

IMPROVING THE SEMANTICS OF CONCEPTUAL-MODELING GRAMMARS: A NEW PERSPECTIVE ON AN OLD PROBLEM

Completed Research Paper

Roger Clarke

School of Politics, International
Studies, and Philosophy
Queen's University Belfast
25 University Square, Belfast BT7 1PB
United Kingdom
roger.clarke@qub.ac.uk

Andrew Burton-Jones

UQ Business School
The University of Queensland
Blair Drive, St Lucia 4072
Australia
abj@business.uq.edu.au

Ron Weber

Monash University
144 Peter Road
Ruimsig, Johannesburg
South Africa
ron.weber@monash.edu

Abstract

A core activity in information systems development involves understanding the conceptual model of the domain that the information system supports. Any conceptual model is ultimately created using a conceptual-modeling (CM) grammar. Accordingly, just as high quality conceptual models facilitate high quality systems development, high quality CM grammars facilitate high quality conceptual modeling. This paper seeks to provide a new perspective on improving the quality of CM grammar semantics. For the past twenty years, the leading approach to this topic has drawn on ontological theory. However, the ontological approach captures just half of the story. It needs to be coupled with a logical approach. We show how ontological quality and logical quality interrelate and we outline three contributions of a logical approach: the ability to see familiar conceptual-modeling problems in simpler ways, the illumination of new problems, and the ability to prove the benefit of modifying CM grammars.

Keywords: Conceptual modeling, Logic, Semantics, Ontology

Introduction

Underlying any information system (IS) is a model of the domain that the information system is intended to support. Such models are often called *conceptual* models, because they focus on the way domain phenomena are to be *conceived* independent of the technology used to represent them. Sometimes conceptual models are made explicit in the form of graphical models during systems development; sometimes they remain implicit. Either way, their importance has long been recognized, from early research on databases (Chen, 1976; Kent, 2000) and systems analysis (Yourdon, 1989) to recent research on model-driven development (Selic, 2003; Utting and Legeard, 2007), configurable packages (Scheer and Habermann, 2000), and user-generated content (Lukyanenko and Parsons, 2011). Because conceptual models are often created early in the systems development process, errors in them can propagate to later phases of systems development where they are much more costly to fix (Boehm, 1981; Moody and Shanks, 2003). As a result, evaluating the quality of conceptual models has long held a central place in IS research (Iivari et al., 2006), practice (Fettke, 2009), and pedagogy (Ginzberg, 2012).

Evaluating the quality of a conceptual model is not a straightforward task. A conceptual model manifests an IS professional's understanding of users' conceptions of the domain in which they work. This understanding is a social construct; it is often the outcome of discourse and negotiation that occur among multiple stakeholders (Hirschheim et al., 1995). Moreover, this understanding is fluid; it may change as stakeholders engage with the domain and each other. Because of these social, fluid properties, researchers have struggled to develop appropriate theoretical perspectives that can be used to guide the evaluation of conceptual models. As Moody (2005, p. 243) observes, "conceptual models continue to be evaluated in ... an ad hoc way, based on common sense, subjective opinions, and experience." Similarly, Siau and Rossi (2011, p. 249) argue, "the blooming production of modeling methods is not the problem, the lack of standardized techniques for evaluating them is."

Our research aims to address the problem of the quality of conceptual models by extending prior theoretical work on the quality of conceptual-modeling (CM) *grammars*. This perspective is critical but surprisingly under-developed. We focus on two attributes of quality relating to CM grammars, which we call "ontological quality" and "logical quality." Ontological quality has been examined at some length in the conceptual modeling literature, and it has provided valuable insights. Logical quality has not been examined in the conceptual modeling literature, however, even though it too can provide valuable insights. We explain why both types of quality are required for a full understanding of the semantics of CM grammars. We explain their meaning, discuss their importance and how they relate, and illustrate the benefit of this dual view for the evaluation and design of CM grammars.

