REVERSE ENGINEERING A DOMAIN ONTOLOGY TO UNCOVER FUNDAMENTAL ONTOLOGICAL DISTINCTIONS

An Industrial Case Study in the Domain of Oil and Gas Production and Exploration

Mauro Lopes[1], Giancarlo Guizzardi[2], Fernanda Araujo Baião[1] and Ricardo Falbo[2]
[1]*NP2Tec – Research and Practice Group in Information Technology,*
Federal University of the State of Rio de Janeiro (UNIRIO), Rio de Janeiro, Brazil
[2]*Ontology and Conceptual Modeling Research Group (NEMO), Computer Science Department*
Federal University of Espírito Santo (UFES), Vitória, Brazil

Keywords: Ontology, Ontology Languages, Conceptual modelling, Oil and Gas domain.

Abstract: Ontologies are commonly used in computer science either as a reference model to support semantic interoperability in several scenarios, or as a computer-tractable artifact that should be efficiently represented to be processed. This duality poses a tradeoff between expressivity and computational tractability that should be taken care of in different phases of an ontology engineering process. In this scenario, the choice of the ontology representation language is crucial, since different languages contain different expressivity and ontological commitments, reflecting on the specific set of available constructs. The inadequate use of a representation language, disregarding the goal of each ontology engineering phase, can lead to serious problems to database design and integration, to domain and systems requirements analysis within the software development processes, to knowledge representation and automated reasoning, and so on. This article presents an illustration of these issues by using a fragment of an industrial case study in the domain of Oil and Gas Exploration and Production. We make explicit the differences between two different representations of this domain, and highlight a number of concepts and ideas (tacit domain knowledge) that were implicit in the original model represented using a lightweight ontology language and that became explicit by applying methodological directives underlying an ontologically well-founded modeling language.

1 INTRODUCTION

There are two common trends in the traditional use of the term ontology in computer science: (i) firstly, ontologies are typically regarded as an explicit representation of a shared conceptualization, i.e., a concrete artifact representing a model of consensus within a community and a universe of discourse. In this sense of a reference model, an ontology is primarily aimed at supporting semantic interoperability in its various forms (e.g, model integration, service interoperability, knowledge harmonization, and taxonomy alignment); (ii) secondly, the discussion regarding representation mechanisms for the construction of domain ontologies is, typically, centered on computational issues, not truly ontological ones. An important aspect to be highlighted is the incongruence between these two approaches. In order for an ontology to be able to adequately serve as a reference model, it should be constructed using an approach that explicitly takes foundational concepts into account; this, however, is neglected for the sake of computational complexity.

The use of foundational concepts that take truly ontological issues seriously is becoming more and more accepted in the ontological engineering literature, i.e., in order to represent a complex domain, one should rely on engineering tools (e.g., design patterns), modeling languages and methodologies that are based on well-founded ontological theories in the philosophical sense (Fielding,2004; Burek, 2006). Especially in a domain with complex concepts, relations and constraints, and with potentially serious risks which could be caused by interoperability problems, a

Lopes M., Guizzardi G., Araujo Baião F. and Falbo R. (2009).
REVERSE ENGINEERING A DOMAIN ONTOLOGY TO UNCOVER FUNDAMENTAL ONTOLOGICAL DISTINCTIONS - An Industrial Case Study in the Domain of Oil and Gas Production and Exploration.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 262-267
DOI: 10.5220/0002014902620267

supporting ontology engineering approach should be able to: (a) allow the conceptual modelers and domain expert to be explicit regarding their ontological commitments, which in turn enables them to expose subtle distinctions between models to be integrated and to minimize the chances of running into a *False Agreement Problem* (Guarino, 1998); (b) support the user in justifying their modeling choices and providing a sound design rationale for choosing how the elements in the universe of discourse should be modeled in terms of language elements.

