Requirements Eng (2014) 19:63–80
DOI 10.1007/s00766-012-0159-y

ORIGINAL ARTICLE

Guidelines for using UML association classes and their effect on domain understanding in requirements engineering

Palash Bera · Joerg Evermann

Received: 4 June 2011 / Accepted: 7 July 2012 / Published online: 26 July 2012

Abstract The analysis and description of the application domain are important parts of the requirements engineering process. Domain descriptions are frequently represented as models in the de-facto standard unified modeling language (UML). Recent research has specified the semantics of various UML language elements for domain modeling, based on ontological considerations. In this paper, we empirically examine ontological modeling guidelines for the UML association construct, which plays a central role in UML class diagrams. Using an experimental study, we find that some, but not all, of the proposed guidelines lead to better application domain models. We use a process-tracing study to investigate in more detail the effects of ontological guidelines. The combined results indicate that ontological guidelines can improve the usefulness of UML class diagrams for describing the application domain, and thus have the potential to improve downstream system development activities and ultimately affect the successful information systems implementation.

Keywords UML association class · Conceptual model · Domain understanding

P. Bera (✉)
Saint Louis University, St. Louis, MO, USA
e-mail: pbera@slu.edu

J. Evermann
Memorial University of Newfoundland, St. John's, NL, Canada
e-mail: jevermann@mun.ca

1 Introduction

In the requirements engineering phase of information systems (IS) development, conceptual modeling is a key activity performed to describe the business and organizational domain. Models termed *conceptual models* or *domain models* represent the application domains and are created for the purpose of analyzing, understanding, and communicating about the application domain and are input to the requirements specification phase for IS development [1]. Conceptual models are developed independent of software implementation considerations.

To remove the need for error-prone translations between domain description languages and software design languages such as unified modeling language (UML), the use of the same language or notation (e.g., UML) for both types of models, that is, domain models as well as software models, has been suggested [2].

Using a software description language also for describing the application domain may conceivably introduce some "design bias," that is, a usage of the language constructs that is appropriate for software models, but less so or entirely inappropriate for domain descriptions. To prevent such bias, explicit modeling rules and guidelines are necessary to ensure domain-appropriate use of the language, rather than software-focused use. Previous work has presented such guidelines for UML (e.g., [2]), based on theoretical arguments. The present paper evaluates some of these UML modeling guidelines in the context of domain modeling.

UML class diagrams describe the static structure of an object-oriented (OO) model and are considered to be the most popular UML diagrams used in practice [3]. UML use in IS development has increased significantly and is now the second most widely used technique, after entity-relationship diagrams [4]. Dobing and Parsons [3, 5] conducted surveys on the use of UML by IS practitioners. They found that more than 70 % of their respondents used UML diagrams for communication and shared understanding of the system design among the development team members, and more than 50 % of their respondents reported UML diagrams to be useful during early phases of systems development to communicate and verify IS requirements with clients or users. Their studies demonstrate the importance and usefulness of UML, and especially UML class diagrams, to the practice of IS development and conceptual modeling. Fettke [4] and Davies et al. [6] found similar results in terms of use of UML models in practice. Improvements in the ability of conceptual modeling grammars to express domain semantics therefore have the potential to improve the degree to which IS developers meet client requirements [7].

In order to use a software design language to represent application domain concepts, the semantics of the language constructs with respect to the application domain elements must be specified. The main problem of using OO IS design languages for conceptual modeling is the lack of meaning of language constructs such as "attribute" and "operation" when used for modeling application domains [8]. As the UML is the de-facto standard for software modeling, its language elements have received considerable attention with respect to their domain semantics [3]. A construct that has received much attention in particular is the association [8–11]. The association construct in UML expresses any kind of semantic relationship between two classes of objects and is therefore extensively used [9]. A UML association class is a construct that extends the association with the ability to attach structural and behavioral features (typically attributes and operations). Constructs such as "association" are reasonably well mapped to programming code, as they are more or less directly mapped to programming language constructs and therefore have clear software semantics. However, when examining for example a business domain, for example, the accounts payable process, it is much less clear what exists in such a domain, and therefore what should or should not be represented by an association or association class (and its attributes and operations). Consequently, our proposal is built on an ontology, a set of assumptions as to what exists in a domain.

As UML class diagrams can be used as conceptual models to represent application domains, this paper focuses on the effects of using attributes and operations in UML association classes on the application domain understanding that is conveyed through UML class diagrams. The use of attributes, and to a lesser extent operations, in association classes is common in the literature and in software development practice. However, the effect of using association class attributes and operations on domain model understanding has not yet been investigated, despite the prominence of the association construct in models. Thus, through two empirical studies, this paper investigates the effect of using attributes and operations in association classes on users' domain understanding. The results of this paper contribute to our understanding of the effects of how modeling languages convey domain understanding. Improvements in domain understanding are beneficial to the subsequent phases of the IS development process and may ultimately contribute in successful IS development.

We emphasize that this paper does not propose a new language or a new software tool which would, in the face of accepted de-facto standards like UML and mature tools like Eclipse, have little impact on practice. Instead, we focus on the usage of the language elements themselves. The modeling guidelines we examine in this paper show how models might be constructed to improve domain understanding. These guidelines are not extensions or adaptations of UML and do not require special tool support. As such, they are immediately applicable by IS developers.

The remainder of the paper is structured as follows. Section 2 presents the background on UML association classes. The two empirical studies are presented in Sect. 3. Section 4 is the discussion and conclusion section.

2 Background

2.1 Association class

An association represents the relationships between classes [12, p. 26]. The OMG definition [13] adds that a UML association class not only connects a set of classes but also defines a set of features that belong to the relationship itself and not to any of the associated classes. However, some aspects of the association construct are unclear and confusing even when it is used in software design [14] and definitions and use of associations in the literature are imprecise. For example, the official UML specification suggests that "an association defines a semantic relationship between classifiers" [13, p. 36], but does not follow through on defining what a "semantic relationship" is. Embley [15] notes that relationships associate one object with another. Taken together, these two form circular definitions. Similarly, Siegfried [16] points out that an association sets up a connection, which is some type of fact that we want to note in our model. No further clarification or definition of the notion of connection and when it might be noteworthy is provided.

The use of association classes and their attributes is also discussed only in general terms. For example, an association class may be viewed as an association that also has class properties [13]. This provides a syntactic definition, but falls short of defining the deeper meaning of such an association class. Liu et al. [17] mention that UML allows association classes to represent associations that have data properties, also focusing on the syntax rather than the meaning of this construct.

Based on the intuitive but imprecise definitions and illustrations in the literature, Evermann [8] and Evermann and Wand [18] have specified the meaning of association classes with respect to the use of UML for modeling of application domains. They argue that UML associations have two distinct purposes. The first is to indicate communication or interaction between different things, while the second is to represent mutual properties of objects that are relevant to a domain model. While these definitions appear as imprecise as other illustrations, the use of an underlying ontology [13] specifies precisely what things, mutual properties, and interactions are. Assigning a clear meaning to language constructs w.r.t. the domain elements they are intended to describe can prevent the construction of models that are illogical or internally inconsistent w.r.t. to the domain ontology. Such inconsistent or illogical models are more ambiguous or are difficult to interpret. In contrast, clear meaning can lead to modeling rules that ensure that only logical and internally consistent models w.r.t. the domain ontology can be constructed. Evermann and Wand [18] used widely used upper-level ontology, developed by Bunge [19], to define the semantics of associations and association classes with respect to application domains and derived a set of guidelines for modeling association classes. These guidelines were developed by translating the ontological premises into UML language constraints [2]. The ontological premises and the guidelines are described next.

2.2 Guidelines for modeling association classes

This section briefly introduces the ontology that is assumed for application domains and the modeling guidelines developed by the ontological assumptions by Evermann and Wand [18]. The main premise of Bunge's ontology [19] is that the world is made up of substantial *things* that are assumed to really exist. A thing possesses *properties* which can be either *intrinsic* or *mutual*. Intrinsic properties are those that a thing possesses by itself, such as "height," whereas mutual properties exist between two or more things, for example, "employed by" exists between a person and an organization. A change of a mutual property in one thing affects the thing or things that share this property. Interactions are described in terms of changes to mutual properties [18]. From these, ontological premises follow the guidelines in Table 1 that apply to the use of UML association classes when describing application domains, but do not apply to software models.

This paper focuses on the first three guidelines. Guidelines 4–6 prohibit particular uses of association classes that we believe are rarely if ever seen in practice. For example, while an association class has two or more member ends that are properties, which are owned attributes of the associated classes, it could in principle also own attributes which are properties that are member ends of an another association or association class. Guideline 4 prohibits this, though we believe that such a model is sufficiently unusual so as not to warrant specific attention. Following are examples of the violations of the first three ontological guidelines for association classes.

Consider the following example where the relationship between Customer and Staff of a restaurant is modeled using an association class termed FoodOrder. As per guideline 1, the mutual properties that arise due to the interaction between Customer and Staff (e.g., OrderSequence, OrderTime, and TotalAmount) are modeled as attributes of the association class (Fig. 1).

