On the importance of Truly Ontological Distinctions for Ontology Representation Languages: An Industrial Case Study in the Domain of Oil and Gas

Giancarlo Guizzardi[1], Mauro Lopes[2,3], Fernanda Baião[2,3], Ricardo Falbo[1]

[1]Ontology and Conceptual Modeling Research Group (NEMO), Computer Science Department, Federal University of Espírito Santo, Espírito Santo, Brazil
[2]NP2Tec – Research and Practice Group in Information Technology, Federal University of the State of Rio de Janeiro (UNIRIO), Rio de Janeiro, Brazil
[3]Department of Applied Informatics, Federal University of the State of Rio de Janeiro (UNIRIO), Rio de Janeiro, Brazil
{gguizzardi, falbo}@inf.ufes.br
{fernanda.baiao, mauro.lopes}@uniriotec.br

Abstract. Ontologies are commonly used in computer science either as a reference model to support semantic interoperability, or as an artifact that should be efficiently represented to support tractable automated reasoning. This duality poses a tradeoff between expressivity and computational tractability that should be addressed in different phases of an ontology engineering process. The inadequate choice of a modeling language, disregarding the goal of each ontology engineering phase, can lead to serious problems in the deployment of the resulting model. This article discusses these issues by making use of an industrial case study in the domain of Oil and Gas. We make explicit the differences between two different representations in this domain, and highlight a number of concepts and ideas that were implicit in an original OWL-DL model and that became explicit by applying the methodological directives underlying an ontologically well-founded modeling language.

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

1 Introduction

Since the word ontology was mentioned in a computer related discipline for the first time [1], ontologies have been applied in a multitude of areas in computer science. The first noticeable growth of interest in the subject in mid 1990's was motivated by the need to create principled representations of domain knowledge in the knowledge sharing and reuse community in Artificial Intelligence (AI). Nonetheless, an explosion of works related to the subject only happened in the past eight years, highly motivated by the growing interest on the Semantic Web, and by the key role played by ontologies in that initiative.

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. Moreover, 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 trends. 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 is, however, typically 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 (e.g., [2]; [3]). Especially in a domain with complex concepts, relations and constraints, and with potentially serious risks which could be caused by interoperability problems, a supporting ontology engineering approach should be able to: (a) allow the conceptual modelers and domain experts 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* [4]; (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 practically all languages used in the tradition of knowledge representation and conceptual information modeling, in general, and in the semantic web, in particular (e.g., RDF, OWL, F-Logic, UML, EER). Although these languages provide the modeler with mechanisms for building conceptual structures (e.g., 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 version of UML 2.0 proposed in [5] and, thereafter, 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 [6]. In particular, any language which attempts at maximizing the explicit characterization of the aforementioned ontological issues

risks sacrificing reasoning efficiency and computational tractability. In contrast, common knowledge representation and deductive database languages (e.g., some instances of Description Logics) have been specifically 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 [7], 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) and satisfying 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 discussing an industrial case study in the domain of Oil and Gas Exploration and Production. However, since we were dealing with a pre-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 the OWL-DL domain ontology and apply a reverse engineering process to it in an attempt to reconstruct the proper underlying reference model in OntoUML. By doing that, we manage to show how much of important domain knowledge had either been lost in the OWL-DL codification or remained tacit in the minds of the domain experts.

The remainder of this article is organized as follows. Section 2 briefly characterizes the domain and industrial setting in which the case study reported in this article took place, namely, the domain of oil and gas exploration and production and in the context of a large Petroleum Organization. Section 3 discusses the reengineering of the original lightweight ontology produced in the settings described in section 2. This reengineering step was conducted by transforming the original ontology to well-founded version represented in OntoUML. Section 4 discusses some final considerations.

2 Characterization of 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 [8]. According to this author, the motivating factors for the use of ontologies in the oil and gas industry include:

- The great data quantity generated each day, coming from diverse sources, involving different disciplines. Integrating different disciplines to take advantage of the real value of your information has been a complex and costly task.
- The existence of data in different formats, including structured in databases and semi-structured in documents. To deal with the great quantity of information, as well as heterogeneous formats, a new approach is needed to handle information search and access.
- The necessity of standardization and integration of information along the frontiers of systems, disciplines and organizations, to support the decision-making with the collaborators, to the extent that better quality data will be accessible on the opportune time.

The case study reported in this paper was conducted in a large 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 few sub domains were taken into consideration on 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 [9] and extended in [10]. The original E&P-Reserve 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 profitably recoverable quantity of hydrocarbons (Oil and Gas) for that given reservoir. The *Mechanical Pump* subdomain 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 to, for instance, [11] or *The Energy Standard Resource Center* (www.energistics.org).