This marks a contrast to almost all languages traditionally used for knowledge representation and conceptual information modeling, in general, and in the semantic web, in particular (RDF, OWL, F-Logic, UML, EER). Although these languages provide the modeler with mechanisms for building conceptual structures (taxonomies or partonomies), they offer no support neither for helping the modeler on choosing a particular structure to model elements of the subject domain nor for justifying the choice of a particular structure over another. Finally, once a particular structure is represented, the ontological commitments which are made remain, in the best case, tacit in the modelers' mind. In the worst case, even the modelers and domain experts remain oblivious to these commitments.

An example of an ontologically well-founded modeling language is the one proposed in (Guizzardi, 2005), which is here dubbed *OntoUML*. This language has its real-world semantics defined in terms of a number of ontological theories, such as theory of parts, of wholes, types and instantiation, identity, dependencies, unity, etc. However, in order to be as explicit as possible regarding all the underlying subtleties of these theories (e.g., modal issues, different modes of predication, higher-order predication), this language strives for having its formal semantics defined in a logical system as expressively as possible. Now, as well understood in the field of knowledge representation, there is a clear tradeoff between logical expressivity and computational efficiency (Levesque & Brachman, 1987). In particular, any language which attempts at maximizing the explicit characterization of the aforementioned ontological issues risk sacrificing reasoning efficiency and computational tractability. In contrast, common knowledge representation and deductive database languages (e.g., some instances of Description Logics) have been designed to afford efficient automated reasoning and decidability.

In summary, ontology engineering must face the following situation: on one side, we need ontologically well-founded languages supported by expressive logical theories in order to produce sound and clear representations of complex domains; on the other side, we need lightweight ontology languages supported by efficient computational algorithms. How to reconcile these two sets of contradicting requirements? As advocated by (Guizzardi & Halpin, 2008), actually two classes of languages are required to fulfill these two sets of requirements. Moreover, as any other engineering process, an ontology engineering process lifecycle should comprise phases of conceptual modeling, design, and implementation. In the first phase, a reference ontology is produced aiming at representing the subject domain with truthfulness, clarity and expressivity, regardless of computational requirements. The main goal of these reference models is to help modelers to externalize their tacit knowledge about the domain, to make their ontological commitments explicit in order to support meaning negotiation, and to afford as best as possible the tasks of domain communication, learning and problem solving. The same reference ontology can then give rise to different lightweight ontologies in different languages (e.g., F-Logic, OWL-DL, RDF, Alloy, and KIF) so as to satisfy different sets of non-functional requirements. Defining the most suitable language for codifying a reference ontology is then a choice to be made at the design phase, by taking both the end-application purpose and the tradeoff between expressivity and computational tractability into account.

In this article, we illustrate the issues at stake in the aforementioned tradeoff by briefly discussing a fragment of an industrial case study in the domain of Oil and Gas Exploration and Production. However, since we were dealing with an existing OWL-DL codified ontology, we had to reverse the direction of model development. Instead of producing a reference model in OntoUML which would then give rise to an OWL-DL codification, we had to start with an OWL-DL ontology and apply a reverse engineer process to it in an attempt to reconstruct the proper underlying reference model in OntoUML. By doing this, we manage to show how much of important domain knowledge had either been lost in the OWL-DL codification or remained tacit in the mind of the domain experts.

The article is organized as follows. Section 2 briefly characterizes the domain and industrial setting in which the case study took place, namely, the domain of oil and gas exploration and production and in the context of a large Brazilian Petroleum Organization. Section 3 illustrates the reengineering

of the original lightweight ontology and its well-founded version represented in OntoUML. Section 4 discusses some final considerations.

2 THE CASE STUDY DOMAIN AND SETTINGS

The oil and gas industry is a potentially rich domain for application of ontologies, since it comprises a large and complex set of inter-related concepts. Ontology-based approaches for data integration and exchange involves the use of ontologies of rich and extensive domains combined with industry patterns and controlled vocabularies, reflecting relevant concepts within this domain (Chum, 2007). According to Chum, the motivating factors for the use of ontologies in the oil and gas industry include: (i) the great data quantity generated each day, coming from diverse sources, involving different disciplines, which hardens the data integration task; (ii) the existence of data in different formats, both structured in databases and semi-structured in documents, which hardens data search and access; and (iii) the need for standardization and integration of information along the frontiers of systems, disciplines and organizations, to better support decision-making processes.

