Domain class diagram validation procedure based on mereological analysis for part-whole relations

Bruna Carolina de Melo Catossi [1]
Alcione de Paiva Oliveira [1]
Jugurta Lisboa Filho [1]
José Luis Braga [1]

Resumo: A dificuldade dos desenvolvedores de software para construir modelos conceituais é atribuída, em parte, à falta de conhecimento do domínio. Existem algumas técnicas de análise ontológica para ajudar o modelador durante o processo de criação do diagrama de classes. No entanto, essas acabam não sendo práticas e não é fácil perceber os seus reais benefícios no desenvolvimento de software, pois envolvem muitos conceitos filosóficos, o que as tornam complexas para modeladores comuns. Por esse motivo, procedimentos capazes de simplificar o entendimento desses conceitos e que se aproximam da realidade prática dos desenvolvedores têm surgido, como o PrOntoCon, que será discutido neste trabalho. O objetivo principal do PrOntoCon é guiar o modelador durante o processo de validação de um diagrama de classes UML para um determinado domínio de aplicação, focando, especialmente, os relacionamentos de especialização/generalização. No entanto, o PrOntoCon não guia o desenvolvedor na tarefa de modelar relacionamentos de associação simples, composição e agregação. O presente trabalho apresenta uma extensão do procedimento PrOntoCon, que adiciona uma fase para ajudar o desenvolvedor a lidar com esses tipos de relacionamento.

***Palavras-chave*:** Análise ontológica. Modelagem conceitual. Processo de software.

Abstract: *The difficulty of software developers face to build liable conceptual models occurs due to the lack of domain knowledge. There are some ontological analysis techniques that can help the modeler during the process of creating the domain class diagram. However, they end up not being easy to use as it involves many philosophical concepts, which makes them complex to the common modeler. Therefore, procedures that simplify the understanding of these concepts and are closer to the reality of developers has emerged. One of them is the PrOntoCon procedure, which will be discussed in this paper. PrOntoCon is a procedure that combines the UML modelling expressiveness with ontological analysis theory to create a procedure able to help generate conceptual models that are clearer and that can generate more robust and maintainable systems. The main objective of PrOntoCon is to guide the modeler during the validation process of a UML domain class diagram, focusing especially, the generalization/specialization relationships. Nonetheless, PrOntoCon initial version does not address relations of simple association and aggregation, also called part-whole relations. This paper presents an extension of PrOntoCon procedure that adds a phase to assist the developer to deal with these types of relationships.*

***Keywords*:** *Ontological analysis. Conceptual modeling. Software process.*

1 Introduction

An incorrect conceptual model, or a conceptual model that does not represent the domain adequately, can lead to problems with respect to systems project, implementation, operation and maintenance. The difficulty inherent to the task of modelling represents one of the main challenges software developers face nowadays. UML has become the standard modelling language and is widely applied on conceptual models construction in order to

[1]Curso de Mestrado em Ciência da Computação,Universidade Federal de Viçosa UFV - Brasil
{mestrecarrasco@gmail.com,alcione@gmail.com,jugurta@ufv.br,zeluisdpiufv@gmail.com}

http://dx.doi.org/10.5335/rbca.2014.3771

establish a straight connection between conceptual elements and executable ones. However, UML on its own is not able to guarantee a model that is free of conceptual mistakes. On trying to improve this situation techniques based on ontological analysis have been created to help modelers validate their UML class diagrams more easily. Among them, we can cite *VERONTO* Technique [1] and *OntoUML Profile* [2]. When compared, VERONTO and OntoUML Profile have many features in common and some complementary characteristics, since both of them were based on Guarino and Welty meta-properties [3, 4]. Also they were both used as the basis for the creation of a technique for structuring domain class diagram called *OntoCon* [5]. It was also felt the need of creation of a procedure that would guide the developer in the use of the diagram validation technique based on ontological properties.Thus, the PrOntoCon procedure was created [5]. The PrOntoCon procedure focused on the generalization/specialization relationships, source of many errors among developers. The procedure guides the modeler through a sequence of steps with the aim to validate existing UML class diagrams, by checking their relations and roles and detecting their possible fails. Nonetheless, the PrOntoCon initial version does not address relations of simple association and aggregation, also called part-whole relations. The creation of additional steps to be inserted into PrOntoCon become a necessity due to the difficulty faced by developers in understanding and representing these types of relationships, in order to eliminate, or at least reduce, ambiguities and misunderstandings generated by their use. The literature reports the difficulties and confusions related to this type of relationship [6]. This paper presents an extension of PrOntoCon procedure that adds a phase to assist the developer to deal with these types of relationships.

The article is organized as follows. Section 2 briefly discusses the concepts of Mereology and Meronymy. Section 3 describes the extended PrOntoCon procedure. Section 4 addresses the main procedure contributions to the case study in question. Finally, Section 5 concludes the ideas contained in this paper.

2 Mereology and Meronymy

According to [7] the development of a theory of parts forming a whole can be traced back to the early days of philosophy, beginning with the Presocratics and continuing throughout the writings of Plato, Aristotle (especially the *Metaphysics*) and Boethius. Because it is a complex issue and loaded with many controversies, it's scientific and philosophical aspects are still widely discussed. The concept of part-whole structures has been a research issue for ontology, cognitive science and linguistics [2, 8, 7, 9]. Mereology and Meronymy are the names given to two branches of study on the part-whole relations. Mereology is the investigation about the formal ontology of part-whole relationships, or relationships of part to whole and the relations of the part with another part within a whole[7]. Moreover, the Meronymy is the study of part-whole relation in the language. The mereology, as a formal theory, attempts to specify the general principles governing the behavior of part-whole structures as well as unravel the nature of these relationships. Therefore, mereology assumes that both parts as the whole should be concrete at the same time during their existence, but also assumes that they may be abstract at the same time during their existence. The common principles governing any base on the theory of part-whole relations are part of the so-called Ground Mereology, composed of three main restrictions: reflection, antisymmetric and transitivity. In natural language, they can be defined as follows [7]:

1. An object is a part of itself. 2. Two distinct things may not be part of one another. 3. Any part of any part of a thing is itself part of this thing.