If one ignores these attributes as mutual properties, then this association class can be considered as a regular class and connected to other classes (as shown in Fig. 2). Such conversion of association classes to regular classes can be termed as objectification of classes and is syntactically correct and thus Fig. 2 is a valid UML class diagram. However, Fig. 2 represents different application domain semantics. Here, FoodOrder is supposed to represent a really existing thing (by virtue of it being a class); yet, it is clear that in the application domain, a food order is an event, not a substantial thing—it reflects interaction between two substantial things—Staff and Customer. We note that a physical record of such an event, for example, an order slip, is not the same as that event.

Now consider the violation of guideline 2, which suggests that operations should not be used in association classes. This can be violated by manipulating Fig. 1 by placing operations from the classes Customer and Staff in the FoodOrder association class. The resultant model is shown in Fig. 3. Again, this is syntactically correct, but in this case, it leads to unclear domain semantics, as FoodOrder represents a set of mutual properties (by virtue of it being an association class) as well as a really existing thing (by virtue of it having behavior).

Now consider the violation of guideline 3, which states that association classes must possess at least one attribute. To create such a violation, the attributes of association classes can be distributed to other classes. As it is unusual in practice to create association classes with neither structural nor behavioral features, one way of violating guideline 3 is to modify Fig. 3 (where operations are included) by removing the attributes of the association class and distributing them to the other classes, so that only the

Table 1 Ontological guidelines to model UML association classes (based on [18])

No.	Ontological guidelines	Explanation
1.	Mutual properties must be represented as attributes of association classes	In UML, attributes of association classes instantiate to slots owned by links between two or more objects. The ontological notion of mutual property is closely related to the idea of slots of links between objects, in that their values characterize a connection between ontological things. The set of association class attributes therefore represents a set of mutual properties. As noted in [12], the association class itself is simply a containing construct without separate ontological meaning
2.	An association class must not possess methods or operations	Since association classes with their attributes do not represent classes of substantial things but sets of mutual properties, they cannot possess methods or operations. Ontologically, only things, not their properties, can engage in action and interaction
3.	An association class must possess at least one attribute	An association class without attributes would represent an empty set of mutual properties. Such a class is ontologically meaningless
4.	An association class must not be associated with another class	As properties in ontology cannot themselves possess mutual properties, an association class, whose attributes represent such mutual properties, must not be associated with any other association class. To use precise UML terminology, this rule specifies that an association class must not own properties, which are member ends of other associations
5.	An association class must not participate in generalization relationships	While properties and associations can be generalized and specialized in UML, this is not possible ontologically. In ontology, mutual properties cannot be generalized and as a consequence, association classes with their attributes representing sets of mutual properties must not be generalized
6.	An association class represents a set of mutual properties arising out of the same interaction between class instances. Sets of mutual properties arising out of different interactions must be modeled using different association classes	Prior work [12] has suggested that mutual properties between things are the result of interactions between those things. We term sets of mutual properties that result from the same interaction event as concurrent mutual properties. Each association class with its attributes expresses a set of concurrent properties. Thus, different association classes and their attributes should be used when mutual properties are not concurrent

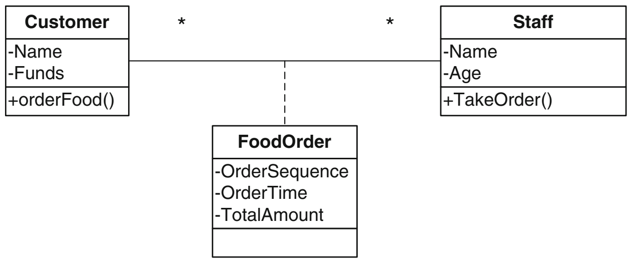


Fig. 1 Example of an association class developed as per guideline 1

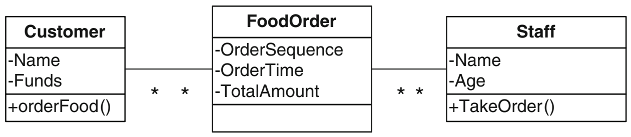


Fig. 2 Example of violation of guideline 1

operations remain in the association class. For example, the attributes of FoodOrder (Fig. 3) can be distributed to the Customer and Staff classes. This manipulation will result in

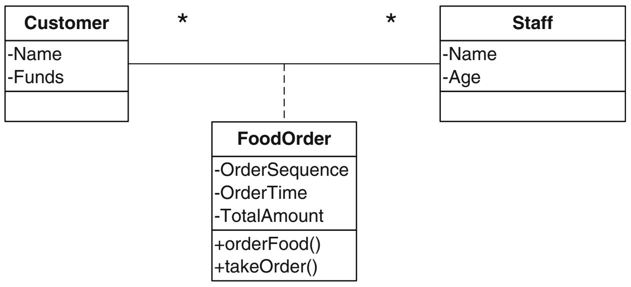


Fig. 3 Example of violation of guideline 2

Fig. 4. Note that by violating guideline 3 in this way, guideline 2 is also violated as the association class now contains operations. However, we discuss the violation of the guidelines separately since guideline 2 can be violated independently of guideline 3 and guideline 3 can in principle be violated without violating guideline 2. Again, Fig. 4 is a syntactically valid UML class diagram, yet its domain interpretation is problematic, FoodOrder represents an empty bundle of mutual properties.

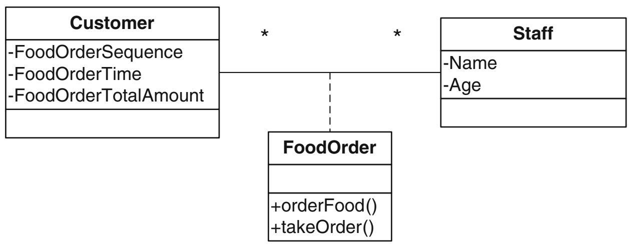


Fig. 4 Example of violation of guideline 3

2.3 Empirical studies on UML association classes

To the best of our knowledge, only three empirical studies [20–22] have looked at the use of attributes in association classes. The findings of these three studies are summarized in Table 2.

Table 2 indicates that in all three studies, the use of ontological guidelines had a clear effect on domain understanding. However, the impact of different ways of modeling associations and association classes cannot be clearly identified because of confounding and other limitations mentioned in the table. In particular, the effects of violating guidelines 1–3 are uncertain from the procedure and results presented in [14, 15]. Moreover, none of these studies focused on the modeling of operations in association classes. Thus, to have a clear understanding of the effect of modeling attributes and operations in association classes, two empirical studies were conducted, described next. In contrast to the previous studies, the two studies presented in this paper isolate the effect of violating individual guidelines 1–3 from potential confounds and improves on the accuracy and validity of the previous results.

3 Empirical studies

Our study is concerned with ensuring that the principles proposed in the form of three modeling guidelines are sound, that is, have a beneficial impact on domain understanding. In other words, we focus on the internal validity of our proposal to make sure it works [23]. We believe it is necessary to demonstrate that our principles are sound before taking the next step and applying them in practice settings in the form of case studies. Case studies focus on the external validity and are intended to show that something is applicable and can be applied, whereas controlled laboratory experiments are best suited to assess the internal validity and are intended to show specific effects. Experimental work must necessarily be prior to assessing practical applicability.

Table 2 Summary of empirical studies related to UML association classes

Author	Results	Comments
Evermann and Wand [20]	This paper empirically compares class diagrams that are developed based on ontological guidelines. The diagrams that had no violations of the guidelines were found to be superior for problem-solving tasks compared to diagrams that had maximum violations	This paper did not examine the effect of violating individual guidelines. Although this paper uses the ontological violations related to association class (Table 1), it also uses violations related to composition and generalization constructs. Thus, the results cannot be attributed specifically to guidelines related to violations of association class In contrast, the present paper separates the association class guidelines and prevents such confounding of results
Evermann and Halimi [22]	This paper empirically compares two class diagrams, one that contained association classes developed with mutual properties (as per guideline 1, Table 1) and the other developed with associations instead of association classes by objectification (see Fig. 2 for an example of objectification). The results indicate that the problem-solving performance was significantly better when subjects used the diagram developed by following guideline 1 as compared to the use of diagram developed by violating guideline 1	This study tested the violation of guideline 1 as discussed in the paper. One limitation of this paper is its low statistical power which might be the result of using few subjects in the study (29 for two groups) In contrast, the present study improves on the sample size and therefore the results are more accurate and valid
Poels [21]	This study focuses on the effectiveness of the resource event and agent (REA)-based model for domain understanding. To find such effectiveness, REA-based models are compared with non-REA-based models. Non-REA-based models are developed by objectification of association classes. The results indicate that problem-solving performance was significantly better with REA-based models as compared to non-REA-based models	In this study, the effect of objectification of association classes cannot be isolated completely, because the non-REA diagrams were created by objectifications of association classes and all the classes were repositioned in the diagram to violate the REA pattern. In other words, the REA pattern and the treatment of association classes are confounded In contrast, the present study is not limited to modeling REA patterns and the guidelines examined here apply more broadly to a larger set of UML models

3.1 Study 1: experiment

3.1.1 Study 1 theory and hypotheses

We base our study on multimedia-learning (MML) theory [24] as a UML domain model may be viewed as a multimedia element, combining graphics and textual notation in a single representation. MML theory suggests that even small differences in the material (diagram) can have a significant impact on understanding and hence on performance. In particular, if concepts are reorganized to align with the task performed by users then it might have a beneficial effect on users' understanding of the material. In our case, the chosen task is developing an understanding of an application domain based on UML class diagrams that include association classes. This domain understanding is operationalized as the ability to solve problems within the domain. Answering problem-solving questions requires subjects to develop creative solutions by reasoning about issues that are not directly answerable by viewing the model [25]. In such case, subjects need to rely on their prior knowledge and integrate this knowledge with the information obtained from the model. High scores in problem-solving questions indicate that a deep level of understanding has occurred [26]. This is a more appropriate operationalization of learning than recall from short-term memory.