The case study reported in this paper was conducted in a large Brazilian Petroleum Corporation, by analyzing and redesigning a pre-existing ontology in the domain of Oil and Gas Exploration and Production, henceforth named *E&P-Reservoir Ontology*. Due to the extensiveness and complexity of this domain, only a small part of it was addressed in the initial version of this ontology, namely, the "*Reserve Assessment"* sub domain, and the "*Mechanical pump"* sub domain.

The knowledge acquisition process used to create the original *E&P-Reservoir Ontology* ontology was conducted via the representations of business process models following the approach proposed in (Cappelli *et al*., 2007) and extended in (Baião et al, 2008). The original ontology was codified in OWL-DL comprising 178 classes, which together contained 55 data type properties (OWL datatypeProperties) and 96 object properties (OWL objectProperties).

In a nutshell, a *Reservoir* is composed of *Production Zones* and organized in *Fields* – geographical regions managed by a Business Unit and containing a number of *Wells*. Reservoirs are filled with *Reservoir Rock* – a substance composed of quantities of Oil, Gas and Water. *Production* of Oil and Gas from a Reservoir can occur via different lifting methods (e.g., natural lifting, casing's diameter, sand production, among others) involving different Wells. One of these artificial lifting methods is the *Mechanical Pump*. The simultaneous production of oil, gas and water occurs in conjunction with the production impurities. To remove these impurities, facilities are adopted on the fields (both off-shore and on-shore), including the transfer of hydrocarbons via *Ducts* to refineries for proper processing. The notion of *Reserve Assessment* refers to the process of estimating, for each Exploration Project and Reservoir, the profitable recoverable quantity of hydrocarbons (Oil and Gas) for that given reservoir. The *Mechanical Pump* sub domain ontology, in contrast, defines a number of concepts regarding the methods of fluid lifting, transportation, and other activities that take place in a reservoir during the production process.

For a more extensive definition of the concepts in this domain, one should refer, for instance, to (Thomas, 2001) or to *The Energy Standard Resource Center* (www.energistics.org).

3 REVERSE ENGINEERING THE E&P-RESERVOIR ONTOLOGY

In this section, we briefly discuss some of the results of producing an OntoUML version of the original E&P-Reservoir ontology in this domain. In particular, we focus at illustrating a couple of important concepts in this domain which were absent in the original OWL model and remained tacit in the domain experts' minds, but which became manifest by the application of methodological directives underlying OntoUML. The reverse engineering process was conducted by systematically searching for the real-world semantics of each concept in the original ontology. This, in turn, was done by analyzing the meta-properties defined for each construct of the OntoUML language in (Guizzardi, 2005) so as to decide the most adequate element to represent the concept.

This section does not aim at serving as an introduction to OntoUML. In fact, due to space limitations we will focus here on one specific ontological distinction, namely, the distinction between *formal* and *material relations* and the associated notion of *relators* (Guizzardi & Wagner, 2008).

3.1 Making the Real-world Semantics of Relationships Explicit

Figure 1 depicts a fragment of the original E&P-Reservoir Ontology and figure 2 depicts the corresponding redesigned fragment in OntoUML.

The OntoUML language, with its underlying methodological directives, makes an explicit distinction between the so-called *material* and *formal relationships*. A formal relationship can be reduced to relationships between intrinsic properties of its relata. For example, a relationship *more-dense-than* between two fluids can be reduced to the relationship between the individual densities of the involved fluids (*more-dense-than(x,y) iff the density of x is higher than of y's*), In contrast, material relationships cannot be reduced to relationships between individual properties of involved relata in this way. In order to have a material relationship established between two concepts C1 and C2, another entity must exist that make this relationship true. For example, we can say that the Person John works for Company A (and not for company B) if an employment contract exists between John and Company A which makes this relationship true. This entity, which is the truthmaker of material relationships, is termed *relator* in OntoUML and the language determines that these *relators* must be explicitly represented on the models (Guizzardi & Wagner, 2008).