Formally, these constraints can be expressed as follows, using a first-order logic where the binary predicate P indicates a part-whole relationship:

1. $\forall x P(x, x)$
2. $\forall xy(P(x, y) \wedge P(y, x)) \longrightarrow x = y$
3. $\forall xyz(P(x, y) \wedge P(y, z)) \longrightarrow P(x, z)$

The development of formal theories promotes a greater understanding of the concept of part, and one can formalize these concepts through axioms. However, there is still much controversy over some properties involved in part-whole relations, since they end up being inconsistent with the conceptual and cognitive theories applied

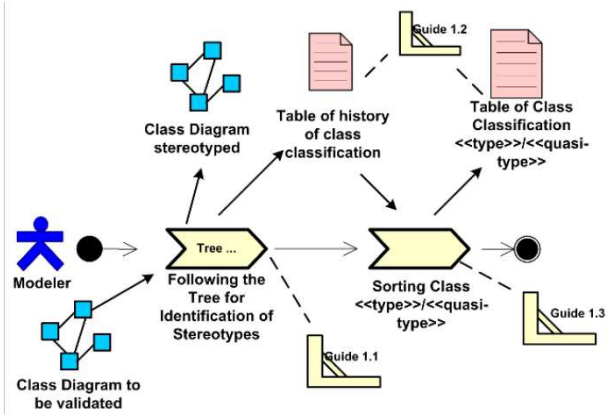
in the real world [2]. The transitivity of part-whole relationships, in particular, has provoked much discussion about its suitability for characterizing these types of structures (e.g. a part of a human cell could not be considered part of the body). Commonly, these questions occur due to ambiguities about the concept of parts in addition to cases in which objects of completely different nature (e.g. door and hand, and house cat) could be considered as part-whole relationships. However, Mereology is divided into several branches to handle different types of situations that the part-whole relations bring to light [7]. Thus, the three basic principles of Ground Mereology (reflection, antisymmetry and transitivity) can be combined to form other axioms on the part-whole relationship, as described in [7], which, in turn, can be combined and added to Ground Mereology forming, therefore, ramifications of research. The problem of transitivity mereological theories is treated in detail in [2], since already several counterexamples are found in the literature and discussions about cases where transitivity is not achieved through cognition or linguistically. To illustrate this, the following examples are given: the hand is part of the person which is part of a research group, but the hand is not part of the research group, Rio de Janeiro is part of Brazil which is part of the United Nations, but in this case, Rio de Janeiro is not part of the United Nations [2]. Thus, [2] proposes a theory and a formal typology of part-whole relations based on linguistics and cognition. (Meronymy) for the issue of transitivity of part-whole relations to be inserted in the formal theory (Mereology). As a result of this research,[10] obtained four distinct types ontological part-whole relationships, namely: (i) subquantity-quantify (e.g., alcohol-wine) - parts are modelled as a quantity of matter that are joined by a topological relationship; (ii) member-collective (eg, tree−forest) - the entity representing the whole is modeled with parts that play the same role in relation to the whole; (iii) subcollective-collective: modelling a relation between a collective and the subcollectives that offer extra structure to the prior (e.g., the south part of the amazon forest-amazon forest); (iv) component-functional complex (e.g., engine - car) - modelling an entity in which all parts accomplish a different role in relation to the whole, thus, contributing to the functionality of the latter. Following the same line of thought, Keet [8] develops its research based on metrological and meronimic studies, the latter being based on the work of [2]. Taking into account the most relevant aspects of these approaches for the conceptual modelling of part-whole relationships [8] formulated a taxonomy of part-whole relations and developed a decision procedure in order to facilitate the task of building conceptual models applicable to any domain. The work of [8] makes up the main theoretical basis for the development of the present work.

2.1 The OntoCon technique and PrOntoCon procedure

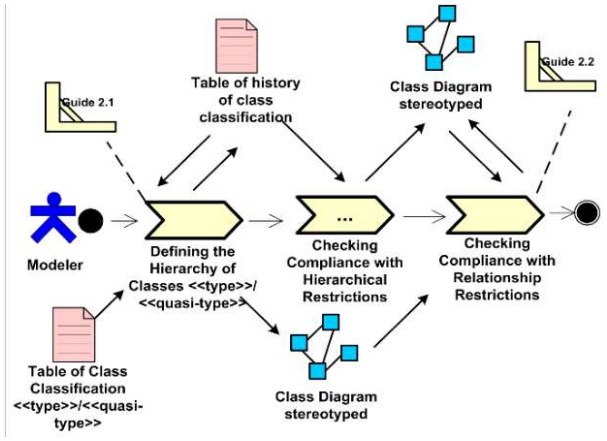
OntoCon has emerged from the combination of VERONTO's [1] and OntoUML Profile's [2] characteristics and it is also based over the meta-properties rigidity, unity and external dependence. Based on the arrangement of these meta-properties, Guarino and Welty [4] have proposed a formal ontological classification that, succinctly, establish a rigid property (+R) as essential for all its instances, that is, it will remain as long as the existence of the element that instantiates it (e.g., the class Person is said to be rigid because an instance of that class will always be a person throughout its existence). Otherwise, a non-rigid property (-R) is not essential to some of its instances and an anti-rigid property (R) is not essential to all its instances (e.g., the class Professor is said to be anti-rigid because all its instances are able to quit that condition at any moment of its life). For a deeper study about these meta-properties it is suggested the reading of [4]. Besides the already mentioned notation +R (-R e R), OntoCon still has other ones, like: the notation +I (-I) indicates that an element has (does not have) an identity condition; the notation +O (-O) determines if an element provides (does not provide) an identity condition; and the notation +D (-D) is used to indicate if an element has (does not have) an external dependency. Through the application of OntoCon technique, three key points can be validated in class diagrams: (i) generalization/specialization relationships; (ii) relations of classes involved in role modeling; and (iii) appropriate definition of UML constructors (concrete class, abstract class and interface) [11]. The PrOntoCon procedure levels organization had been based on these three points, and it generated four defined phases: (i) stereotype identification; (ii) hierarchy checking; (iii) application of the analysis pattern; and (iv) UML constructors checking. For the phases' formalization, SPEM (Software Process Engineering Metamodel) [12] has been adopted for a better activity diagrams achievement of each step that will guide the modeler during the validation of domain class diagrams. Fig. 1 shows the SPEM diagrams of phases 1 (a) and 2 (b), respectively. The PrOntoCon phase 1 goal is to conduct the modeler in order to identify which OntoCon stereotype corresponds to the existing classes. Therefore, the modeler will explore a decision tree containing questions to help him on that task, as well as several examples and counterexamples to reduce the doubts and misunderstandings that domain analysis can yield. However, on that PrOntoCon phase it is not possible to distinguish the stereotypes «type» and «quasi-type», so it creates classes labelled as «type»/«quasi-type». The