In Sect. 2, we discussed an example of an association class (Fig. 1) that is created by following the ontological guideline as per Table 1. Figure 1 does not violate any of the ontological guidelines mentioned in Table 1. In the first study, we tested the effectiveness of such diagrams for domain understanding by comparing their effect with that of diagrams that violate the guidelines 1–3 (such as Figs. 2, 3, and 4). For reasons outlined above, we chose to focus on guidelines 1–3 only. Table 3 illustrates how modeling association classes to comply with the ontological guidelines may lead to better learning performance than otherwise. In the Figures in Table 3, only the positions of the attributes and/or operations have changed. We hypothesize that such changes might cause difficulty to users (referring to Fig. 3B–3D) in interpreting the meaning of the association classes (such as FoodOrder). Therefore, the performance of the problem-solving tasks for these users might be affected, especially when the problem-solving questions focus on concepts represented by association classes.

On the basis of the theory and preceding arguments, we now present our hypotheses.

H1 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will obtain a better understanding of the domain as compared to individuals reading UML class diagrams where association classes contain *both* attributes and operations.

H2 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will obtain a better understanding of the domain as compared to individuals reading UML class diagrams where association classes contain *only* operations.

By objectifying association classes, the semantics of the association as a conceptual relationship is lost. This will impact the domain understanding of the users who receive objectified classes instead of association classes and thus we propose:

H3 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will obtain a better understanding of the domain as compared to individuals reading UML class diagrams where association classes are objectified.

Individuals might also be consciously aware that the different types of diagrams foster or hinder domain understanding and problem solving. This awareness might be reflected in subjects' assessment of how useful the diagrams were to the task of answering the problem-solving questions. Thus, we advance the following three hypotheses related to perceived usefulness of the diagrams in problem solving:

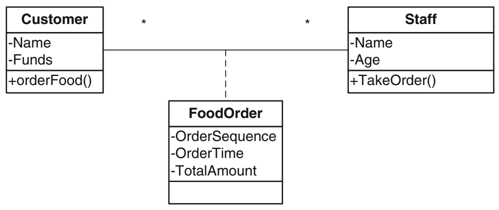
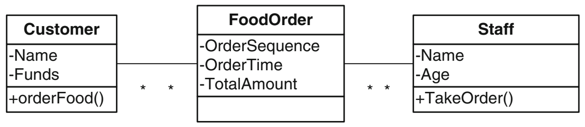
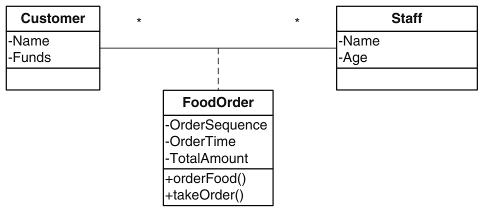
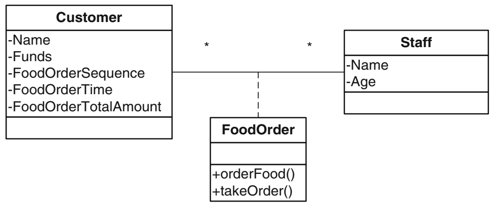
H4 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will perceive that the diagrams are more useful in performing the tasks as compared to individuals reading UML class diagrams where association classes contain *both* attributes and operations.

H5 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will perceive that the diagrams are more useful in performing the tasks as compared to individuals reading UML class diagrams where association classes contain *only* operations.

H6 Individuals reading UML class diagrams where association classes are modeled to comply with ontological guidelines will perceive that the diagrams are more useful in performing the tasks as compared to individuals reading UML class diagrams where association classes are objectified.

In summary, the outcome measures or dependent variables are the problem-solving question scores and the perceived usefulness of the class diagrams. The independent variable is the violation of ontological guidelines to test the above six hypotheses, a laboratory study with

Table 3 Illustrating the effects of the four types of UML association classes on problem-solving question

Sample problem solving question: *A customer tried to order food. She has selected the food she wanted to purchase but no food was delivered to her. What could have caused this problem?*	
 Customer -Name -Funds +orderFood() * * Staff -Name -Age +TakeOrder() FoodOrder -OrderSequence -OrderTime -TotalAmount **Fig 3A:** Example of association class developed as per the ontological guideline 1 (Treatment group T1).	 Customer -Name -Funds +orderFood() FoodOrder -OrderSequence -OrderTime -TotalAmount * * * * Staff -Name -Age +TakeOrder() **Fig 3B:** Example of association class developed by violating ontological guideline 1 (Treatment group T4).
 Customer -Name -Funds * * Staff -Name -Age FoodOrder -OrderSequence -OrderTime -TotalAmount +orderFood() +takeOrder() **Fig 3C:** Example of association class developed by violating ontological guideline 2 (Treatment group T2).	 Customer -Name -Funds -FoodOrderSequence -FoodOrderTime -FoodOrderTotalAmount * * Staff -Name -Age FoodOrder +orderFood() +takeOrder() **Fig 3D:** Example of association class developed by violating ontological guidelines 2 and 3 (Treatment group T3).
Relevant responses to the problem solving question	**Why 3A should be more effective than the other three figures in identifying such responses?**
1. The food order was misplaced.	Figure 3A shows that FoodOrder is associated with Customer and Staff. The mutual properties are placed in FoodOrder, suggesting that these properties are shared. Thus Figure 3A provides a hint that FoodOrder may have been misplaced by the Staff. In Figure 3B the meaning of FoodOrder is unclear as the class is no longer an association class. In Figures 3C and 3D the semantics of FoodOrder is unclear because of the presence of operations. For example, it may be unclear who takes the order.
2. Wrong order time was noted and thus the food was not delivered.	The meaning of OrderTime is clear in Fig 3A as it is modeled as a mutual property. OrderTime is not a mutual property in Fig 3B and Fig 3D. Although OrderTime is a mutual property in Figure 3C, the meaning of FoodOrder is unclear because of the operations present in it.

student subjects was conducted. Use of students as subjects is common in conceptual modeling studies [27] and has the advantage of increasing internal validity by minimizing subject heterogeneity and reducing possible effects of prior and differential domain and modeling experience.

3.1.2 Study 1: design

The study had one factor at four levels that reflect the class diagram not violating the guidelines and the three variations of the class diagrams violating the guidelines. A between-subjects study was designed where each subject received only one type of diagram. Four sets of class diagrams were developed accordingly. To increase external validity and remove domain knowledge as a potential confound, each set had class diagrams developed in two domains—"fast food operation" and "hotel reservation."

3.1.3 Study 1: tasks

The class diagrams and problem-solving questions were modified and adopted from two prior empirical studies [20, 22]. Parts of the class diagrams from the fast-food operation domain are shown in Figs. 1, 2, 3, and 4. These figures are accompanied by descriptions on how they were created. To have consistency in violating guideline 3 and to create class diagrams containing association classes with operations only, we employed two rules. First, the attributes of the association classes were not arbitrarily distributed to the classes, rather they were placed in those classes to which they are most closely related, for example, OrderSequence was placed in Customer and not in Staff (Fig. 4). Second, to clarify the origin of these attributes, a prefix was added. For example, the prefix "FoodOrder" was added to all the attributes of the Customer class that were moved from the FoodOrder association class. These rules were used to systematically create models that violate the guidelines, yet are informationally equivalent [28] to the compliant models. Appendix 1 shows the complete class diagram for the fast-food operation domain developed by the following ontological guidelines and diagram containing association classes with both attributes and operations.

3.1.4 Study 1: subjects and procedure

Ninety-seven subjects participated in the study, 24 in each group except 25 for the group where subjects received objectified classes instead of association classes. Undergraduate and graduate students from a Southern US University who were enrolled in IS analysis and IS design courses were recruited as subjects over two terms. Subjects were exposed to database and UML concepts prior to the study. Each subject was randomly assigned to one of the four types of class diagrams. Henceforth, subjects who received diagrams containing association classes complying with ontological guidelines will be referred to as group T1 and those who received diagrams containing association classes with both attributes and operations form group T2. Those who received diagrams containing association classes with only operations will be called group T3 and those who received diagrams where association classes were objectified make up group T4.

A pilot study was conducted with 12 subjects placed in two groups—T1 and T2. As a result of the pilot test, one problem-solving question on the food operation domain was reworded in the main study as it was ambiguous.