3 Reverse engineering an OntoUML version of the E&P-Reserve Ontology

In this section, we discuss some of the results of producing an OntoUML version of the original E&P-Reserve Ontology in this domain. In particular we focus at illustrating a number 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. It is important to emphasize that this section does not aim at serving as an introduction to OntoUML neither as a complete report on the newly produced version of the E&P-Reserve Ontology.

3.1 Making the Real-World Semantics of Relationships Explicit

Figure 1 depicts a fragment of the OWL ontology and figure 2 depicts the correspondent fragment transformed to 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 makes 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 (for the case of material relationships) these *relators* must be explicitly represented on the models [12].

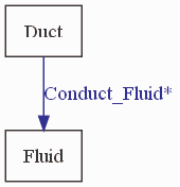

Fig. 1. Representation of *Fluid transportation* (OWL).

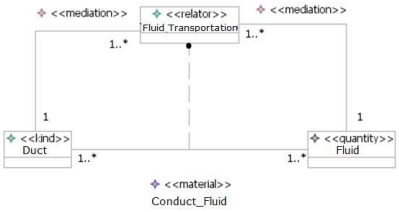

Fig. 2. Alternative Representation of Fluid transportation (OntoUML), an interpretation of Fluid transportation with unique Duct and Fluid.

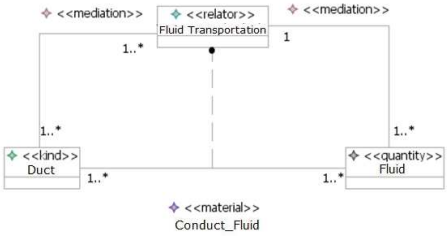

Fig. 3. Interpreting Fluid transportation with multiples Ducts and Fluids.

The *Conduct_Fluid* relationship of figure 1 is an example of a material relationship. Therefore, 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. 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 2 and 3.

On the model of figure 2, 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 3, 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.

When comparing these two models in OntoUML we can see that the original OWL model collapses these two interpretations (among others) in the same representation, which have substantially different real-world 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 [5]. 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 there is an instance z of *Fluid Transportation* that mediates x and y. As discussed in depth in [5,12], 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 automatically 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 [5,12]. 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.

3.2 The Ontological Status of Quantities

Figures 4 and 5 represent fragments of the domain ontology that deal with the notion of Fluid.

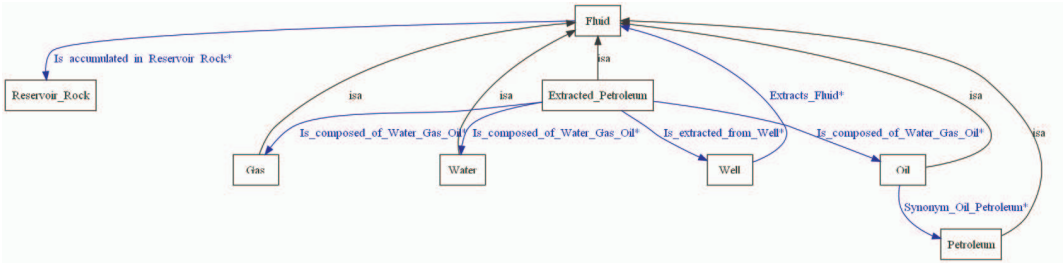

Fig. 4. The representation of *Fluid* and related notions in OWL.