distinction will be made in the second phase with the task that leads to the specific stereotype. The following phase, named Hierarchy Checking, aims to do the distinction between «type»/«quasi-type» classes and to verify if there are or there should be generalization/specialization relationships among them obeying super/subclasses permissions defined in OntoCon. Depending on the modelling mistake found during that process, it can redirect the modeler to the third phase or conduct him to redo the stereotypes classification in order to find the error cause .

Figure 1: (a) Phase 1 Diagram: Stereotypes Identification. (b) Phase 2 Diagram: Hierarchy Checking (extracted from [5])



(a) Phase 1



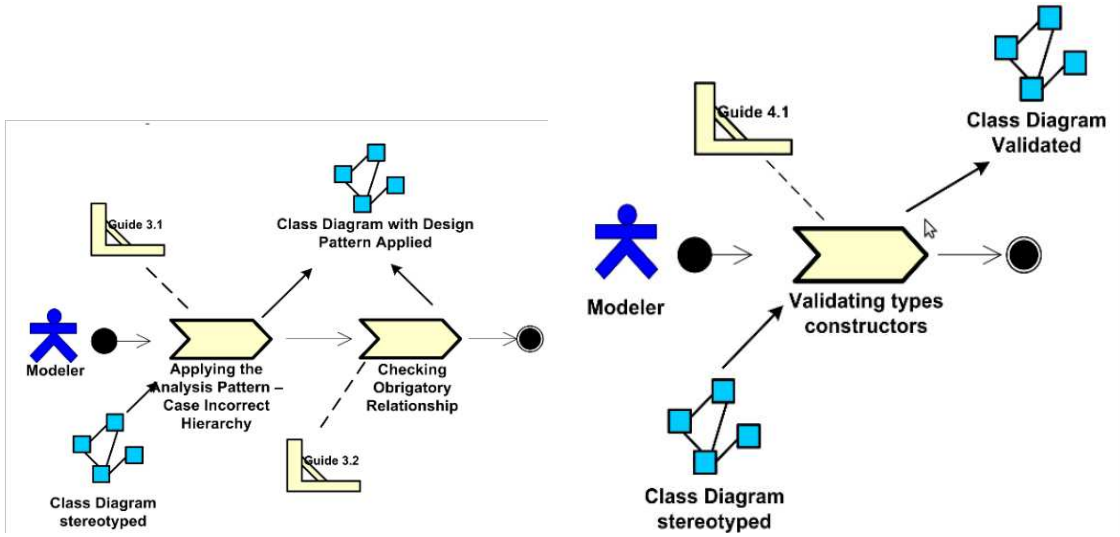
(b) Phase 2

Fig. 2 shows the SPEM diagrams of phases 3 (a) and 4 (b), respectively. The third phase of PrOntoCon addresses two problems: first of all is with respect to «type» classes being subclasses of « material role» classes; and the second is w.r.t. a «material role» class being subclass of more than one «type» class. The fourth and final procedure phase is responsible for the mapping of UML constructors to be used by modelers to finish the domain conceptual model construction according to the stereotypes in which the classes were classified. Thus, classes stereotyped as «type», «quasi-type», «material role» or «phased sortal» should be mapped as abstract or concrete classes; whereas classes stereotyped as «category» or «formal role» become interfaces or abstract classes.

3 Extending PrOntoCon procedure

Up to this point, PrOntoCon procedure can help developers in the task of modelling the generalization/ specialization relationships. Nevertheless, it does not allow that they advance to the part-whole or simple association relations analysis. There are books that merely suggest the avoidance of part-whole relations or they superficially

Figure 2: (a) Phase 3 Diagram: Application of the Analysis Pattern. (b) Phase 4 Diagram: UML Constructors Checking (extracted from [5])