For the main study, the following experimental procedure was followed. First, subjects answered background questions on their modeling and domain knowledge. Because the differences in the diagrams for each condition were subtle, it was important to ensure that subjects had a good understanding of the association class concept prior to solving the problem-solving tasks. Therefore, subjects were asked to draw a UML class diagram that involves at least one association class. Next, subjects browsed through the general concepts of a UML class diagram with examples. Following this, subjects were trained to answer problem-solving tasks. This was important as answering problem-solving questions require assimilation of information from the diagram and integration with subjects' prior knowledge, and subjects are generally untrained to conduct such tasks. Subjects then first described the content of the diagram in details. This was done so that they are familiarized with the diagram. Following this, subjects answered three problem-solving questions. This sequence of content writing and answering problem-solving questions was repeated for another diagram from a different domain but in the same experimental condition. Thus, each subject performed the problem-solving tasks twice. Finally, subjects answered questions regarding the perceived usefulness of the diagrams. The order of the diagrams provided to the subjects was reversed for half of the subjects. This was done to cancel any domain order effect. The experimental materials used in this study are provided in Appendix 2.

3.1.5 Study 1: results

As discussed above, subjects were required to create a class diagram with association class to ensure sufficient understanding of the association class construct. The quality of these diagrams was assessed by an independent coder who was unaware of the objective of the study on a scale of 1–5 (1—very poor and 5—excellent). The average scores for each treatment ranged from 4.42 to 4.62, indicating that all subjects in the experiment had good-to-excellent command of the association class construct. Satisfied with this check of subject proficiency, we proceeded to analyze the descriptive statistics of the main study.

Table 4 indicates that the mean values and standard deviations for problem-solving performance, perceived usefulness of diagrams, UML modeling knowledge, and domain knowledge in each of the two domains across the four experimental groups T1–T4. Except for problem-solving performance, all variables were measured by multiple questions on a 7-point Likert scale (see Appendix 2). The mean scores for domain knowledge (in both domains) and modeling knowledge in all four groups were similar. The average correct problem-solving responses for all subjects were 2.58. The scores for the dependent variables (problem solving) were generally higher for the T1 group than for the other three groups. The pattern of correct problem-solving responses was consistent across the two domains for the group T1.

Figure 5a, b shows the problem-solving averages scores and the perceived usefulness scores in graphical form. Figure 5a shows that the mean score was highest for T1 and low for T2 and T4. However, the mean score for T3 was higher than T2 and T4. For perceived usefulness (Fig. 5b), the mean score for T2 was lowest compared to the other three groups. We investigate this pattern in an analysis of covariance (ANCOVA), reported below. We also checked for outliers in the data set using these plots (Fig. 5a, b), but none affected our results (reported later).

Table 4 Descriptive statistics

Variables	Scale	M T1	SD T1	M T2	SD T2	M T3	SD T3	M T4	SD T4	M Total	SD Total
Problem-solving average	1–4[a]	2.81	0.48	2.35	0.42	2.72	0.25	2.45	0.25	2.58	0.40
Problem-solving average—food	1–4[a]	3.01	0.84	2.49	0.67	2.89	0.30	2.60	0.51	2.75	0.64
Problem-solving average—reservation	1–4[a]	2.61	0.44	2.22	0.45	2.54	0.42	2.31	0.30	2.42	0.43
Perceived usefulness	1–7	5.58	0.90	4.58	1.32	5.28	0.91	5.67	0.93	5.28	1.10
Modeling knowledge	1–7	4.65	0.44	4.60	0.51	4.57	0.78	4.57	0.38	4.60	0.53
Food domain knowledge	1–7	4.29	0.99	4.65	0.79	4.44	0.56	4.26	0.77	4.41	0.79
Reservation domain knowledge	1–7	4.08	0.70	3.98	0.99	4.08	0.69	4.20	0.48	4.09	0.73

M mean, *SD* standard deviation T1 ($N = 24$): use of diagram without violation, T2 ($N = 24$): use of diagram with association class containing both attributes and operations, T3 ($N = 24$): use of diagram with association class containing only operations, T4 ($N = 25$): use of diagram with association class replaced by classes, problem solving: average correct responses of the problem-solving questions

[a] The problem-solving questions were open-ended, so the maximum score is undefined. However, the list of correct answers suggests a "practical" maximum of 4 for each domain

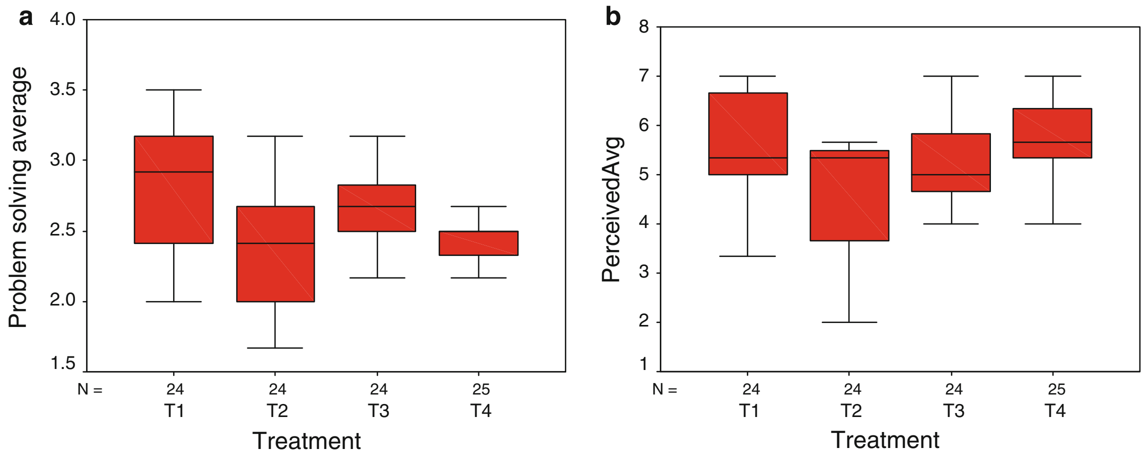


Fig. 5 **a** *Box plot* for problem-solving score. **b** *Box plot* for perceived usefulness score

Before testing the hypotheses, we checked the reliability of our measurements of the dependent variables—problem solving and perceived usefulness. For the reliability of the perceived usefulness measurements, for which we used three questions (Appendix 2), we employed Cronbach's α metric as a commonly used reliability metric [29]. Cronbach's α for this variable was 0.91, which is significantly greater than the generally acceptable threshold value of 0.7 [30].

Two coders, unaware of the objective of the study, evaluated the responses to the problem-solving tasks. The coders were given a code book that contained sample answers for each task. Coder 1 coded all the responses and coder 2 coded one-third of the responses. Their inter-rater reliability, the average Pearson correlation coefficient for the correct number of responses, was high (0.9).

To test hypotheses H1–H3, three sets of ANCOVA analyses were performed. The ANCOVA technique compares the effects of the different treatments (T1–T4) on a dependent variable (problem solving) and is able to control for other variables (domain knowledge, modeling knowledge, problem-solving time, and presentation order) that might have an impact on the dependent variable [31]. Such control is necessary, as subjects with high domain or modeling knowledge could potentially generate a high number of correct problem-solving responses even when referring to models that violate the ontological guidelines. Similarly, by spending a longer time with the models violating the ontological guidelines, subjects might still be able to generate a high number of correct responses [25]. Subjects might also perform better if the domains of the models are presented in a certain order (such as reservation first and then followed by food). Details of the logic of the statistics are provided in Appendix 3. Table 5 shows the treatment and control variables included in the ANCOVA analyses, their effects (*F* statistic), and the statistical significance of the effect (*P* value).

Table 5 ANCOVA analysis for H1–H3

ANCOVA	H1 (T1–T2 difference)		H2 (T1–T3 difference)		H3 (T1–T4 difference)	
Variables	*F*	*P*	*F*	*P*	*F*	*P*
Treatment	12.30	0.00*	0.90	0.17	10.71	0.00*
Order	0.29	0.30	0.00	0.47	0.21	0.32
Domain knowledge	1.44	0.12	0.62	0.22	0.58	0.23
Modeling knowledge	0.54	0.23	0.06	0.42	0.04	0.41
Problem-solving time	0.02	0.44	0.31	0.29	0.25	0.31
(Adjusted) r^2	0.26		−0.04		0.21	

Order order of the domains presented to the subjects

* Significance at $P < 0.05$ level

Table 6 ANCOVA analysis for H4–H6

ANCOVA	H4 (T1–T2 difference)		H5 (T1–T3 difference)		H6 (T1–T4 difference)	
Variables	*F*	*P*	*F*	*P*	*F*	*P*
Treatment	11.13	0.00*	2.03	0.08	0.10	0.37
Order	0.28	0.30	0.05	0.40	0.00	0.49
Domain knowledge	4.90	0.02*	3.64	0.03*	2.31	0.06
Modeling knowledge	0.30	0.28	3.06	0.04*	0.38	0.27
(Adjusted) r^2	0.28		0.14		0.08	

Order order of the domains presented to the subjects

* Significance at $P < 0.05$ level

To test hypothesis 1, the average number of correct responses of the problem-solving questions (both domains) was used as the measure for dependent variable and these numbers were compared between groups T1 and T2. The results indicate that the treatment had a significant effect ($F = 12.30$, $P < 0.05$), consistent with our hypothesis H1. For testing H2 and H3, the performance of groups T3 and T4 on the problem-solving questions were compared with the performance of group T1. Results indicate that H3 was supported as the treatment had a significant effect ($F = 10.71$, $P < 0.05$). H2 was not supported ($F = 0.90$, $P > 0.05$). Similar ANCOVA results were obtained when individual domains were considered (not shown in detail to conserve space). However, the effects were stronger for the reservation domain than for food operation domain. For testing H1 and H3, Table 5 also indicates that there was no statistically significant effect of the covariates. Thus, the differences in problem-solving performance in both cases can be solely attributed to the experimental treatment, that is, the different UML class diagrams.

