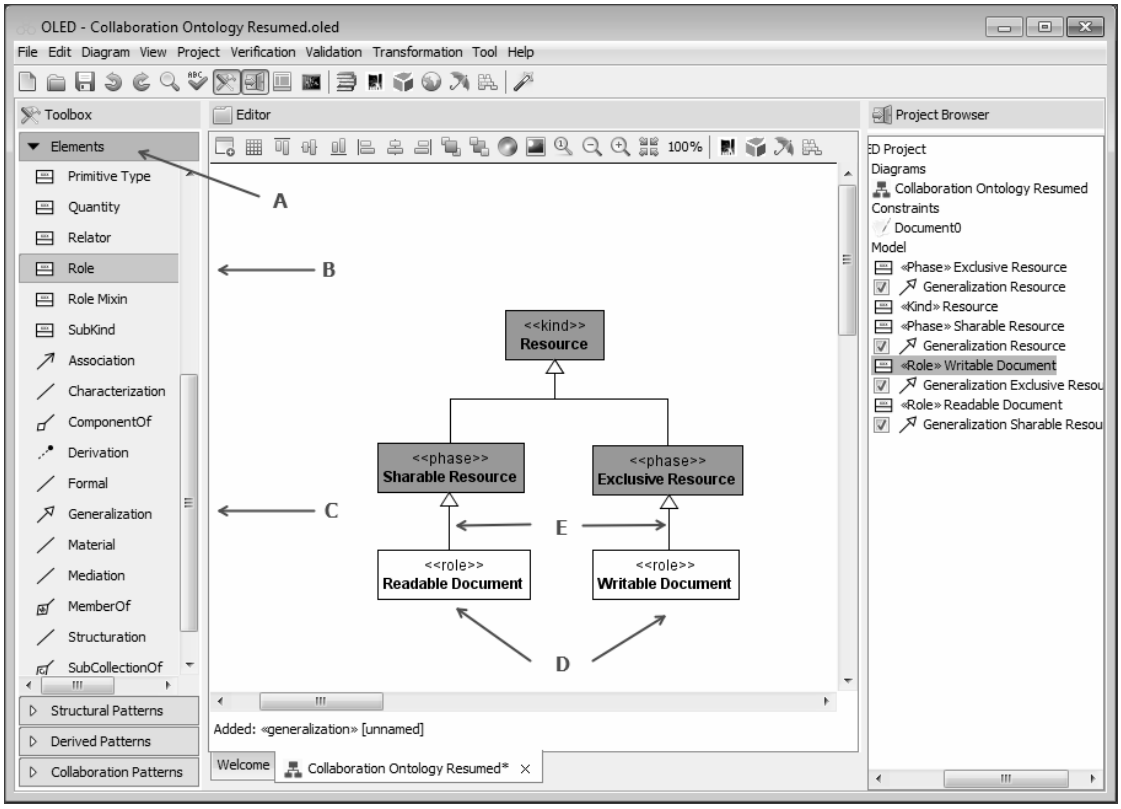

Figure 7. OLED project diagram with RN OP extended.

6. Running Example

To prove the feasibility of the use of the implemented OP catalogue, we here present a case on the collaboration editing domain. The idea is to develop an ontology in this domain on top of the developed Collaboration Ontology Patterns. The developed ontology can then be used as the analysis model for the development of an online collaborative editor. Collaborative editing allows multiple users to simultaneously visualize, edit and share texts. In this scenario, multiple users interact with each other in a virtual environment collaboratively producing textual content.

In Ontology Engineering, the ontology's requirements are usually expressed by a set of competency questions, which are supposed to be answered by the ontology to be. Here, competency questions are also applied to help guide the user to chose the right OPs to be used in the development of the ontology. Table 4 presents the CQs for the collaborative editing scenario, also showing to which pattern each of the CQ maps.

Table 4. Competency Questions and Patterns Mapping.