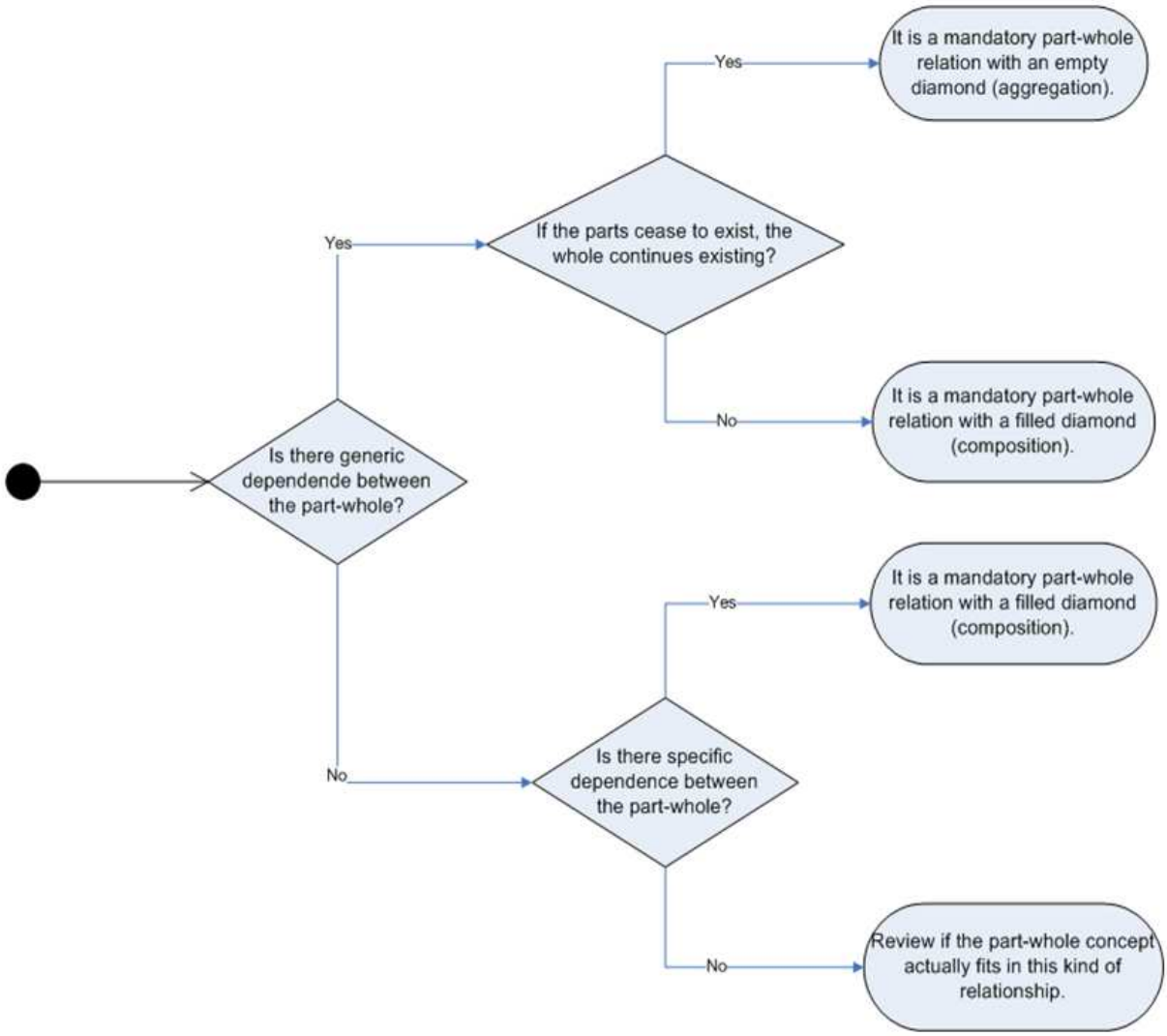
(a) Phase 3 (b) Phase 4

explain how to use them [6]. This approach is not a valuable support when that information is necessary to carry out a correct modelling expressed by a conceptual diagram. Even among published articles it is possible to see that this is a subject treated in just a few papers. Thus, many modelers "invent" their own UML extensions in order in order to represent such relationships, which can cause problems of understanding of the model generated. For this reason, the necessity of PrOntoCon extension for part-whole and association relations has arisen. Thus the procedure becomes more powerful and able to meet the modelling needs of a larger number of developers. The additions made by this research were based on the theoretical principles outlined in the work of [8, 13]. They have as main principle that part-whole relations have both mereological and meronymic concepts, which forms, a taxonomy based on those two characteristics including the kinds of relationships that are more relevant to the conceptual modelling of part-whole relations. Nonetheless, the decision diagram created by [8] is aimed to help create models in the ORM language. However, analyzing the principles underlying the work of Keet is possible to see that it could be adapted to this research's goal. This adaptation resulted in the decision diagram of Fig. 3. From this diagram, the Extended PrOntoCon procedure steps could be constructed with SPEM, in order to maintain the syntax of the existing method. Fig. 4 shows the new PrOntoCon step that possesses two distinct activities: (i) part-whole verification; and (ii) association verification.

At the moment the modeler has access to that procedure phase, it indicates that part-whole verification activity should be executed before associations verification activity. Then he/she receives the first orientations contained in Guide 1.1 (as indicated in Fig. 4). In general, the partially validated class diagram refers to the final diagram obtained through PrOntoCon application, in which only relationships of hierarchy were validated. For this reason, those relationships should not be analyzed again in this present activity. All the others must pass through part-whole relations verification, even those that are modelled as associations. The goal of this first activity is not only verify if the part-whole relations are correct, but also identify relationships that, at first, were not identified as part-whole relationship, but that could be classified as such. To start the activity, the modeler must click on "Tree...", as illustrated by Fig. 4, in order to have access to a decision diagram containing questions and examples. Those questions and examples are necessary to identify if the relation being examined will fit in some dependence case, characterizing, in this way, a part-whole relation. Fig. 5 presents the decision diagram encountered by the modeler when he/she executes that action.

