Conceptual Modeling: Enhancement Through Semiotics

Veda C. Storey[1] and Bernhard Thalheim[2(✉)]

[1] Computer Information Systems, J. Mack Robinson College of Business, Georgia State University, Atlanta, GA, USA
vstorey@gsu.edu
[2] Department of Computer Science, Christian-Albrechts-University, Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract. Conceptual modeling uses languages to represent the real world. Semiotics, as a general theory of signs and symbols, deals with the study of languages and is comprised of syntax, semantics, and pragmatics. Pragmatics includes the explicit representation of the intentions of users. A common assumption is that all levels of database design (user, conceptual, logical, and physical) can be modeled using the same language. However, languages at the conceptual level are often enhanced by concepts that attempt to capture inherent pragmatics. This research proposes that concepts from semiotics can provide the background needed to understand an application. Specifically, pragmatics and semantics are considered at both the user and conceptual level, based on proposed constraints.

Keywords: Conceptual modeling · Languages · Semiotics · Semantics · Constraints

1 Introduction

Conceptual models act as mediators between the application and an implementation [11]. Conceptual modelers often attempt to model situations that occur in the real world using one language as a construction mechanism, and a model for a schema. Representing how the world operates must be described at the right level of specification. This tends to be done, for example, using an entity-relationship diagram as a modeling tool. However, it is difficult to expect one language to be able to handle all phases of modeling. Semantic issues need to be captured and modeled during both the design phases. The objective of this research, therefore, is to understand how to create better conceptual models by considering these different levels of abstraction and how they might be addressed. Although language is usually the main vehicle for modeling, additional understanding is needed for collaboration among stakeholders. Semiotics, as a general theory of signs and symbols, deals with the study of languages, and could serve as the needed background. The contributions are to: propose that models should be defined from the perspective of semiotics, and propose an additional set of constraints.

H.C. Mayr et al. (Eds.): ER 2017, LNCS 10650, pp. 182–190, 2017.
https://doi.org/10.1007/978-3-319-69904-2_15

2 Modeling Challenges in Conceptual Modeling

Levels of Abstraction. Many modeling languages are applied at different levels of abstraction. Business issues might be applied at the application level. Prescription issues for implementation are at a detailed level of specification. Although different, they are often all represented by an entity-relationship diagram.

Semantics. Semantics (meaning of terms) is challenging [5]. Constraints are often used as a surrogate for business rules [6]. Attempting to capture and represent semantics in terms of first-order predicate logic seems restrictive. Implicit or lexical semantics contribute to complete semantics.

Inclusion Constraints. These could be class-based; for example, a student is a person. The person identification is reused for student as a co-existence constraint, expressible via identification (becoming a foreign key constraint in the relational model). Then an enforcement mechanism can be: (1) canonically declared based on reference existence and reference enforcement; or (2) expressed by the on-*event*-if-*condition*-then-*action* (ECA) paradigm. The enforcement can be refined for control, application, optimization, and exception handling. If the inclusion constraint is not class-based, but value-based, then support and enforcement become more challenging. For example, the *Student* type may use an attribute *Name,* which corresponds to a person's *Name* in a type *Person*.

Cardinality Constraints. These have two main approaches to define their semantics: look-up and participation. Look-up works well for binary associations without relationship attributes. Participation constraints mix two different kinds of semantics with rigidity for extreme cases, despite the need to represent normal cases. 'Min/Max' captures the absolute extreme for all potential cases. The 'min' captures a (generalized) inclusion constraint; 'max' is intended to capture a (generalized) multiplicity constraint. For a relationship where the minimum participation could be '0' (someone is a student but not taking courses yet), a null value would be allowed in an implementation. However, a "*normal*" interpretation of the relationship is that a student must be registered for at least one course (null not allowed). Cardinality constraints impact other constraints in the schema [3].

Implicit Constraints. Constraints can be implicit or hidden due to syntax construction. The eER modeling language uses relationship types with inherent (construction) inclusion and existence constraints as *based-on constraints*. Relationship objects reference their component objects; for example, entity objects. Therefore, the relationship objects can only exist if the corresponding entity object exists, making the semantics implicit, based upon the way in which relationships are constructed and used. They become explicit in the corresponding SQL specification.