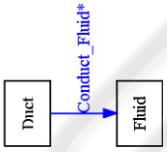

Figure 1: Representation of Fluid transportation (OWL).

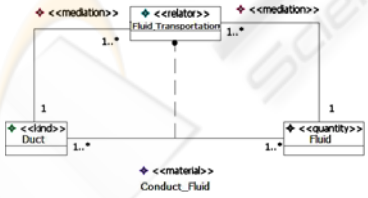

Figure 2: Alternative Representation of Fluid transportation (OntoUML).

The *Conduct_Fluid* relationship of figure 1 is an example of a material relationship. This relationship only takes place (i.e., the *Conduct_Fluid* relationship is only established) between a specific duct x and a specific portion of fluid y, when there is at least a fluid transportation event that involves the participation of x and y.

Besides making explicit the truthmakers of these relations, one of the major advantages of the explicit representation of *relators* is to solve an inherent ambiguity of cardinality constraints that exists in material relationships. Take for example the cardinality constraints of one-to-many represented for the relationship *Conduct_Fluid* in figure 1 (minimum cardinality constraints are not shown). There are several possible interpretations for this model which are compatible with these cardinality constraints but which are mutually incompatible among themselves. Two of these interpretations are depicted in figures 3 and 4.

On the model of figure 3, given a fluid transportation event, we have only one duct and only one portion of fluid involved; both fluid and duct can participate in several transportation events. In contrast, on the model of figure 4, given a fluid transportation event, we have possibly several ducts and portions of fluid involved; a duct can be used in several transportation events, but only one fluid can take part on a fluid transportation.

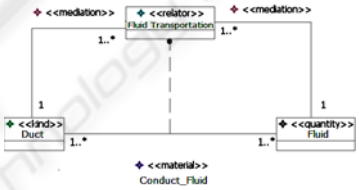

Figure 3: Interpreting Fluid transportation with unique Duct and Fluid.

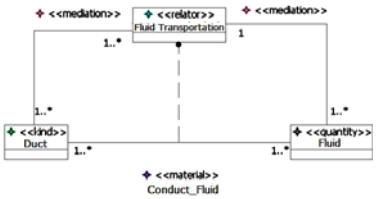

Figure 4: Interpreting Fluid transportation with multiples Ducts and Fluids.

When comparing these two models in OntoUML we can see that the original model collapses these two interpretations (among others) in the same representation, which have substantially different semantics. This semantic overload can be a source of many interoperability problems between applications. In particular, applications that use different models and that attach distinct semantics to relationships such as discussed above can wrongly assume that they agree on the same semantics (an example of the previously mentioned *False Agreement Problem*).

Finally, in the OntoUML models in this section, the dotted line with a filled circle on one of its endings represents the *derivation* relationship between a *relator type* and the *material relationship* derived from it (Guizzardi, 2005). For example, the derivation relationship *Fluid Transportation* (*relator type*) and *Conduct_Fluid* (*material relationship*) represents that for all x, y we have that: <x,y> is an instance of *Conduct_Fluid* iff only there is an instance z of *Fluid Transportation* that *mediates* x and y. As discussed in depth in (Guizzardi, 2005), mediation is a specific type of existential dependence relation (e.g., a particular Fluid Transportation can only exist if that particular Duct and that particular Fluid exist). Moreover, it also demonstrated that the cardinality constraints of a material relationship R derived from a relator type U_R can be systematically derived from the corresponding *mediaton* relationships between U_R and the types related by R. In summary, a relator is an entity which is existentially dependent on a number of other individuals, and via these dependency relationships it connects (mediates) these individuals. Given that a number of individuals are mediated by a relator, a material relationship can be defined between them. As this definition makes clear, relators are ontologically prior to material relationships which are mere logical/linguistic constructions derived from them (Guizzardi, 2005). To put it in a different way, knowing that x and y are related via R tells you very little unless you know what are the conditions (state of affairs) that makes this relationship between this particular tuple true.