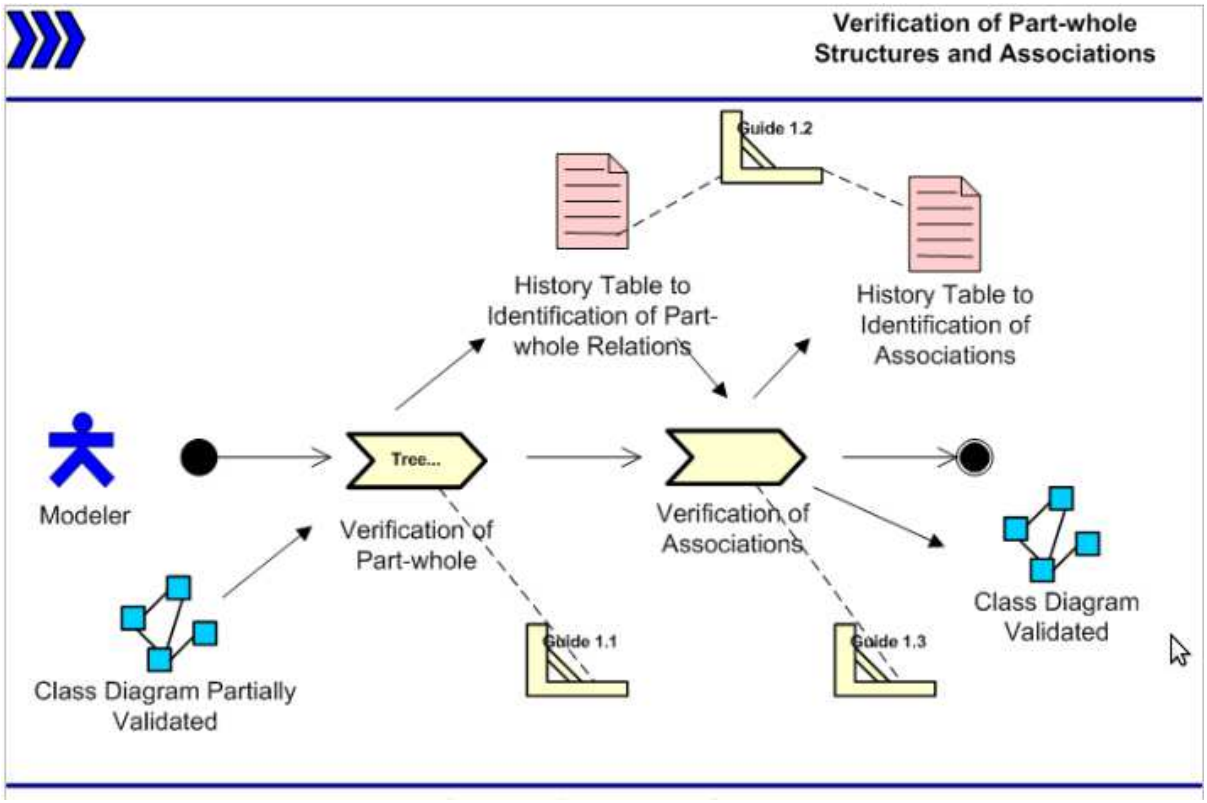
The first thing the decision diagram tries to identify is that there is generic dependence between the parts, that is, if both the *whole* as the *part* can have independent existences. If the answer is affirmative, then it is asked

Figure 3: Decision Diagram simplified for UML use



if there is the whole existence without its parts. Depending on the answer (yes or no), the relation can be an aggregation or a composition. On the other hand, if there is not generic dependence between the part and the whole, a question about specific dependence is examined, i.e., the parts cannot be substituted without modifying or destroying the whole. If, in this case, the answer is affirmative, the relation is a composition. Otherwise, the modeler is guided to the next activity to verify associations. At that time, when the modeler had identified the part-whole relation kind of the domain classes being analyzed, this is the moment to verify if the use of composite structures (an important addition to UML 2.0) is possible for the relationships classified as composition [14]. Thus, an additional question was added to those relations that already could be modelled as composition using the common UML notation, asking if the parts interact with each other. If the answer is negative, the model should not be altered; and, even if the answer be affirmative, i.e. the use of composite structures is available to turn modelling task easier, the final decision will always be the modeler's one. When the relations being analyzed by the modeler do not fit in any kind of dependence between the part and the whole, the Extended PrOntoCon leads the modeler to perform the second activity of that new phase, which is the association verification. To start that activity, the modeler should click on Guide 1.3 (as shown in Fig. 4) to have access to commoner association categories that are possible, presented in Table 1. Each one has enlightening examples to help the modeler at that identification, in addition to question supports, trying to follow the same pattern of the previous activity.

Figure 4: Activity Diagram of the new phase to verify part-whole and associations structures