Type Semantics. eER modeling uses a Salami-slice strategy, oriented on the homogeneity of types and thus on decomposition into small, meaningful semantic units. Things in the application domain are multifaceted. A human is represented via a *Person* type that is separated from the *Student* type, which is associated via an IsA relationship (or subclass), to the *Person* type. At the same time, *Student* can be associated with other types, such as: *student_engagement*, *student_facilities, dormitory*, etc. Depending upon

the view, a student might best be considered using the notion of a student or the notion of the more general object, person. Research has analyzed classification challenges [4].

Implicit Representation of Viewpoints. At the application level, it might be beneficial to consider user viewpoints that are represented as views [11]. For instance, a student might best be considered, including more general objects, e.g. person.

Separation of Syntax and Semantics. The separation of syntax and semantics is generally problematic. Most modelers learn a language using simple problems. However, real world problems are complex, so one language, or modeling technique, is not appropriate for all. It is impossible to represent a business problem at an application level of abstraction and implementation issues based on a singleton diagram. The problem is understanding and representing semantics.

Restricted and Mixed Semantics. Instead of general constraint frames, specific cases are often considered; e.g., mapping ratios (1:N, N:M, 1:1) to capture some binary relationship semantics. Sometimes, N:M ratios declare the maximum to be higher than 1. Look-up and participation cardinalities may be used with the same syntactic notion.

3 Models, Expressions, and Stakeholder Levels

Models and Conceptual Models. The notion of a *model* is complex and not necessarily well understood; similarly, for the process of modeling. Consider four perspectives: (1) the origins to be considered by the model; (2) the profile of the model (e.g. its function, purpose, or goal); (3) the stakeholders or the community of practice that the model must satisfy; and (4) the context within which the model and the origins are considered. The first two perspectives are internal; the second two, external.

A model is guided on its background [10]: the *grounding* of the model (paradigms, postulates, theories, culture, and conventions); and the *basis* for the model (e.g. languages used, concepts and conceptions, community, and commonly accepted practices). The *basis* of a model may change on demand. The perceptions of users might need to be represented in a model. Multiple coherent perceptions, a description of a system, or an augmented system might also be useful. A model can have many different purposes: to describe or explain a situation; specify and represent a concept someone has in mind; to aid in communication among stakeholders; or to decompose complex situations. A *model* is a well-formed, adequate and dependable artifact, commonly accepted by its community of practice within a given context [10, 11].

Semiotics of Signs: Icons, Symbols and Indexes. Semiotics, the study of the theory of signs, emphasizes the properties of things in their capacity. It is reasonable to apply semiotics to aid in this understanding since, before using a modeling language, it is first necessary to understand the language and its inherent bias.

Syntax refers to the arrangement of words in sentences and phrases. Syntax should be simple, parsimonious, and harmonic.

Semantics is concerned with the meaning of sentences and defines the interpretation of a sentence in the real world, depending on its context. It refers to the meaning of signs and what they represent in the real world.

Pragmatics considers the relationship between parts of sentences or signs and their users within a situation and context. It is user-dependent.

Although language is the main vehicle for modeling, semiotics is the background needed for understanding so that collaboration among stakeholders can result. Syntax, semantics, and pragmatics may follow different paradigms, leading to some effective use. The strictness of first-order predicate logic might be inappropriate during modeling. It is, however, needed in the final result. For example, natural utterances use the connective "and/or" with the meaning of logical OR. Similar observations can be made for all connectives, especially, for quantifiers.

Syntax has been well investigated for formal languages. Semantics can be defined in a variety of ways; e.g. for evaluation of variables, incorporation of context, scope of states, exceptions, and matching between syntactic language and semantic structure [8]. Problems arise when pragmatics is taken into consideration because the pragmatic interpretation depends on the community of practice, its culture, scope and attention.

Syntax, semantics and pragmatics of models are all important issues, and depend upon the needs of a model and its context.