Question	Description	Pattern
CQ01	Where does the collaborative editing session happen?	CSL
CQ02	What kinds of collaborative sessions are there?	CSS
CQ03	What kinds of actions are there in a collaborative editing session?	CSA
CQ04	What kinds of commitments are there in the collaborative editing session?	CCAC
CQ05	What are the versions of a document?	ACPPS
CQ06	What changes in document comparing two versions of it?	ACPPS
CQ07	A particular document is available for writing?	RN
CQ08	What documents are sharable?	RN
CQ09	What documents are exclusive?	RN

After mapping the competency questions and the Collaboration OPs, we selected them from the OLED's catalogue, and then extended them, by including the concepts of the collaborative editing domain. The resulting ontology can be seen in Figure 8.

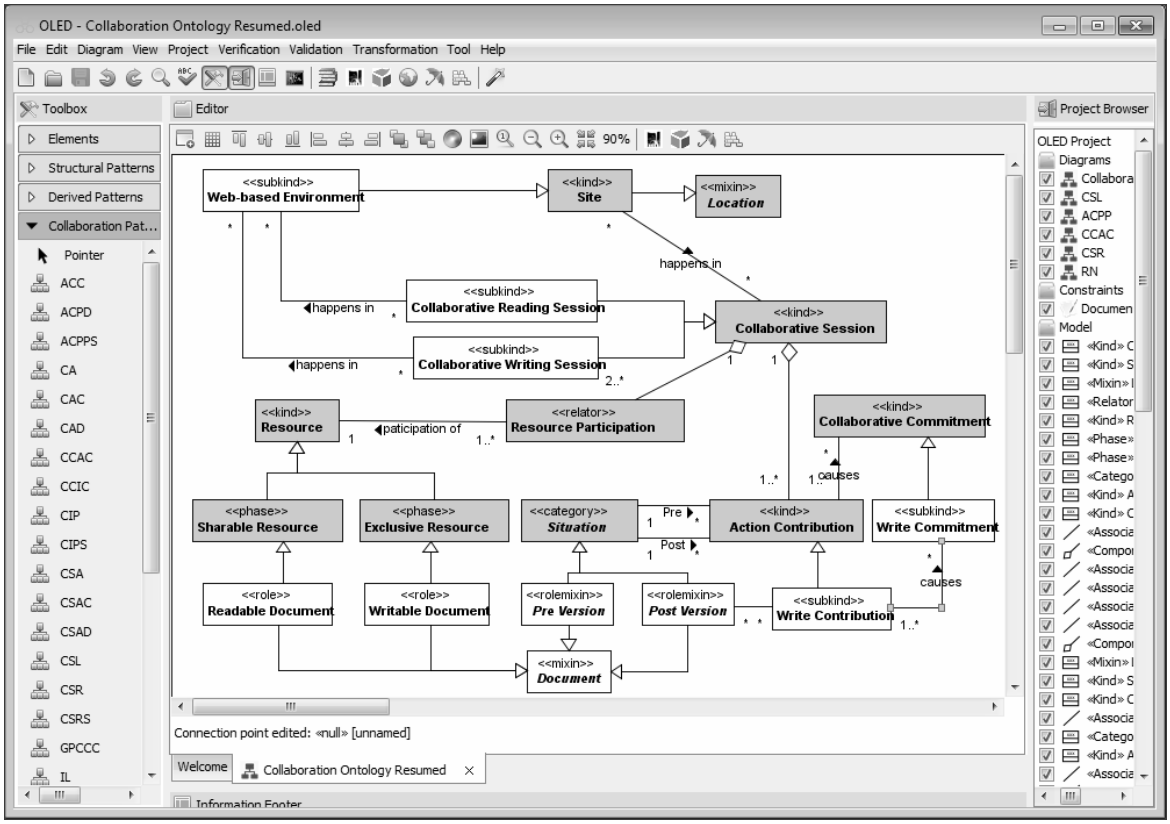

Figure 8. A Simple Ontology based on OPs from Collaboration Patterns.

7. Related Works

Other efforts have been made to provide automatic support to build OP catalogues. OntoCase is one of them. OntoCase is a general framework for building semiautomatic ontologies interactively, based the notion of OP. The framework is based on a case-based reasoning (CRB) methodology that proposes cycles of applying learned information acquired from the past to solve present problems [Blomqvist 2009].