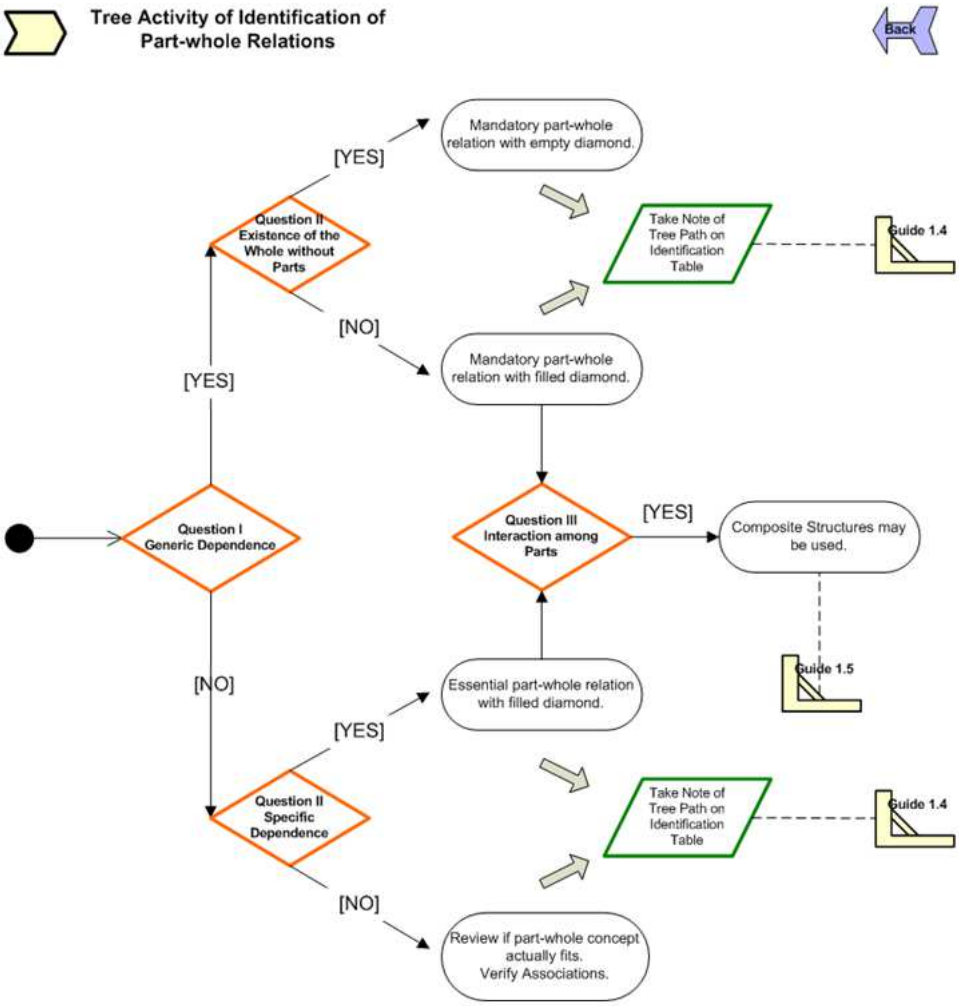


The methodological support for the identification of those association categories was based on Larman's book [15], since it gives a detailed explanation of the commonest associations that can be found during a domain modelling. It Also, provides a check-list, separated by association categories, in order to help the modeler to distinguish those class associations that are relevant from those that are not, and can be omitted from the model. However,based on studies conducted in this research about part-whole relations, it was found that several categories that were being suggested to be modelled as associations fitted, actually, in part-whole structures. That fact was used to reassert the idea that the identification of part-whole relations should precede the identification of associations. So, the categories of possible associations that have fitted in part-whole structures were removed from the associations' identification, remaining only the ones shown in Table 1(the decision diagram defined in Fig. 5 was applied on that task). If at the end of the verification of those categories there were still classes without relationships in the model, that is, isolated classes in the diagram, Extended PrOntoCon notifies the modeler that this phase must be redone. And, even if mistakes are not found, then it will be necessary to redo PrOntoCon previous phases (those related to validation of generalization/specialization relationships). Nonetheless, if the validation of all classes and relations is done in a satisfactory manner, that is, without any "leftover", the final class diagram is considered validated for both part-whole and associations and generalization/specialization relationships.

3.1 Related work

We already have discussed the works of [8, 13] that served as theoretical basis for this article. However, these works were not aimed to help in the validation of UML class diagrams. The work of [16] focused on the analysis of algorithms for identification of correctness problems that are caused by aggregation/composition constraints. But this work focused on checking the cardinality constraints of relationships and has not addressed the issue of verifying the occurrence of the relationship. In [17] was presented a code generation process that systematically obtains the implementation of the UML association, aggregation and composition concepts. But in this case the relationships should be modelled correctly previously.

Figure 5: Identification tree of part-whole relations



4 Case study

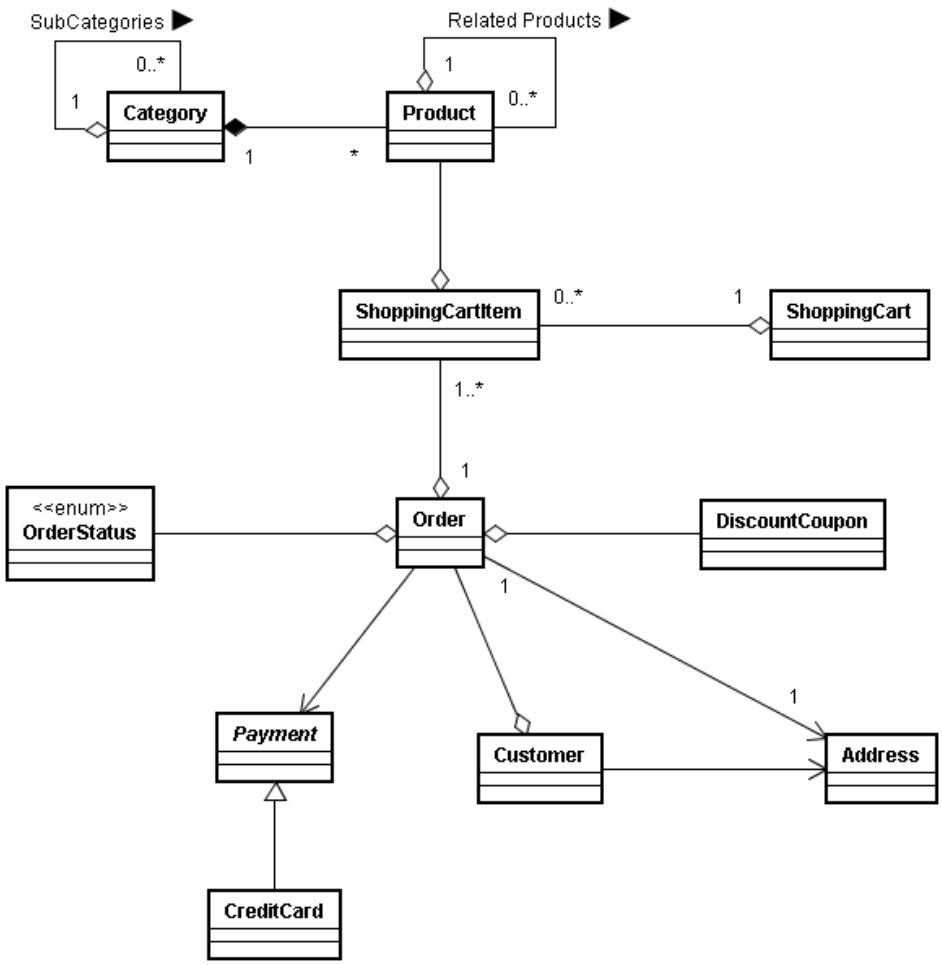
The purpose of this section is to demonstrate the use of the Extended PrOntoCon procedure. The domain chosen as an example is a domain of e-commerce. Fig. 6 presents a simplified model of an e-commerce domain, in which "... one category can have none or many subcategories. Categories are composed of products that can have many related products. A shopping cart item has a product. [...] An order possesses one or many shopping cart items, and it can have a discount coupon, a delivery address, payment information, a status referring to the actual condition of the order and it always belongs to a customer" [18].

That class diagram served as a starting point for applying the PrOntoCon procedure along with Extended PrOntoCon. First, with the application of PrOntoCon developed by [11], it has identified that Customer should have a superclass (for classes stereotyped as «material role» must exist a superclass, immediate or no, classified as «type») and that the hierarchy between Payment and CreditCard was correct. Since the required superclass of the Customer class did not exist in the model described in Fig. 6, it was necessary its creation, represented by the Person class. With those modifications done, it was possible to proceed the procedure using the Extended PrOntoCon. As shown in Section 3, the first activity of Extended PrOntoCon is the verification of aggregation/composition that is constituted of a decision tree that identifies the part-whole structures. Thus, each classes' pair was analyzed according to the questionnaire contained in that diagram.