Abstraction Levels of Stakeholders. At the application level, the perceptions of the users must be considered and combined with the context. At the conceptual modeling level, the resulting conceptual model must be based on what was developed at the application level. The logical level is typically based on an understanding of the platform, with the best practice being to use models that are mappings or compilations of the conceptual model.

4 Illustrative Example

A *conceptual database model* consists of a conceptual schema and a number of view schemata [11]. The view schemata are the result of transformations [1, 9] that map the viewpoints of the application level to sub-schemata of the conceptual schemata.

Consider a student-dormitory-course schema in Fig. 1. Suppose a student is enrolled in several programs at a university. The dormitory association is dependent upon the program that a student takes. Specifically, a student lives in a dormitory that corresponds to the program (business, music, etc.) in which the student is enrolled. A student might obtain some financial support from a program, depending upon the level of completion of the program. A student makes courses that are required for a given program. The credit hours assigned to a course, may vary across courses, depending upon whether the course is intended for one program, or whether it is a mandatory or elective course. Any course can only be counted one time towards one program. A student is required to take a minimum number of classes per term. If a student fails a course, then the student may retake the course, up to a maximum of three times. A course has an associated tuition fee that must be within the limits of a given term, which may vary from term to term.

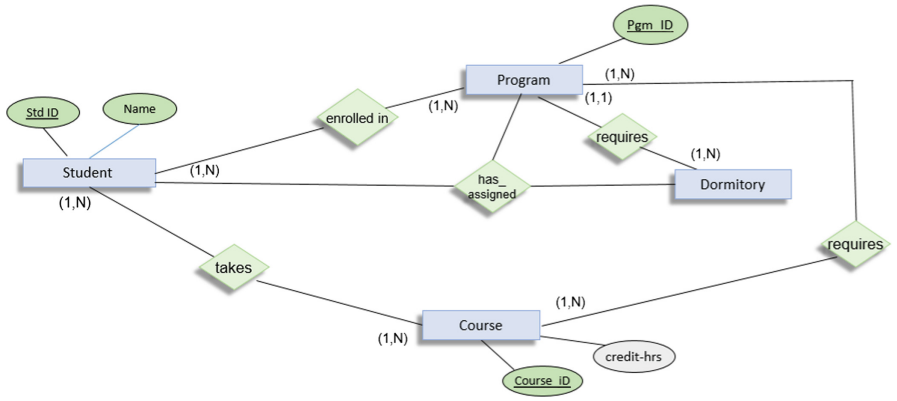

Fig. 1. Entity-relationship model of student-dormitory application

There are, however, some aspects of this situation that are difficult to model.

- A student can only take a course a maximum of three times. This might be overcome by adding a separate entity, called class or section, and a relationship; Course has Classes, with min/max cardinalities of (0, 3) from student to class.
- A course can have different credit hours depending upon the program.
- A student can have multiple majors, which requires a decision about the dormitory to which a student should be assigned.
- The normal case for *enrolled in* does not capture freshmen who are not enrolled.
- The student must take courses that are required by the program.

These problems are at the application level. Someone must represent the university situation correctly and implement the corresponding results into a database. Also, involved is the end-user, a student. The database designer must attempt to models these in one conceptual model.

5 Semiotics Reconsidered

Semantics and Pragmatics at the Application Level. Models at the application level have their own origins that they represent, profile, context, and community. The origins are consolidated perception models, enhanced by situation models that are commonly accepted in the application domain. Each community has a community-specific model; that is, a "local-as-design" approach. Objects under consideration are not homogeneous, for example, a department is considered together with its department head. Or, a student view incorporates all of the classes a student takes and refers to a university program class view from the university administration. A student is typically enrolled in one and only one program. There might be other students. Generalization and specialization follow natural semantics.

Models at this level of abstraction can be used at the conceptual level for communication and negotiation within and between communities of practice. Semantics and pragmatics differ based on the perception and understanding within the communities. Models may not be complete. Semantics may not be rigid. Objects are often considered to be holistic; for example, students together with their courses based on their programs. Therefore, we are not bound to normal data type construction. Constraints typically consider normal cases instead of extreme ones. Class planning might not require that students take classes, but student planning is based on the minimum and maximum credit hours a student must acquire in a given term.