To test hypotheses H4–H6, another three sets of ANCOVA analyses were performed, similar to those for H1–H3 above, but now using perceived usefulness as the dependent variable. Table 6 shows the treatment and control variables, their effects (*F* statistic), and the statistical significance of the effects (*P* values). The results indicate that only Hypothesis H4 was supported as the treatment was significant ($F = 11.13$, $P < 0.05$).

The focus of this study was on the dependent variable problem solving, and the results indicate that out of three ontological guidelines that we examined in this study, violations of guidelines 1 and 2 have a significant detrimental effect on domain understanding. One would expect that the pattern of results will be same for both dependent variables—problem solving and perceived usefulness. This was the case for the results of the hypotheses H1 and H2 with H4 and H5. However, while H3 was supported, H6 was not. For the T4 group (used for testing H3 and H6), the mean perceived usefulness was the highest and the mean problem-solving score was one of the lowest among all four groups (Table 4). This indicates that in general, subjects make a correct assessment of the usefulness of the models for problem solving, as the observed usefulness (problem-solving performance) and their perceived usefulness are similarly higher or lower than the comparison group. One possible explanation for the exception in group T4 could be that subjects realized that an objectified association class does not carry the same relationship semantics as an association class and would therefore be less useful; yet, that relationship aspect is still implicit in the model as the objectified association class links the same classes and carries the same attributes.

3.1.6 Study 1: discussion

It is important that when class diagrams are compared for domain understanding, they should be developed consistently for a given domain. Care was taken to ensure this. The diagrams that violated guidelines were systematically derived from the diagram that had no violations. All diagrams used in the study were informationally equivalent [28], meaning that a model element that appeared in one diagram can also be found in others.

For two reasons, this paper does not focus on testing empirically the effect of violations of guidelines 4–6 (Table 1). First, as we indicated above, the modeling situations prohibited by these guidelines are not commonly encountered in practice. Second, unlike violations of guidelines 1–3, violations of the rest of the guidelines cannot be introduced consistently. For example, violating guideline 4 would require connecting association classes with other classes, resulting in models that will not be informationally equivalent with models developed without violations (such as Fig. 1). A similar situation will arise for violating guidelines 5 and 6. Thus, the paper focuses on violations related to only attributes and operations of the association classes.

A surprise finding of this paper is that there is no significant effect of violating guideline 3 on domain understanding. This is contrary to the expectation that recovering the true meaning of association classes will be more difficult if these classes have more number of violations as diagrams that violated guideline 3, also violated guideline 2. To investigate this unexpected finding, we conducted a verbal protocol analysis study. This study, described below, was intended to more closely identify the sources of answering the problem-solving questions and to understand the process of problem solving by subjects in the two groups.

3.2 Study 2: protocol analysis

Protocol analysis is a technique that is often used in experimental psychology and has been widely adopted in studies related to IS [33–35]. In protocol analysis, subjects are asked to verbalize their thought processes while performing a task (such as problem solving) and the verbal data thus obtained is analyzed. The data obtained from protocol analysis are argued to reveal the mental processes that take place as individuals work on problem-solving tasks [36, 37]. Protocol analysis is primarily an exploratory technique and is used to identify evidence in support or in contradiction of theory [38]. In our case, we wish to explore plausible reasons to explain the results of our first study, especially the lack of support for our hypotheses H2, H5, and H6.

Specifically, the protocol study was conducted to identify difficulties faced by the subjects in answering problem-solving questions. We focus on difficulties, also called *breakdowns* [39], as the number of breakdowns are an indication of the cognitive difficulty of performing a task [40]. The theory of cognitive fit [41] suggests that when the information emphasized in a problem representation matches the type of information that users use in a task, users perform the task better than when there is no good match between information representation and task. Accordingly, when users perform problem-solving tasks related to domain understanding on conceptual models that violate the proposed guidelines (such as those who are in T2 and T3 groups), they might exhibit a large number of cognitive breakdowns.

The main drawback of protocol analysis is that the verbalization has the potential to intrude into the cognitive process of the subject. It is generally acknowledged that there is a trade-off between the ability to gather data and the level of intrusion into the cognitive processes. For example, the experimental study reported in Sect. 3.1 did not introduce into the thought processes, but as a result only relatively limited data were collected, sufficient to allow us to test hypotheses, but insufficient to provide further explanation of why subjects behaved in a certain way.

3.2.1 *Study 2: design*

To use protocol analysis technique, subjects verbalize their thought processes and strategies as they perform the tasks. Consistent with other protocol analysis studies (e.g., [27, 34, 40]), a small number of subjects was recruited in this study. To contrast the sources of information for problem-solving tasks and the cognitive difficulties faced by subjects in group T3, subjects in group T2 were also selected. T2 was chosen as the difference in the diagrams received by subjects in the two groups (T2 and T3) is only the placement of operations—diagrams for T2 had association classes with operations and attributes, whereas diagrams for T3 had association classes with only operations. This allowed us to identify why the placement of operations in the association classes had different effects on the performance of the problem-solving tasks depending on the presence or absence of attributes in the association class. A single-factor, two-level, between-subject design was employed for this purpose. Five subjects were placed in group T2 and five subjects in group T3. The subject profile and the procedure of this study were exactly the same as for study 1, except that the subjects verbalized their thought processes while solving the problem-solving tasks.

3.2.2 *Study 2: results*

Two coders identified the cognitive breakdowns faced by the subjects while answering the problem-solving questions. Breakdowns are either *explicit* or *implicit* [40]. Explicit breakdowns can be identified when subjects specifically verbalize the difficulty that they face in answering. Breakdowns are implicit when subjects experience the breakdowns, but do not state them explicitly. One way to measure an implicit breakdown is to identify the incomplete lines of thought of modelers. An example of such difficulty faced by a subject is “class is a set of objects that has same properties … ahhh … staff has delivery receipt.” It was found that most difficulties faced by subjects were implicit and subjects in group T3 faced fewer implicit difficulties than subjects in group T2 (Table 7).

Next, we analyzed what aspect of the model that subjects referred to in answering the problem-solving questions. As the models consist of classes and association classes only, the number of instances of direct references to the classes and association classes made by the subjects were identified. Examples of association classes verbalized by subjects include “association between Guest and Hotel classes,” “membership association class,” and “the association membership.” Similarly, when subjects verbalized

Table 7 Implicit cognitive difficulties faced by subjects

Domain	Food operation	Reservation	Total
A2	3	1	4
B2	3	2	5
C2	4	1	5
D2	2	1	3
E2	2	1	3
Total	14	6	20
A3	1	2	3
B3	3	1	4
C3	1	1	2
D3	1	0	1
E3	1	1	2
Total	7	5	12

A–E subjects, *2* group T2, *3* group T3

classes directly (e.g., Privileged Guest classes or subclasses of Guest classes), then such verbalizations were considered. These references indicate that the subjects were using these constructs as depicted in the models. However, if a subject referred to the concept of a class or association class in general (e.g., Guest can reserve a room), then such references were not considered. This is because such references may be based on existing cognitive models and prior domain knowledge. Recall that multimedia-learning theory suggests that learning is the integration of new information (model) into existing cognitive models (prior domain knowledge). Hence, rather than being a confound or irrelevant, instances of the use of prior knowledge are an important part in the cognitive process. Table 8 shows that to answer problem-solving questions, subjects who used association classes developed with only operations (group T3), depended mainly on the classes (referred to 16 times explicitly). On the other hand, subjects who used association classes developed with both attributes and operations (group T2) depended primarily on association classes (referred to 32 times explicitly).

3.2.3 Study 2: discussion

Several observations were made based on the results of this protocol analysis study. First, the subjects in group T2 (association classes with attributes and operations) faced more difficulties than subjects in T3 (association classes with operations only). This result was consistent with the testing of Hypotheses H2 and H3 in our first study. Second, subjects in group T2 used a larger number of association classes for their reasoning than subjects in T3. One possible reason is the fact that, ontologically, association classes with attributes represent mutual properties, whereas those without attributes do not and are in fact ontologically without meaning. The observed dependence of subjects in group T2 on association classes is in agreement with the idea that only association classes with attributes provide meaningful information. In contrast, the association classes without attributes in group T3 do not provide any ontologically meaningful information (they do not represent mutual properties). At this point, we cannot resolve these two alternative explanations (information content vs. ontological meaning) and suggest further research.