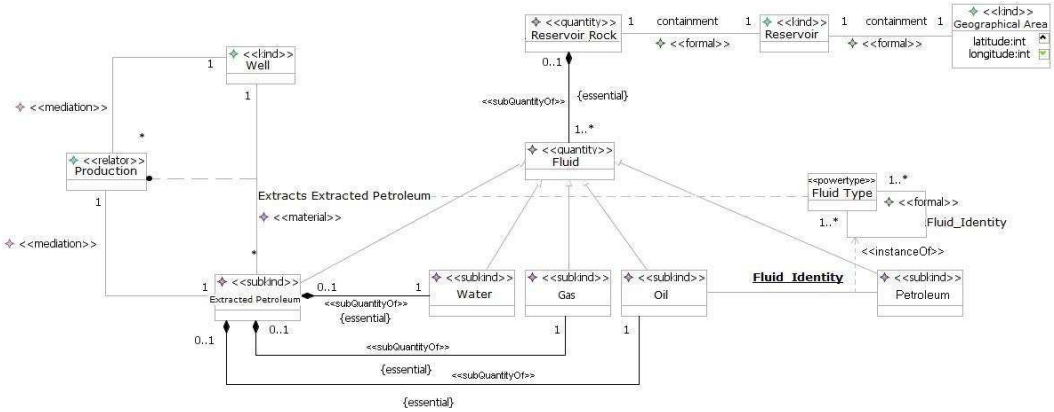

Fig. 5. The Representation of *Fluid* and related notions in OntoUML

In general, quantities or amounts of matter (e.g., water, milk, sugar, sand, oil) are entities that are homeomerous, i.e., all of their parts are the same type as the whole. Alternatively, we can say that they are infinitely divisible in subparts of the same type. Homeomerousity and Infinite divisibility causes problems both to determine the referent of expressions referring to quantities and, as a consequence, also problems to specify finite cardinality constraints of relationships involving quantity types [5]. In OntoUML, these problems are avoided by defining a modelling primitive <<quantity>> whose semantics are defined by invoking the ontological notion of *Quantity*. In OntoUML, a type stereotyped as <<quantity>> represents a type whose instances represent portions of amounts of matter which are maximal under the relation of topological self-connectness [5].

In figure 5, the type Fluid is represented as a *quantity* in this ontological sense. As a consequence we have that Fluid: (i) is a rigid type, i.e., all instances of this type are necessarily instances of this type (in a modal sense); (ii) provides an identity principle obeyed by all its instances; (iii) represent a collection of essential properties of all its instances [5,13]. Specializations of a *quantity* are represented with the stereotype *subkind*. In figure 5, these include the specific types of Fluid:Water, Oil and Gas. Subkinds of Fluid have meta-properties (i) and (iii) above by inheriting the principle of identity defined by the quantity kind Fluid that should be obeyed by all its subtypes.

On the original ontology in OWL, the equivalence between the Oil and Petroleum concepts is represented by the *Oil_Petroleum_synonym* relationship defined between these concepts. This relationship is declared as being symmetric. On the original ontology, these concepts simply represent the general concepts of Oil or Petroleum and do not represent genuine types that can be instantiated. As consequence in this case, the *Oil_Petroleum_synonym* relationship represents also a relational type that cannot be instantiated and only exists in fact between this pair of concepts. Therefore, it does not make sense to characterize it as a symmetric relationship, since it functions as an instance and not genuinely as a type.

In the semantics adopted on the revised model, Oil and Petroleum are *quantity* types, the instances of which are specific portions of these Fluids. Therefore, in this

case, there is no sense in defining an *Is_synonym_of* relationship between Oil and Petroleum. After all, defined this way, since these are genuine types that can be instantiated, this relationship would have as instances ordered pairs formed by specific portions of Oil and Petroleum, which definitely does not correspond to the intended semantics of this relationship. In fact, the relationship *Is_synonym_of* is a relationship between the Oil and Petroleum types and not between its instances. In particular, this relationship has a stronger semantics than simply symmetry, being an equivalence relationship (reflexive, symmetric, transitive).

The problem of the proper representation of an *Is_synonym_of* relationship that could be established between any two types of fluid is solved on the model of figure 5. Firstly, the model makes an explicit distinction between the fluid types instances of which are individual portions of fluid and a type instances of which are the concepts of Oil, Water, Gas and Petroleum themselves. Since OntoUML is an extension of standard UML, this can be represented by the use of notion of *powertype*[1]. In a nutshell, a *powertype* is a type instances of which are other types. On this specific model, the relationship between the *Fluid Type powertype* and *Fluid* defines that the subtypes of the latter (Oil, Water, Gas and Petroleum) are instances of the former. Once this distinction is made, the formal relationship of *Fluid_identity*[2] can be defined among the instances of *Fluid Type*. This relationship can, then, be defined as an equivalence relationship which semantics is characterized by the following rule: two fluid types are identical iff they possess necessarily (i.e., at any given circumstance) the same instances. In the OntoUML language, this rule is defined outside the visual syntax of the language and as part of the axiomatization of the resulting model (ontology).

Finally, as a result of this modeling choice, particular instances of the *Fluid_identity* relationship can be defined. For example, in figure 5, the *link* (instance of a relationship) between Oil and Petroleum (instances of *Fluid Type*) is defined explicitly as an instance of *Fluid_Identity*.