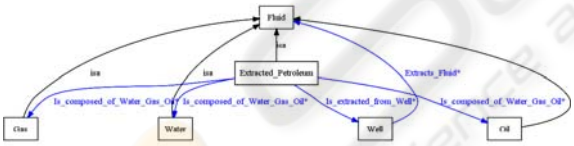

Figure 5: Representation of Fluid and related notions in the original OWL ontology.

Another example of a material relation is the *Extracts_Fluid* relationship between *Fluid* and *Well* in the model of figure 5. Once more, following the methodological directives of the language, the modeling process seeks to make explicit which would be the *relator* that would substantiate that relationship. The conclusion that one would come is that the relationship *Extracts_Fluid(x,y)* is true iff there is a *Production* event involving the Well x from where the Fluid y is produced. The semantic investigation of this relationship makes explicit that the resulting fluid of this event in fact only exists after the occurrence of this event. In other words, the portion of the Extracted Petroleum only exists after it is produced from the event of production involving a well. Therefore, a mixture of water, gas and oil is considered *Extracted Petroleum* only when it is produced by an event of this kind.

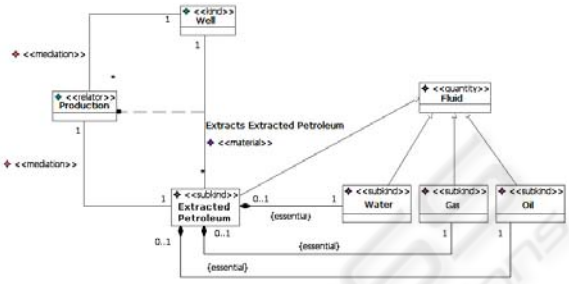

Figure 6: The Representation of Fluid and related notions in the re-designed ontology (OntoUML).

In the model of figure 6, the original *Extract_Fluid* relationship between Well and Fluid and the *Is_extracted_from_Well* relationship between Extracted Petroleum and Well on the original ontology are replaced by the material relationship *Extracts_Extracted_Petroleum* between Well and Extracted Petroleum and by the *partOf* relationships between the Extracted Petroleum portion and its sub portions of Water, Gas and Oil. This representation has the additional benefit of making clear that an event of Production has the goal of generating an Extracted Petroleum portion that is composed of particular portions of these Fluid types and not by directly extracting portions of these other types of fluid. Finally, as previously discussed, the explicit representation of the *Production* relator makes the representation of the cardinality constraints involving instances of Well and Extracted Petroleum precise, eliminating the ambiguity on the representation of the *Extract_Fluid* relationship on the original model.

4 FINAL CONSIDERATIONS

An ontology engineering process is composed of phases, among them conceptual modeling and implementation. During the whole process, the ontology being built must be made explicit by a representation language. The diverse ontology representation languages available on the literature contain different expressivity and different ontological commitments, reflecting on the specific set of available constructs in each one of them. Therefore, different ontology representation

languages, with different characteristics, are suitable to be used in different phases of the ontology engineering process so as to address the different set of requirements which characterizes each phase.

In particular, conceptual ontology modeling languages aim primarily at improving understanding, learning, communication and problem solving among people in a particular domain. Therefore, these languages have being designed to maximize expressivity, clarity and truthfulness to the domain being represented. In contrast, ontology codification languages are focused on aspects such as computational efficiency and tractability and can be used to produce computationally amenable versions of an ontologically well-founded reference conceptual model. The inadequate use of a representation language, disregarding the goal of each ontology engineering phase, can lead to serious problems to database design and integration, to domain and systems requirements analysis within the software development processes, to knowledge representation and automated reasoning, and so on.