Models at the application level have their own coherence. The underlying model allows us to integrate the different models. Models at the user level are typically not denotative but connotative, and follow cultural or community interpretations. For this reason, ontologies are appropriate for specifying domain-specific content [2].

Model Semantics at the Conceptual Level. A conceptual data model reflects, integrates and harmonizes the user views. Types specify homogeneous classes and are decomposed accordingly. The functionality definition is based on an entity-relationship algebra and given only after the structure model is complete. Constraints refine the structure; that is, semantics are defined only after the syntax is complete. The entity-relationship schema uses a diagram that is assumed to be complete, and represents its component at the same level of granularity and precision. Pragmatics tend to be hidden in a conceptual model, even though it is, in essence, an underlying model. It is assumed to be defined though external views.

Constraints at the Application Level and Conceptual Level. Constraints are generally considered valid for all of an application. However, a user's community might consider the 'normal' case or abstract (generalize) from exceptions, or omit them. Users use different scope, context, origins, and purposes. E.g., cardinality constraints represent some aspect, within specific semantics and pragmatics.

The Nature of Constraints. At the conceptual level, pragmatics must be handled by syntax and semantics. Cardinality constraints can do so, but are rigid and based on participation or lookup definition [7]. In the participation approach, extreme cases are included, in an attempt to represent exceptional cases. For example, an (1, N) constraint states that a corresponding relationship must exist for all entity classes. One solution is to use a harmonization of all user models and integrate them into the conceptual model. In this "global-as-design" approach, user views represent the external views of users, resulting in the challenge of properly representing finer semantics and pragmatics of these views. Due to the "local-as-view" design, constraints are introduced from the user's point of view. A conceptual model should harmonize all of these views to provide a holistic view of all constraints. A similar harmonization can occur at the logical level.

In Fig. 1, a freshman could be enrolled in a program or not. If the freshman is enrolled, then a dormitory can be assigned based on the program enrolled. Later the freshman might also take courses. Then, a student is either a normal student, a student who does not take courses, or a student who does not have yet a dormitory. At the logical level, we can use tables for each of these specific cases and define a view that

combines them. At the logical level, horizontal decomposition can be applied [10]. A relation type can be decomposed by selection expressions E1, …, En into separate types, provided this decomposition forms a partition on the class for this type. Therefore, we might also use a conceptual type, made up of conceptual base types. The base type has semantics without any context, but all subclasses are identified.

Objectives for Developing Better Constraints. Semantics can vary, depending on the user. This results in problems when mapping to a conceptual model, so the conceptual model should be more flexible. In most practices, normalization deals with the exceptional case where semantics causes a change of structure and the schema. That is, semantics drives syntax, in contrast to "semantics follows syntax." DBMS provide a much finer means for integrity maintenance. Maintenance can be deferred (eager or lazy integrity enforcement). Consistency can be supported at the row level. Integrity constraints can be maintained at the application level. Integrity can be made through views. Finally, flexible strategies may be used, besides the no-action and rollback approach; for example, on the basis of triggers or stored procedures.

These observations show that conceptual integrity constraints can be more elaborated if we can map the constraints to DBMS features. Here, we simply aim to show how semantics and syntax can be developed in a holistic approach. We further assume that pragmatics is defined at the application level, based on views, leading to the following observations and requirements.

(1) DBMS technology must provide a better way of treating syntax and semantics at the conceptual level, which captures pragmatics at the user level.
(2) A holistic view is needed for integrated usage of syntax together with semantics.
(3) Flexibility is required for changes needed to accommodate new technology.
(4) A mapping procedure for advanced integrity constraints should be supported.

Proposed Extensions of Integrity Constraints by Context as Part of Semantics.

1. *Actions* on a database are insert, delete and update for: a single object, one class, or objects tightly bundled via class inclusion constraints. Actions might be defined as an *action pattern*. This extends single-object actions to a complex object action while disabling the basic actions whenever a complex pattern exists.
2. The *scope pattern* is a view-defining query. This query defines either a single type view or, in general, the view schema on the conceptual schema.
3. *Enforcement style pattern* is for constraints that are timed as eager (default) or lazy (with(out) delay) enforcement, after an action (default), or as control before an action, with a level statement (e.g. DBMS, transaction, and interface levels).
4. *Reaction pattern* is for immediate enforcement or exception handling with a timed exit sub-pattern or timed enforcement, based on an enforcement obligation.