In the revised model of figure 5, in the same manner as Fluid and its subtypes, *Reservoir Rock* is explicitly represented as a *quantity* type. Once more, this type represents a genuine type instances of which are particular portions of *Reservoir Rock*. The *Is_accumulated_in_Reservoir_Rock* relationship in the original model of figure 4 is, hence, replaced by a special type of part-whole relationship (*subQuantityOf*) between Reservoir Rock and Fluid. The SubQuantityOf relationship defined as a primitive in OntoUML contains a formal characterization that implies: (i) a partial order (irreflexivity, asymmetry, transitivity) relation; (ii) An existential dependency relation, i.e., in this particular example a particular portion of Reservoir Rock is defined by the aggregation of the specific particular portions of its constituent Fluids; and (iii) Non-sharing of parts, i.e., each particular portion of fluid is part of at most one portion of Reservoir Rock. It is important to emphasize that the explicit representation of the semantics of this relationship eliminates an implicit ambiguity on the original model.

---

[1] http://www.omg.org/spec/UML/2.1.2/

[2] The preference for the term *Fluid_identity* instead of *Is_synonym_of* is motivated by the fact that the former refers to an identity relation among types while the latter refers merely to an identity relation among terms.

3.3 The *Containment* relation to represent the spatial inclusion among physical entities: Reservoir, Reservoir Rock and Geographic Area

The model on figure 5 also depicts the Reservoir and Geographic Area concepts and defines the formal relationship of *containment* [14] between Reservoir and Reservoir Rock and between Reservoir and Geographic Area. This relationship contains the semantic of spatial inclusion between two physical entities (with the spatial extension) that is also defined on the ontology's axiomatization, e.g., outside the visual syntax of the model.

On the original model of figure 4, there is only one relationship *Is_composed_of_Water_Gas_Oil* defined between the Extracted Petroleum and the Water, Gas and Oil concepts. On the revised ontology, this relationship is replaced by composition relationships (*subQuantityOf*). As previously discussed, the richer semantics of this relationship type makes important meta-properties of the relationship among these elements explicit in the model. As discussed in [5, 15, 16], the formal characteristics of this relationship, modeled as a partially order, existential dependency relation with non-sharing of parts, have important consequences both to the design and implementation of an information system as to the automated processes of reasoning and model evaluation.

3.4 Making the *Production* Relator Explicit

As already discussed, OntoUML makes an explicit distinction between formal and material relationships. The *Extracts_Fluid* relationship between *Fluid* and *Well* in the original model is an example of the latter. In this way, following the methodological directives of the language, the modeling process seeks to make explicit which is the appropriate relator that would substantiate that relationship. The conclusion would one come to 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. The *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 *subQuantityOf* 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.

3.5 Representing the Historical Dependence between *Extracted Petroleum* and *Reservoir Rock*

As previously discussed, the *subquantityOf* relation defined in OntoUML to hold between portions of quantities is a type of existential dependency relation from the whole to the part. In other words, all parts of a quantity are essential parts of it. For instance, in figure 6, we have the type Reservoir Rock stereotyped as <<quantity>>. As a consequence, once we have the case that specific portions of water, gas and oil are extracted from a specific portion of Reservoir Rock *x* (creating a portion of Extracted Petroleum *y*) that specific portion *x* ceases to exists. Indeed, the resulting portion of Extracted Petroleum *y* and the Reservoir Rock *x* from which *y* originates cannot co-exist at the same circumstances. In fact, the same event that creates the former is the one that destroys the latter. However, it is important to represent the specific connection between *x* and *y*, for instance, because some characteristics from an Extracted Petroleum could result from characteristics of that Reservoir Rock. Here, this relation between *x* and *y* is modeled by the formal relation of *historical dependence* [17]: in this case, since *y* is historically dependent on *x* it means that *y* could not exist without *x* having existed.

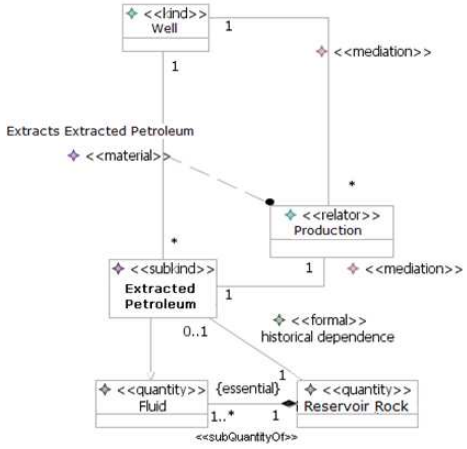

Fig. 6. Extracted Petroleum and its historical dependence to a Reservoir Rock

4 Final Considerations

An ontology engineering process is composed of phases, among them are 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 in 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 characterize 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 an industrial case study in the domain of Oil and Gas Exploration and Production. The case study consists in the generation of a Conceptual Ontological Model for this domain from an existing domain ontology in the organization where the case study took place.