Table 1: Definition of relevant categories for identification of associations.

Category	*Examples*
A is physically contained in B	Register-Store, Item-Shelf, Passenger-Airplane
A uses or manages B	Cashier-Register, Pilot-Airplane
A communicates with B	Customer-Cashier, BookingAgent-Passenger
A is related to a transaction B	Customer-Payment, Passenger-Ticket
A is a transaction related to another transaction B	Payment-Sale, Reservation-Cancellation
A is adjacent to B	Product-Product, City-City

Figure 6: Simplified original representation of an e-commerce domain (extracted and adapted from [18])

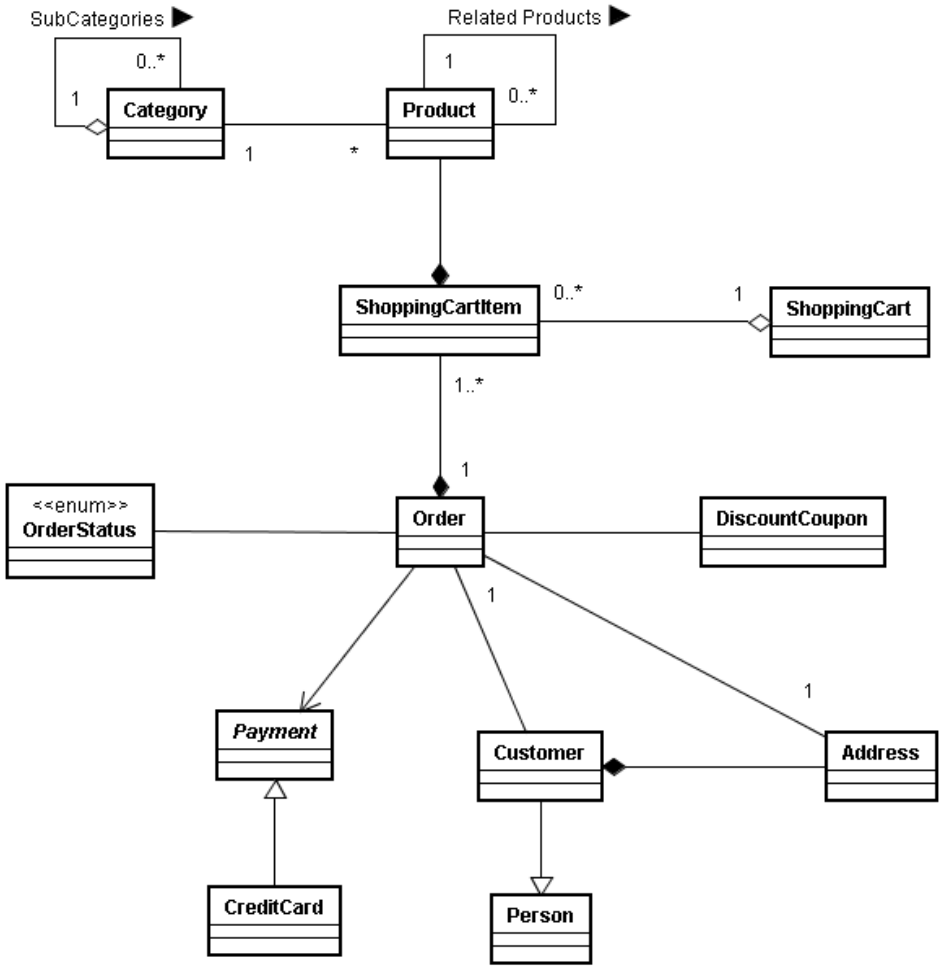


Considering Customer class as the whole and Order class as the part, the first question tries to identify if there is a generic dependence between those classes, that is, if the whole must have a part that can be substituted over time. The answer given was “no” for Question I – Generic Dependence, because if it is taken into account that the client register must stay stored, then the customer continues being a client even without doing an order. This negative answer conducts to Question II – Specific Dependence, which is also answered “no”, because the replacement of the orders does not modify or destroy the client. The result of that activity was to review if the part-whole concept applies, but before that it is necessary to pass to the next activity to verify associations. For the classes Order as the whole and ShoppingCartItem as the part, the answer to Question I – Generic Dependence was

"yes", since the items of an order can be replaced. That answer has conducted to Question II – Existence of the Whole without Parts, which was answered no, because there is no sense in an existing order without items. The result of that activity was the construction of a composition. For the classes Customer as the whole and Address as the part, the answer to Question I – Generic Dependence was "yes", since a client can have more than one delivery address registered and he/she can substitute them anytime. That response has guided to Question II – Existence of the Whole without Parts which was answered "no", because there is no sense when we think about a client doing an order without a destination to his delivery. The end product to that activity was a creation of a composition. And, thus, for all the other classes the same diagram shown in Fig. 5 was applied. However, due to lack of space, the details of all analysis will not be presented. In this domain example, the use of composite structures for composition relationships was not identified, since it was not verified any case in which parts of different classes' instances interacted with each other to the same whole.

Thus, at the end of the application of the first Extended PrOntoCon activity, it is possible to proceed to the second activity in order to verify if the relationships between those classes that have not fitted as part-whole structures. They are simple association relationships. As previously identified, the classes Customer and Order should be verified to confirm if a simple association would describe that relation better. Therefore, sequentially analyzing the categories (listed in Table 1), it has identified that the classes Customer and Order fitted in case "A is related to a transaction B", that is, a client is related to an order that is a transaction. Thus, the relationship between those classes will be, in fact, an association. This procedure must be repeated for all the other classes that were not framed as part-whole relations in the first activity of Extended PrOntoCon. At this point, in which all the classes' pairs were examined for both part-whole and associations relationships and there is not any isolated class in the diagram, the application of Extended PrOntoCon comes to an end, providing the class diagram presented in Fig. 6.