This article presents an illustration of these issues by using a fragment of an industrial case study in the domain of Oil and Gas Exploration and Production. The case study consists in generating a Conceptual Ontological Model for this domain from an existing domain ontology.

The ontology representation language used to produce the redesigned model was OntoUML, a theoretically sound and highly expressive language based on a number of Formal Ontological Theories. The choice of this language highlights a number of explicit concepts and ideas (tacit domain knowledge) that were implicit in the original model coded in OWL-DL. By making these concepts explicit as well as defining a precise real-world semantics for the notions represented, the newly E&P-Reservoir ontology produced in OntoUML prevents a number of ambiguity and interoperability problems which would likely be carried out to subsequent activities (e.g., database design) based on this model.

ACKNOWLEDGEMENTS

The authors would like to thank Petrobras for the case study.

REFERENCES

Artale, A., Keet, M., 2008. Essential and Mandatory Part-Whole Relations in Conceptual Data Models, *21st Int Workshop Description Logics*.

Baião, F. *et al.*, 2008. *Towards a Data Integration Approach based on Business Process Models and Domain Ontologies*. 10th Int Conf on Enterprise IS (ICEIS2008), Barcelona, 338-342.

Burek, P. *et al.*, 2006. A top-level ontology of functions and its application in the Open Biomedical Ontologies. *Bioinformatics* 22(14), pp. e66-e73.

Cappelli, C. *et al.*, 2007. An Approach for Constructing Domain Ontologies from Business Process Models (in Portuguese), *II Workshop on Ontologies and Metamodeling in Software and Data Engineering (WOMSDE)*, João Pessoa.

Chum, F., 2007. Use Case: Ontology-Driven Information Integration and Delivery - A Survey of Semantic Web Technology in the Oil and Gas Industry, W3C. Available in: http://www.w3.org/2001/sw/sweo/public/UseCases/Chevron/. Accessed in Dec 2007.

Fielding, J. *et al.*, 2004. Ontological Theory for Ontology Engineering. *Int Conf. on Principles of Knowledge Representation and Reasoning (KR 2004)*, Canada.

Guarino, N., 1998, Formal Ontology and Information Systems. *1st Int Conf on Formal Ontologies in Information Systems*, 3-15, Trento.

Guizzardi, G., 2005. Ontological Foundations for Structural Conceptual Models, *Telematica Instituut Fundamental Research Series* 15, ISBN 90-75176-81-3, Universal Press.

Guizzardi, G.; Halpin, T., 2008. Ontological Foundations for Conceptual Modeling, *Applied Ontology* 2 (1-2), pp. 91-110, ISSN 1570-5838.

Guizzardi, G.; Wagner, G., 2008. What´s in a Relationship: An Ontological Analysis, 27th International Conference on Conceptual Modeling (ER 2008), Barcelona. Lecture Notes in Computer Science, v.5231, p.83 - 97, 2008.

das Graças, A. 2008. Extending a Model-Based Tool for Ontologically Well-Founded Conceptual Modeling with Rule Visualization Support. *Computer Engineering Monograph*, NEMO Research Group, Federal University of Espirito Santo, Brazil.

Keet, M.; Artale, A., "Representing and Reasoning over a Taxonomy of Part-Whole Relations", in Guizzardi, G. and Halpin, T. (Editors), Special Issue on Ontological Foundations for Conceptual Modeling, Applied Ontology, pp. 91-110, Volume 3, Number 1-2 / 2008, ISSN 1570-5838.

Levesque, H.; Brachman, R. 1987., Expressiveness and Tractability in Knowledge Representation and Reasoning. *Computational Intelligence* 3 (1), pp.78-93.

Thomas, J. E., 2001, Fundamentals of Petroleum Engineering, *Interciência* (*in Portuguese*).