Table 8 Information sources for problem-solving tasks

Subject	Number of direct references to association classes			Number of direct references to classes		
Domain	*F*	*R*	Total	*F*	*R*	Total
A2	6	3	9	2	2	4
B2	1	2	3	0	2	2
C2	6	2	8	3	2	5
D2	6	4	10	0	0	0
E2	1	1	2	0	0	0
Total	20	12	32	5	6	11
A3	0	0	0	1	2	3
B3	0	1	1	3	5	8
C3	0	0	0	1	0	1
D3	0	0	0	1	0	1
E3	1	1	2	2	1	3
Total	1	2	3	8	8	16

F fast-food operation, *R* reservation, *A–E* subjects, *2* group T2, *3* group T3

Subjects in group T3 received diagrams where the association classes had no ontologically meaningful information. Instead of using association classes for their reasoning, subjects in group T3 were forced to examine the associated classes. The information in the classes might have been sufficient to understand the domain, with subjects' background knowledge and existing cognitive structures filling in the relationships that were not provided by the association classes. Hence, subjects in group T3 might have had to acquire and integrate less information from the diagram (they were not required to acquire and integrate the relationship information) and therefore faced less difficulty than subjects in group T2. While positive from a cognitive perspective, such an explanation is negative from the requirements engineering perspective, as the interpretation is being left at the discretion of the analysts, rather than being explicitly modeled, resulting in possibly incorrect representations. Future research might test this assumption of subjects filling in the "missing links" from their existing background knowledge by using entirely unfamiliar domains, rather than familiar domains as used in this study.

The total number of classes and association classes verbalized by subjects in group T2 (43) is significantly higher than that by subjects in group T3 (19). A higher number of references might indicate more difficulty in solving the problems, which is in line with the previous explanation about the possible ease of acquisition and integration of the information in group T3.

Finally, we interpret the information in Tables 7 and 8 with some caution as there is a limitation in using verbal protocol analysis method. In spite of reminding subjects that they need to verbalize the thought process in performing problem solving, subjects might not always do so. Hence, the exact number of difficulties faced by subjects or the exact number of concepts referred to by subjects is difficult to determine precisely. However, we believe that the data obtained from this study provide sufficient evidence of users' mental processes to at least give initial insights to explain the results of our first study (Sect. 3.1). Clearly, further research with a larger sample, as well as more granular process-tracing methods, similar to those in [35], would be useful to obtain a clearer picture of the process and a more certain explanation.

4 Overall discussion and conclusion

Based on ontological arguments, we developed different versions of UML class diagrams through systematic manipulation of attributes and operations in association classes. We tested the effectiveness of these diagrams for application domain understanding. Three of our six hypotheses are supported. Based on the evidence from a protocol analysis study, we have provided possible reasons as to why hypotheses 2 and 5 are not supported. In summary, we suggest that our results offer empirical support for the ontological guidelines for the modeling of application domains with UML, in agreement with much previous work (e.g., [20, 27, 40, 42, 43]).

As with any empirical study, there are limitations on what can be achieved and claimed. We have examined only three of the six guidelines concerning associations and association classes, and neglected those guidelines that we believed are not widely applicable in practical modeling situations. Future work may extend our investigation to the remaining guidelines. Further, as discussed in Sect. 3, the present study focuses on establishing internal validity, that is, checking whether it works in principle. It neglects external validity, that is, generalization to IS practitioners and realistically sized models and development projects. Having found supporting evidence for our hypotheses, future research is now required to ensure that the rules are applicable in practice and that the benefits to domain understanding that we have demonstrated can be realized in applied settings.

We believe that the results of the protocol analysis are interesting and raise further questions for future investigation. The protocol analysis indicates that when association classes are poorly constructed (possessing only operations), subjects must rely on other parts of the diagrams (i.e., classes) to understand them. In this case, subjects were still able to perform the problem-solving tasks well as they obtained the necessary information about the application domain from the associated classes rather than from the association classes. This indicates that the information in the classes *compensated* for the information that was not present in the association classes (developed with only operations). Models that are developed by violating ontological guidelines and thus constructed poorly can still be useful if such models have sufficient information to perform the tasks. To our knowledge, this is the first time that such compensating behavior has been examined, as most prior studies have not employed process-tracing methods. While this is a welcome result, it raises further questions as to precisely how this compensation works, what the limits of compensation are, and what specific information is required for subjects to be able to compensate for the lack of ontological meaning of models. Further, the precise interpretation of this compensating effect now requires effort in theorizing the mechanism that underlies this effect.

The results of this study are immediately applicable. Specifically, the modeling guidelines do not require any extensions or adaptations to UML, let alone a new language or new development tools. Instead, they are based on restricting the set of valid UML models when applied to domain modeling. As we discussed earlier in the paper, these guidelines are applicable only to domain modeling, not software modeling and can in fact serve to remove "design bias" from domain models.

The contributions of this paper are not only to provide initial validation of three proposed modeling guidelines. To the best of our knowledge, this is also the first empirical paper that has studied the effect of using operations in association classes for domain understanding. Further, the fact that ontologically derived modeling rules are found to be advantageous to domain understanding also serves to support the choice of the particular ontology for deriving such modeling rules. This implicitly confirms previous results in other studies that are based on the same ontological foundation.

Appendix 1: sample diagrams (fast-food operation) used in the study

See Figs. 6, 7.