Figure 7: Class diagram after the application of Extended PrOntoCon



The changes have occurred in the relationships between the classes Product, ShoppingCartItem, Order, OrderStatus, DiscountCoupon, Customer and Address. The improvements that can be evidenced with this new modeling are referred to: (a) a closer approach with the reality of the domain studied, since concepts of that domain could be deeper explored and, thus, clearly comprehended, besides elucidating restrictions contained in domain w.r.t. the linking of life times of instances of those classes; thus, it is possible to verify in this example the modification of part-whole relations between the classes Product, Customer, DiscountCoupon and OrderStatus to simple association enlightens the independence of their existences, that is, the life time of instances of these classes are not linked (e.g., a discount coupon can be created before the order, as well as it can be valid after the order has been concluded, if the coupon was not used); (b) indicate where the creation-destruction dependencies of the part in relation to the whole occur (and vice-versa), what will have impact, in terms of referential integrity and cascading delete paths, in the connection among software elements (classes) and databases elements that represent the whole and the parts, e.g., the composition identified between the relations of classes Order, ShoppingCartItem and Product indicates, precisely, the creation-destruction dependencies among them, meaning that if a product is deleted, its respective shopping cart item should be too; (c) facilitate the addition of new roles in the future, besides Customer, with the creation of the Person class. Thus, it is possible to realize that with those benefits reached, those domain systems will be easier to maintain (with the referential integrity improved), more scalable (since the concepts of that domain were more clearly comprehended) and robust, what will provide the facility of domain expansion as well as its integration with others domains.

5 Conclusions

The difficulty software developers encounter to construct conceptual models that are close to the real world is still visible nowadays. Modelers often fail to express the concepts they capture in a clear and correct manner, culminating in several problems during software development and even after product deliver. This certainly will cause extra expense to correct the software and possible cost overrun. This is an uncomfortable and undesired situation for both developer and clients, which can shake the confidence of the latter in the former. Thus, the creation of methods and procedures that have as main goal join practice with theory tends to decrease more and more those inconveniences. In the specific case of this work, related to the early stages of software development, the complexity of ontological domain analysis was broken so that any modeler of UML class diagrams could obtain the benefits of using a formal procedure to identify part-whole and association's relationships that usually generate many doubts during modelling. We try to keep the procedure as simple as possible, but without diverting the underlying ontological principles. Therefore, it is expected that the effective use of PrOntoCon produce class diagrams more clear and faithful, avoiding, in turn, the various disorders caused by a misunderstood domain analysis. Field testing may confirm the usability of the procedure.

Acknowledgment

We would like to thank the funding agencies FAPEMIG and CNPq for the financial support for this project.

References

[1] VILLELA, M. L. B.; OLIVEIRA, A.; BRAGA, J. L. Modelagem ontológica no apoio à modelagem conceitual. *Simpósio brasileiro de engenharia de software*, v. 18, 2004.

[2] GUIZZARDI, G. *Ontological foundations for structural conceptual models*. [S.l.]: CTIT, Centre for Telematics and Information Technology, 2005.

[3] GUARINO, N.; WELTY, C. A formal ontology of properties. In: *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*. [S.l.]: Springer, 2000. p. 97–112.

[4] GUARINO, N.; WELTY, C. Towards a methodology for ontology-based model engineering. In: *Proceedings of the ECOOP-2000 Workshop on Model Engineering*. [S.l.: s.n.], 2000.

[5] TAVARES, D. B. et al. Analysis procedure for validation of domain class diagrams based on ontological analysis. In: *Advances in Conceptual Modeling-Challenging Perspectives*. [S.l.]: Springer, 2009. p. 159–168.

[6] FOWLER, M. *UML distilled: a brief guide to the standard object modeling language*. [S.l.]: Addison-Wesley Professional, 2004.

[7] VARZI, A. Mereology. In: ZALTA, E. N. (Ed.). *The Stanford Encyclopedia of Philosophy*. Winter 2012. [S.l.: s.n.], 2012.

[8] KEET, C. M. Part-whole relations in object-role models. In: SPRINGER. *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*. [S.l.], 2006. p. 1118–1127.

[9] WAND, Y.; STOREY, V. C.; WEBER, R. An ontological analysis of the relationship construct in conceptual modeling. *ACM Transactions on Database Systems (TODS)*, ACM, v. 24, n. 4, p. 494–528, 1999.

[10] GUIZZARDI, G. Ontological foundations for conceptual part-whole relations: the case of collectives and their parts. In: SPRINGER. *Advanced Information Systems Engineering*. [S.l.], 2011. p. 138–153.

[11] TAVARES, D. *Procedimento de análise para validação de diagrama de classes de domínio baseado em análise ontológica*. Dissertação (MSc thesis) — Universidade Federal de Viçosa, 2008.

[12] OMG. *Software and systems process engineering meta-model specification*. [S.l.], 2008.

[13] KEET, C. M.; ARTALE, A. Representing and reasoning over a taxonomy of part–whole relations. *Applied Ontology*, IOS Press, v. 3, n. 1, p. 91–110, 2008.

[14] BOCK, C. Uml 2 composition model. *Journal of Object Technology*, v. 3, n. 10, p. 47–74, 2004.

[15] LARMAN, C. *Applying UML and Patters: An introduction to Object-oriented analysis and design and the Unified Process*. [S.l.]: Prentice Hall, ISBN, 2002.

[16] BALABAN, M.; MARAEE, A. Simplification and correctness of uml class diagrams–focusing on multiplicity and aggregation/composition constraints. In: *Model-Driven Engineering Languages and Systems*. [S.l.]: Springer, 2013. p. 454–470.

[17] ALBERT, M. et al. Implementing uml association, aggregation, and composition. a particular interpretation based on a multidimensional framework. In: SPRINGER. *Advanced Information Systems Engineering*. [S.l.], 2003. p. 143–158.

[18] PERíLIO, R. Domain model: Uma forma mais eficiente de construir aplicações enterprise. *MundoJ*, v. 7, n. 42, 2010.