Our paper proceeds as follows. To situate our work, we first introduce and clarify important concepts used in the evaluation of conceptual models and some key frameworks proposed in past research. In the following section, we introduce and explain the notions of "ontological quality" and "logical quality." Because ontological quality has been discussed already in the literature, we focus on "logical quality." Specifically, we operationalize logical quality via the dimension of *completeness*. We subsequently discuss the benefits of considering this dimension over and above existing ontological perspectives. We conclude by examining the implications of taking this more-comprehensive (ontological and logical) perspective on the quality of conceptual models for future research and practice.

Background

Providing a concise summary of prior research on the quality of conceptual models is difficult, because the field is large and diverse. In this regard, Moody (2005, p. 247) writes, "over 50 different proposals have been published and new ones [are] published every year." Because the focus of our paper is on a *theoretical* evaluation of conceptual models, one way to categorize prior literature is into theoretical evaluations versus atheoretical evaluations. The literature describing theoretical evaluations of conceptual models can then be categorized using Wand and Weber's (2002) framework and Lindland et al.'s (1994) framework. Table 1 shows such a categorization.

		Dimension of representation (per Lindland et al. 1994) ¹		
		Syntax	Semantics	Pragmatics
Aspect of conceptual modeling process (per Wand and Weber 2002) ¹	Method	General principles for syntax, semantics, and pragmatics: (Frederiks and van der Weide 2006; Nelson and Monarchi 2007)		
			Semantics (mapping): (Evermann and Wand 2005)	
	Grammar	Physics of notations: (Moody 2009)	Semantics (mapping): (Wand and Weber 1993; Wand 1996; Shanks et al. 2003; Harel and Rumpe 2004; Evermann and Wand 2005; Guizzardi 2007)	Task and cognitive fit: (Agarwal et al. 1996; Aguirre-Urreta and Marakas 2008)
	Script	General principles for syntax, semantics, and pragmatics: (Lindland et al. 1994; Schuette 1999; Krogstie et al. 2006)		
		Physics of notations: (Moody 2009)	Semantics (enactment) and pragmatics: (Lyytinen 1987; Eriksson and Ågerfalk 2005)	
			Semantics (mapping): (Harel and Rumpe 2004; Shanks et al. 2003)	Cognition and complexity: (Agarwal et al. 2000; Gemino and Wand 2003; Genero et al. 2008; Maes and Poels 2007; Parsons and Wand 2008; Rockwell and Bajaj 2004; Siau and Tan 2005; Stark and Esswein 2012; Teo et al. 2006)
Table 1. Prior work				

The vertical axis of Table 1 reflects three dimensions of conceptual modeling proposed by Wand and Weber (2002): script, grammar, and method. Specifically, a conceptual model can be conceived as a particular kind of *script* generated using a CM *grammar*. A conceptual modeller employs a CM *method* in conjunction with a CM grammar to generate a script that represents a domain. A CM method might cover a number of aspects of the modeling process—such as how to model a domain, how to engage with stakeholders to elicit their perceptions of a domain, how to resolve disputes among stakeholders about the underlying semantics of a domain, and so on. The quality of a conceptual model can therefore be assessed from each of these perspectives (script, grammar, and method).

The horizontal axis of Table 1 reflects the three dimensions of conceptual modeling proposed by Lindland

¹ In addition to *method*, *grammar*, and *script*, Wand and Weber (2002) included *context*. We do not show it because it is captured by the column for pragmatics, i.e., what is practical in a given context (per Burton-Jones et al. 2009). Similarly, Lindland's et al. framework of *syntax*, *semantics*, and *pragmatics* was supplemented with social and physical by Krogstie et al. (2006). We exclude these because they have rarely been studied and are quite distant from our paper's focus on semantics.

et al. (1994). In principle, a conceptual model should be evaluated according to its “totality of features and characteristics” (Moody, 2005, p. 252), but the wide variety of such features and characteristics makes this difficult to achieve in practice. By concentrating on the linguistic aspects of conceptual models, Lindland et al. (1994) were able to propose a parsimonious set of three dimensions: (a) *syntactic quality*—the extent to which a conceptual model conforms to the constructs and rules of the grammar used to express it; (b) *semantic quality*—the extent to which a conceptual model captures the perceived meaning of the domain; and (c) *pragmatic quality*—the extent to which stakeholders find a conceptual model easy to use and useful. Their framework has since been used widely (e.g., Maes and Poels 2007; Moody and Shanks 2003; Siau and Tan 2005).