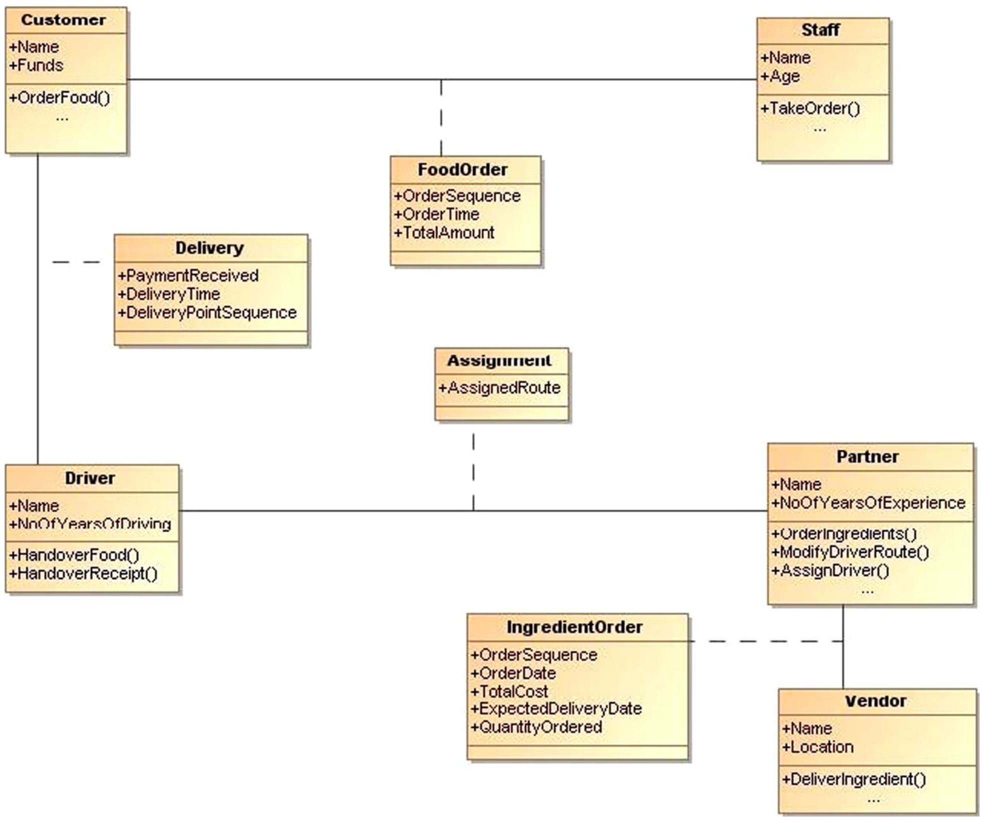


Fig. 6 Class diagram developed by following the ontological guidelines

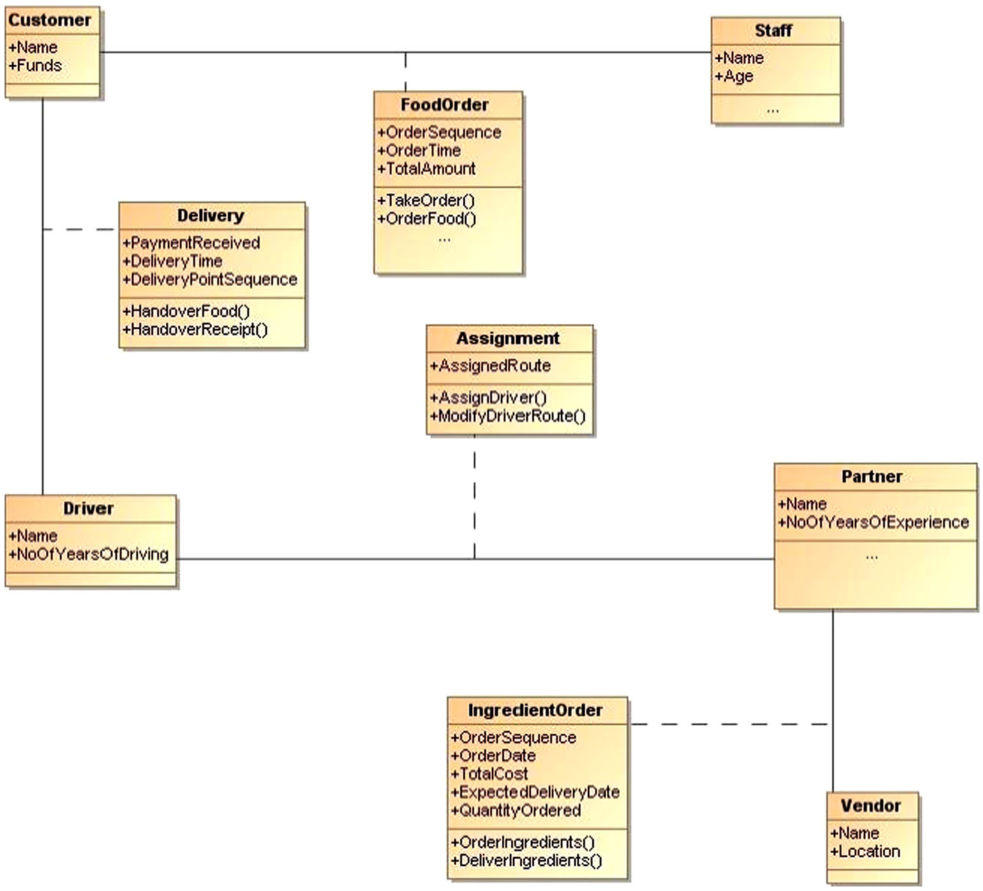


Fig. 7 Class diagram developed by violating guideline 2

Appendix 2: experimental materials

Variables: modeling and domain knowledge

1. To what extent do you know data modeling concepts (such as classes, operations, and attributes)?
2. To what extent do you have experience in using data modeling concepts (such as classes, operations, and attributes)?
3. To what extent do you know UML association class?
4. To what extent do you have experience in using UML association class?
5. To what extent do you know the operation of a food restaurant?
6. To what extent do you have experience in the operation of a food restaurant?
7. To what extent do you know the operation of a hotel reservation?
8. To what extent do you have experience in the operation of a hotel reservation?

Variable: perceived usefulness of the diagrams

1. To what extent do you think that the diagrams helped to answers the questions?
2. To what extent do you think that the diagrams made it easier to complete answering the questions?
3. To what extent do you think that the diagrams enhanced your effectiveness on answering the questions?

Variable: domain understanding

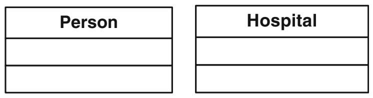
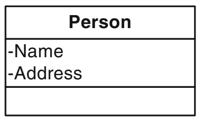
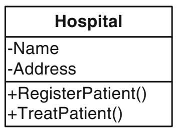
1. A customer tried to order food. She has selected the food she wanted to purchase but no food was delivered to her. What could have caused this problem?
2. A Driver went about his route to drop off the ordered food. However, when he reached a delivery point, he could not deliver the ordered food. What could have caused this problem?
3. On a particular day, the partner of the restaurant ordered ingredients for preparing food. The ingredients did not reach on the expected delivery date. What could be the possible reasons?
4. A guest was not a privileged hotel guest but was allowed to get a car pick up service. How could this have happened?
5. A guest had 7 days of reservation in the hotel. At the end of the stay, the guest did not pay for her stay. How could this have happened?
6. A privileged guest received the pick up service even after his membership expired. How could this have happened?

Task on developing UML class diagram

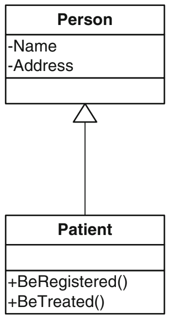
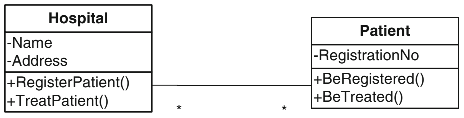
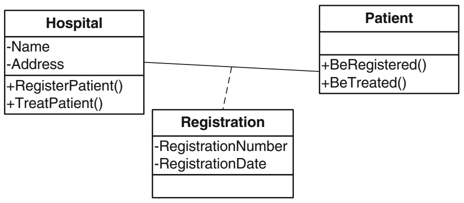
In the following space draw a UML class diagram for the description below using at least one association class.

> A hospital treats patients. For each treatment, the hospital needs to record the doctor, the treatment code, and the date.

UML class concepts

Concept	Definition	Example
Class	A class is set of objects that share the same properties and/or behaviors	*Person* and *hospital* are concepts and therefore are modeled as classes 
Attribute	Attributes are properties held by the members of a class. Attributes can have constant (such as date of birth) or variable values (such as address)	The person class can have name and address as attributes 
Operations	Operations are functions or services that are provided by all the instances of a class to invoke behavior in an object	The two operations of the hospital class are register patients and treat patients 

continued

Concept	Definition	Example
Subclasses	A subclass has more attributes or/and more operations than the general class	A patient is a subclass of a person  Person -Name -Address Patient +BeRegistered() +BeTreated()
Association	Association is the relationship among instances of classes	Hospital and patient are related as hospital treats patients  Hospital -Name -Address +RegisterPatient() +TreatPatient() Patient -RegistrationNo +BeRegistered() +BeTreated() * *
Association class	An association class is an association that has attributes or/and operations of its own	Registration is an association class that has attributes registration number and registration date  Hospital -Name -Address +RegisterPatient() +TreatPatient() Patient +BeRegistered() +BeTreated() Registration -RegistrationNumber -RegistrationDate

Training on answering problem-solving questions

Please look at Fig. 8 carefully. The figure is drawn using the concepts mentioned in the earlier page. The figure describes the following situation.

A *patient class* has the attributes name and age and an operation get treated. *Admitted patient* is a subclass of *patient* as it has additional attributes—*admission date* and *bed number* and an additional operation—get admitted. The *physician* class is associated with the *patient* class.

A physician is dissatisfied with her work. Why might this be?

Sample answers

Using Fig. 8, you come up with answers by making inferences based on the information in the diagram combined with your own background information. For example, to come up with answer 1 (in Table 9), you have to look at the classes *admitted patient* and *patient* in Fig. 9 and infer that some patients might not be admitted.

Appendix 3: the ANCOVA statistical technique

The ANCOVA technique evaluates the effect of each treatment or control variable by first calculating the mean of the dependent variable (e.g., problem-solving scores) for each experimental group ("treatments") or control variable. Next, the sum over all observations of the squared differences of the dependent variable score from the mean of the dependent variable score of the group of each observation is computed, called the sums of squares within groups.

$$\mathrm{SS}_{\text{within}} = \sum_{i=1}^{g}\sum_{j=1}^{n_i}\left(Y_{ij} - \bar{Y}_l\right)^2$$

Here, Y denotes the dependent variable, a bar denotes the mean, g is the number of groups, n_i is the number of observations in group i, and c is the overall number of observations. Also, the sum over all groups of the product of the number of observations in that group and the squared differences of the dependent variable mean for that group from the overall mean of the dependent variable is computed, called the sums of squares between groups.

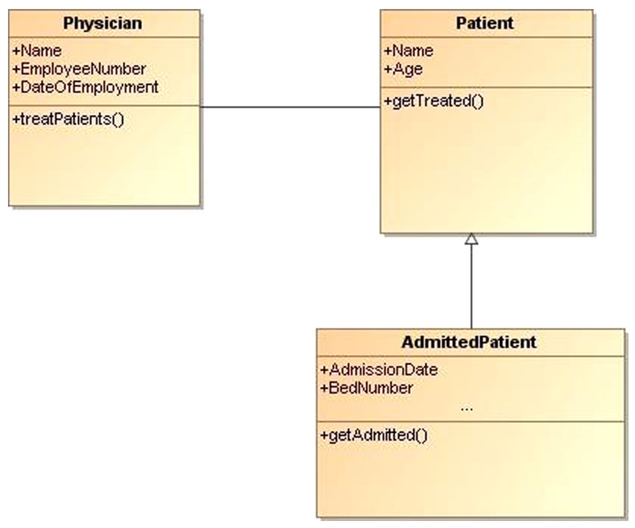


Fig. 8 A patient admission situation

Table 9 Possible answers with explanation

Example answers	Explanation
Some patients are refused admission	Even after recommended by the physician, the hospital refuses admission
Treatment not helping to some patients	The physician's treatment does not help patients for treatment
The physician is overworked	Too many patients to handle

$$\mathrm{SS_{between}} = \sum_{i=1}^{g} n_i(\bar{Y}_l - \bar{Y})^2$$

Next, each of these "sums of squares" is divided by their degrees of freedom, defined as the number of data points for that calculation minus the number of parameters calculated. This equals the number of groups minus one (the overall mean is calculated) for the sums of squares between groups, and the number of total observations minus the number of groups (each group has a group mean that is calculated) for the sums of squares within groups. This yields the "mean sums of squares" per degree of freedom.

$$\mathrm{MS_{within}} = \frac{1}{g-1}\mathrm{SS_{within}} \quad \mathrm{MS_{between}} = \frac{1}{n-g}\mathrm{SS_{between}}$$

The logic of the ANCOVA rests on the observation that, if the treatment variable had no effect on the group means (i.e., all group means are equal, and thus equal to the overall mean), the mean sums of squares between the groups would be equal to the mean sums of squares within groups; in other words, their ratio should be one. This ratio is called the F statistic, reported in our tables in the text, and it is distributed according to an F distribution.

$$F = \frac{\mathrm{MS_{between}}}{\mathrm{MS_{within}}}$$

One can test whether the F statistic that is calculated is significantly different from the expected value of one. This is done by calculating, from the F cumulative distribution function, that probability with which the observed F statistic would be found, if the true F statistics was one. This is the P value reported in our tables in the text. If this probability is sufficiently low (generally this cutoff is assumed to be 0.05), one concludes that the observed F statistic does not come from a distribution for which the true F statistic is one; in other words, the true F statistic is different from one, thus the ratio of mean sums of squares is different from one, and therefore the dependent variable mean differs between the treatment groups or categories. To assess to what extent the different treatment groups or categories explain the observed variation of dependent variable scores, one can compare the sums of squares calculated between the groups to the sum over all observations of the squared difference of the dependent variable from the overall mean of the dependent variable.