The above illustrates the need to deal with structure versus semantics. They can be formally defined and implemented. Then, in contrast to traditional approaches in which "semantics follows syntax," syntax and semantics may be treated as a whole.

Holistic View. A *conditional integrity constraint* is a pair of a context and a constraint. Constraints can be combined to partition a problem based on a scope pattern. For example, cardinality constraints Card(R, R') = (1, 1) are for R = *enrolled_in*, and

R' = *Student* with a selection predicate for: freshmen, a student who does not yet have an assigned dormitory, and students who did not yet take courses. The cardinality constraint is only valid for "normal" students. Adding an attribute *term* to the type *takes* could ensure that a student has not taken a course more than three times.

For example, for freshman with a dormitory, we may use a relaxed enforcement style. For freshman without a dormitory, we might use an interface style. That is, an insertion of such a student is only possible by an encapsulated insertion of the student, the programs, and the dormitory with a temporary insertion into the corresponding basic types; and a transfer of the object to another basic class whenever additional data are inserted. However, problems that exist or can be deduced for these constraints are not usually considered. All user needs cannot be represented by semiotics. View integration is difficult with global constraints, and usually completed based on user views. From a semiotics perspective, the user view should be considered as much as possible.

6 Conclusion

Many problems arise from the need to carry out modeling at multiple levels, depending upon the stakeholders. Since semiotics deals with language, it is proposed as an underlying basis from which to understand and capture semantics at different levels of abstraction. Additional conditional constraints are needed to model context, namely, action, scope, enforcement style and reaction.

Acknowledgements. This research was supported by the J. Mack Robinson College of Business, Georgia State University. Thanks to Melinda McDaniel for her assistance.

References

1. Embley, D.W., Mok, W.Y.: Mapping conceptual models to database schemas. In: Embley, D., Thalheim, B. (eds.) Handbook of Conceptual Modeling, pp. 123–163. Springer, Heidelberg (2011). doi:10.1007/978-3-642-15865-0_5
2. Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. **5** (2), 199–220 (1993)
3. Hartmann, S.: On the characterization and construction of entity-relationship database populations obeying cardinality constraints. Ph.D. thesis, University Rostock (1996)
4. Parsons, J., Wand, Y.: Using cognitive principles to guide classification in information systems modeling. MIS Q. **32**(4), 839–868 (2008)
5. Storey, V.C.: Relational database design based on the entity-relationship model. Data Knowl. Eng. **7**(1), 47–83 (1991)
6. Storey, V.C.: Understanding semantic relationships. VLDB J. **2**(4), 455–488 (1993)
7. Storey, V.C.: Comparing relationships in conceptual modeling: mapping to semantic classifications. IEEE Trans. Knowl. Data Eng. **17**(11), 1478–1489 (2005)
8. Schewe, K.-D., Thalheim, B.: About semantics. In: Schewe, K.-D., Thalheim, B. (eds.) SDKB 2010. LNCS, vol. 6834, pp. 1–22. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23441-5_1

9. Thalheim, B.: Entity-Relationship Modeling-Foundations of Database Technology. Springer, Berlin (2000)
10. Thalheim, B.: Syntax, semantics and pragmatics of conceptual modelling. In: Bouma, G., Ittoo, A., Métais, E., Wortmann, H. (eds.) NLDB 2012. LNCS, vol. 7337, pp. 1–10. Springer, Heidelberg (2012). doi:10.1007/978-3-642-31178-9_1
11. Thalheim, B., Tropmann-Frick, M.: Enhancing entity-relationship schemata for conceptual database structure models. In: Johannesson, P., Lee, M.L., Liddle, S.W., Opdahl, A.L., López, Ó.P. (eds.) ER 2015. LNCS, vol. 9381, pp. 603–611. Springer, Cham (2015). doi:10.1007/978-3-319-25264-3_47