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. To cite just one example, in the original representation of *Conduct_Fluid* relationship, it is possible to define that a duct can conduct several fluids and a fluid can be conducted by several different ducts. However, the lack of the Fluid Transportation concept (a relator uncovered by the methodological directives of OntoUML) hides important information about the domain. For instance, it is not explicit in this case how many different fluids can be transported at the same time or even if a duct can have more than a fluid transportation at a time. By making these concepts explicit as well as defining a precise real-world semantics for the notions represented, the newly E&P-Reserve 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.

In [18], an extension of OntoUML (OntoUML-R) is presented. This version of the language allows for the visual representation of domain axioms (rules), including integrity and derivation axioms in OntoUML. As future work, we intend to exploit this new language facility to enhance the transformed E&P-Reserve Ontology with visual representations of domain axioms. This enhanced model can then be mapped to a new version of the OWL-DL codified lightweight ontology, now using a combination of OWL-DL and SWRL rules. This enhanced lightweight model, in turn, shall contemplate the domain concepts uncovered by the process described in this article and, due to the combination of OWL-DL and SWRL, afford a number of more sophisticated reasoning tasks.

References

1. Mealy, G. H.: Another Look at Data. Proceedings of the Fall Joint Computer Conference, November 14–16, Anaheim, California (AFIPS Conference Proceedings,

Volume 31), Washington, DC: Thompson Books, London: Academic Press, 525–534, 1967.
2. Burek, P. et al.: A top-level ontology of functions and its application in the Open Biomedical Ontologies. Bioinformatics Vol. 22(14), pp. e66-e73, 2006.
3. Fielding, J. et al.: Ontological Theory for Ontology Engineering. In: International Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004), 9th, Whistler, Canada, Proceedings, 2004.
4. Guarino, N.: Formal Ontology and Information Systems. In: 1st International Conference on Formal Ontologies in Information Systems, pp. 3-15, Trento, Italy, June, 1998.
5. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models, Telematica Instituut Fundamental Research Series No. 15, ISBN 90-75176-81-3, Universal Press, The Netherlands, 2005.
6. Levesque, H.; Brachman, R.: Expressiveness and Tractability in Knowledge Representation and Reasoning. Computational Intelligence, 3(1): 78-93, 1987.
7. Guizzardi, G.; Halpin, T.: Ontological Foundations for Conceptual Modeling, Applied Ontology, pp. 91-110, Vol. 3, Number 1-2 / 2008, ISSN 1570-5838, 2008.
8. Chum, F.: Use Case: Ontology-Driven Information Integration and Delivery - A Survey of Semantic Web Technology in the Oil and Gas Industry, W3C, April 2007. Available in: http://www.w3.org/2001/sw/sweo/public/UseCases/Chevron/. Accessed in Dec 2007.
9. Cappelli, C., Baião, F., Santoro, F., Iendrike, H., Lopes, M., Nunes, V. T.: An Approach for Constructing Domain Ontologies from Business Process Models (in Portuguese). II Workshop on Ontologies and Metamodeling in Software and Data Engineering (WOMSDE), 2007.
10. Baião, F., Santoro, F., Iendrike, H., Cappelli, C., Lopes, M., Nunes, V. T. and Dumont, A. P.: Towards a Data Integration Approach based on Business Process Models and Domain Ontologies. 10th International Conference on Enterprise Information Systems (ICEIS2008), Barcelona, pp. 338-342, 2008.
11. Thomas, J. E.: Fundamentals of Petroleum Engineering. Rio de Janeiro, Interciência, in Portuguese, 2001.
12. Guizzardi, G.; Wagner, G., 2008. What´s in a Relationship: An Ontological Analysis, 27th International Conference on Conceptual Modeling (ER 2008), Barcelona, 2008.
13. Guizzardi, G., Wagner, G., Guarino, N., van Sinderen, M., 2004. An Ontologically Well-Founded Profile for UML Conceptual Models. 16th International Conference on Advances in Information Systems Engineering (CAiSE), Latvia, 2004.
14. Smith, B. et al., "Relations in biomedical ontologies", Genome Biology. 6(5), 2005.
15. Artale, A.; Keet, M.: Essential and Mandatory Part-Whole Relations in Conceptual Data Models. 21st International Workshop on Description Logics, Dresden, (2008).
16. 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, 2008.
17. Thomasson, A. L., "Fiction and Metaphysics", Cambridge University Press, ISBN-13: 9780521065214, 1999.
18. das Graças, A.: Extending a Model-Based Tool for Ontologically Well-Founded Conceptual Modeling with Rule Visualization Support. Computer Engineering Monograph, Ontology and Conceptual Modeling Research Group (NEMO), Federal University of Espirito Santo, Brazil, 2008.