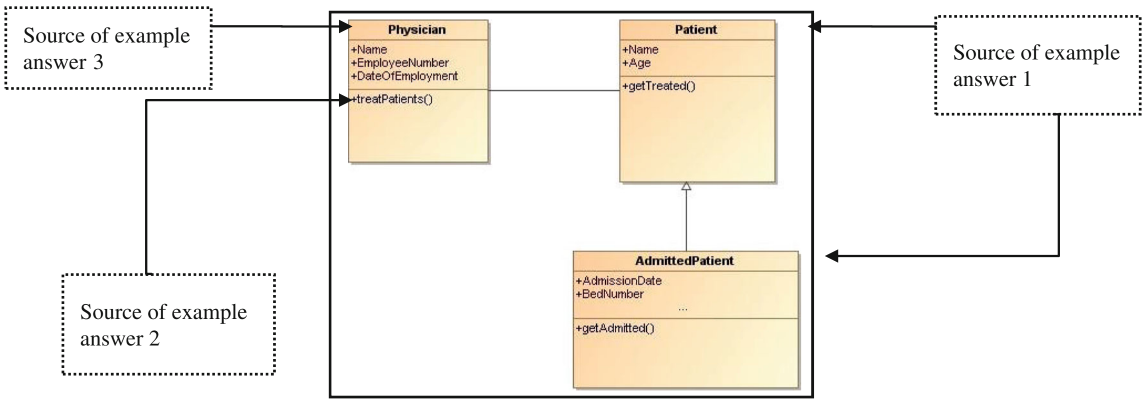


Fig. 9 A patient admission situation—sources of answers

This is the r^2 value, which can be adjusted to account for the effects of sample size and number of groups.

$$SS_{\text{total}} = \sum_{i=1}^{n} (Y_i - \bar{Y})^2$$

$$r^2 = \frac{SS_{\text{between}}}{SS_{\text{total}}} \qquad r^2_{\text{adj}} = 1 - \frac{n-1}{n-g}(1 - r^2)$$

Generally, a higher r^2 value is better, though a value of approximately 0.25 is suggested to indicate a medium-strength effect [31, 32].

References

1. Wand Y, Weber R (1993) On the ontological expressiveness of information systems analysis and design grammars. J Inf Syst 3:217–237
2. Evermann J, Wand Y (2005) Toward formalizing domain modeling semantics and language syntax. IEEE Trans Softw Eng 31:21–37
3. Dobing B, Parsons J (2006) How UML is used? Commun ACM 49:109–113
4. Fettke P (2009) How conceptual modeling is used. Commun Assoc Inf Syst 25:571–592
5. Dobing B, Parsons J (2008) Dimensions of UML diagram use: a survey of practitioners. J Database Manag 19:1–18
6. Davies I, Green P, Rosemann M, Indulska M, Gallo S (2006) How do practitioners use conceptual modeling in practice? Data Knowl Eng 58:358–380
7. Parsons J (2011) An experimental study of the effects of representing property precedence on the comprehension of conceptual schemas. J AIS 12:441–462
8. Evermann J (2005) The association construct in conceptual modeling—an analysis using the Bunge ontological model. CAiSE, Porto
9. Milicev D (2007) On the semantics of associations and association ends in UML. IEEE Trans Softw Eng 33:238–251
10. Rumbaugh J, Blaha WP, Eddy F, Lorensen W (1991) Object oriented modeling and design. Prentice Hall, Englewood Cliffs
11. Martin J, Odell J (1992) Object oriented analysis and design. Prentice Hall, Englewood Cliffs
12. Bahrami A (1999) Object-oriented systems development using UML, 3rd edn. McGraw-Hill, New York
13. OM Group (2004) UML 2.0 superstructure specification, revised final adopted specification. Available: http://www.omg.org
14. Stevens P (2002) On the interpretation of binary associations in the unified modeling language. Softw Syst Model 1:68–79
15. Embley DW (1992) Object-oriented systems analysis: a model-driven approach. Prentice Hall, Englewood Cliffs
16. Siegfried S (1995) Understanding object-oriented software engineering. IEEE Press, New York
17. Liu Z, He Z, Li J, Chen Y (2003) A relational model for formal object-oriented requirement analysis in UM. In: LNCS 2885. Springer, Berlin, pp 641–664
18. Evermann J, Wand Y (2005) Ontology based object-oriented domain modelling: fundamental concepts. Requir Eng J 10:146–160
19. Bunge M (1977) Ontology I: the furniture of the world, vol 3. D. Reidel, Dodrecht
20. Evermann J, Wand Y (2006) Ontological modelling rules for UML: an empirical assessment. J Comput Inf Syst 47:156–184
21. Poels G (2011) Understanding business domain models: the effect of recognizing resource-event–agent conceptual modeling structures. J Database Manag 22(4):69–101
22. Evermann J, Halimi H (2008) Associations and mutual properties—an experimental assessment. In: Americas conference on information systems, Toronto
23. Calder BJ, Phillips LW, Tybout AM (1981) Designing research for application. J Consum Res 8:197–207
24. Mayer R (2001) Multimedia learning. Cambridge University Press, Cambridge
25. Gemino A (1998) Comparing object oriented with structured analysis techniques in conceptual modeling. PhD thesis, Sauder School of Business, University of British Columbia, Vancouver
26. Gemino A, Wand Y (2004) A framework for empirical evaluation of conceptual modeling techniques. Requir Eng J 9:248–260
27. Burton-Jones A, Meso P (2006) Conceptualizing systems for understanding: an empirical test of decomposition principles in object-oriented analysis. Inf Syst Res 17:38–60
28. Parsons J, Cole L (2005) What do the pictures mean? Guidelines for experimental evaluation of representation fidelity in diagrammatical conceptual modeling techniques. Data Knowl Eng 55(3):327–342
29. Allen MJ, Yen WM (2002) Introduction to measurement theory. Waveland Press, Long Grove
30. Nunnally J, Bernstein I (1994) Psychometric theory, 3rd edn. McGraw Hill, New York
31. Levine T, Krehbiel T, Berenson M (2010) Business statistics: a first course, 5th edn. Prentice-Hall, Englewood Cliffs
32. Cohen J (1988) Statistical power analysis for the behavioral sciences, 2nd edn. Lawrence Erlbaum Associates, Hillsdale
33. Stephen O, Pearl B, David B (2006) Protocol analysis: a neglected practice. Commun ACM 49:117–122
34. Hungerford BC, Hevner A, Collins RW (2004) Reviewing software diagrams: a cognitive study. IEEE Trans Softw Eng 30:82–96
35. Evermann J (2008) An exploratory study of database integration processes. IEEE Trans Knowl Data Eng 20:99–115
36. Newell A, Simon HA (1972) Human problem solving. Prentice Hall, Englewood Cliffs
37. Ericsson KA, Simon HA (1984) Protocol analysis: verbal reports as data. MIT Press, Cambridge
38. Gobet F, Charness N (2006) Chess and games. In: Ericsson KA, Charness N, Fletovich PJ, Hoffman RR (eds) The Cambridge handbook of expert performance. Cambridge University Press, New York, pp 41–67
39. Vessey I, Conger S (1994) Requirements specification: learning object, process, and data methodologies. Commun ACM 37:102–113
40. Bera P, Krasnoperova A, Wand Y (2010) Using OWL as a conceptual modeling language. J Database Manag 21:1–28
41. Vessey I, Galletta D (1991) Cognitive fit: an empirical study of information acquisition. Inf Syst Res 2:63–84
42. Gemino A, Wand Y (2005) Complexity and clarity in conceptual modeling: comparison of mandatory and optional properties. Data Knowl Eng 55:301–326
43. Shanks G, Tansley E, Nuredini J, Tobin D, Weber R (2008) Representing part-whole relations in conceptual modeling: an empirical evaluation. MIS Q 32:553–573