Another example is SEAM (SEmi-Automated ontology Maintenance) that is an ontology development system that leverages practices in information and relationship extraction from text to streamline the process of generating knowledge structures, in this case, specifically used for medical specialty ontologies. These ontologies are used for machine assisted clinical diagnostic decision support (CDS). The goal of the SEAM system is to facilitate the necessary information acquisition to build ontologies that represent settled science and common term usage with respect to either medical specialty or particular disease [Doing-Harris et al. 2015].

Finally, we may also cite the Extreme Design (XD) plug-in for the NeOn toolkit. The NeOn Toolkit is the ontology engineering environment which provides comprehensive support for the ontology engineering life-cycle and XD is a plugin to select relevant ontology design patterns and specialize them for the specific requirements and domain of the target ontology, also supporting the integration of these patterns into the ontology [d'Aquin et al. 2009]

With respect to our work, the aforementioned systems deal with Ontology Coding Patterns, rather than DROPs like the ones our catalogue supports. We are currently not aware of any other system supporting DROPs.

8. Conclusion

In this paper, we showed how to use an automated Ontology Pattern Catalogue, developed by extending an existing ontology editor. We exemplified the use of the extended editor with a catalogue of patterns from the Collaboration domain. However, this same procedure may be used for patterns defined in any domain. To exemplify the use of the developed tool, we used it to create an ontology for the collaborative editing domain, by extending the Collaboration OPs present in the catalogue.

For the future, we hope to keep improving the tool to provide more support for DROPs management and reuse. For instance, the next step is to add the competency questions specified for each pattern in the editor at the moment of the catalogue creation, to assist in the patterns choice, thus providing further guidance to the ontology engineer. Another interesting feature to be developed is to enable the configuration of alternative patterns of a given pattern, depending on the modeling needs. Another direction for future research is applying the automated catalogue for different case studies and developing empirical studies to validate its use for ontology development.

References

Blomqvist, E. (2009) Semi-automatic ontology construction based on patterns. Linköping Studies in Science and Technology. Dissertations, ISSN 0345-7524. Available at http://liu.diva-portal.org/smash/get/diva2:207543/FULLTEXT01.pdf. Last access: July 8th 2016.

Blomqvist, E., Gangemi, A., Presutti, V. (2009) Experiments on Pattern-based Ontology Design. In Proceedings of K-CAP 2009, pp. 41-48.

Buschmann, F., Henney, K., and Schmidt, D. C. (2007) Pattern-oriented software architecture: On patterns and pattern languages. John Wiley Sons Ltd. page 8.

Doing-Harris, K., Livnat, Y., and Meystre, S. (2015) Automated concept and relationship extraction for the semi-automated ontology management system (SEAM). Journal of Biomedical Semantics, Vol. 6, N. 15.

d'Aquin, M., Gangemi, A., Motta, E., Dzbor, M., Haase, P., and Erdmann, M. (2009) Neon tool support for building ontologies by reuse. In Demo International Conference on Biomedical Ontologies.

Falbo, R. A., Barcellos, M. P., Nardi, J. C., and Guizzardi, G. (2013) Organizing ontology design patterns as ontology pattern languages. Springer. pp 61–75.

Falbo, R. A., Guizzardi, G., Gangemi, A., and Presutti, V. (2011) Ontology patterns: Clarifying concepts and terminology. Springer. page 2.

Guerson, J., Sales, T. P., Guizzardi, G., and Almeida, J. P. A. (2015) Ontouml lightweight editor: A model-based environment to build, evaluate and implement reference ontologies. In EDOC Workshop. pp. 144-147.

NEMO, O. . C. M. R. G. (2016). Oled - ontouml lightweight editor. Available at: http://nemo.inf.ufes.br/projects/oled/.

Oliveira, F. F. (2009). Uma ontologia de colaboração e suas aplicações. (in Portuguese) Masters Dissertation, Universidade Federal do Espírito Santo. Available at: http://portais4.ufes.br/posgrad/teses/tese_3335_.pdf

OMG, O. M. G. (2016). OMG Meta Object Facility. Available at: http://www.omg.org/mof/.

Presutti, V., Daga, E., Gangemi, A., and Blomqvist, E. (2009) Extreme Design with Content Ontology Design Patterns. In Proceedings of the Workshop on Ontology Patterns. pp. 83-97.