The cells of Table 1 show exemplary papers of each major type. As the table shows, some papers have provided principles that cover all three dimensions of representation (syntax, semantics, and pragmatics), with a focus on either *methods* (Frederiks and van der Weide, 2006; Nelson and Monarchi, 2007) or *scripts* (Krogstie et al., 2006; Lindland et al., 1994; Schuette, 1999). In contrast, other papers have focused on just one or two cells of Table 1. For instance, Moody's (2009) work on the physics of notations addresses the syntax of grammars and scripts, whereas Wand and Weber's (1993) work on ontology focuses on the semantics of CM grammars alone.

Two trends in Table 1 are particularly relevant to note. First, prior literature has focused predominantly on assessing (a) *pragmatics* over syntax and semantics, and (b) *scripts* over grammars and methods. Second, research on semantics has taken two perspectives: a “mapping” perspective, and an “enactment” perspective. The “mapping” perspective views semantics as a mapping from “one domain (the universe of discourse) to another (the model)” (Eriksson and Ågerfalk, 2005, p. 201). In contrast, the “enactment” perspective views semantics in terms of action—what people *do* with words (Austin, 1962). In Table 1, we show entries for the mapping perspective in the “semantics” column, whereas we show entries for the enactment perspective in the “semantics” and “pragmatics” columns.

We highlight these two traditions on semantics because (a) each offers value, and (b) each is based on a different philosophical tradition that is somewhat incommensurable (Lyytinen, 1987; Wand et al. 1995). In this study, we take the mapping perspective only, because we see several opportunities to extend prior work that adopts this perspective. We leave extensions of the enactment perspective to future research.

Although all cells of Table 1 reflect important domains of research, we focus on just one—namely, the middle cell of Table 1, which covers the semantics of CM grammars. We chose this cell for four reasons. First, the *raison d'être* of conceptual models is to convey a domain's semantics (at least according to the “mapping” view), so semantics is arguably its most-critical dimension. Second, a CM grammar often *precedes* a CM method and script—systems analysts first choose a grammar and then employ a method to create a script using that grammar. Third, research on the semantics of CM grammars can be used to offer rules and guidelines (methods) for modeling (e.g., Evermann and Wand, 2005). Thus, advancing research on the semantics of grammars has benefits for research on methods too. Finally, recent research has emphasized the need to clarify the semantics of CM grammars (e.g., Harel and Rumpe 2004).

As Table 1 shows, several papers have proposed ways to evaluate the semantics of CM grammars. The predominant approach is to use theories of ontology (see, e.g., the work of Wand, Weber, and Guizzardi in Table 1). Since the ontological approach was proposed (Wand and Weber, 1993), it has been used widely. For instance, according to Moody (2009, p. 774) “ontological analysis is a widely accepted approach for analyzing semantics of notations, which supports rigorous, construct-by-construct analysis.” Despite its maturity, we show in this paper that it requires significant extension. Indeed, we show it addresses only half the problem of the quality of CM grammars. The extension we provide stems from an analysis of the *logical* quality of language semantics and its relationship to ontological quality. Some papers have alluded to logic when considering the semantics of CM grammars (Carasik et al., 1990; Guizzardi, 2007; Harel and Rumpe, 2004; Lyytinen, 1987), but we are aware of no detailed treatment.

The field of computer science has a long history of using formal logic to provide the semantic underpinnings of formal languages. Nonetheless, this work has quite different aims from our work, because it focuses on implementation-oriented issues (such as computation and automated reasoning). For instance, Calvanese et al. (1998) use Description Logic (DL) to give semantics for entity-relationship diagrams and object-oriented data models. In part, their aims are to compare these two grammars and to automate reasoning. More recently, Fillotrani et al. (2012) present a DL-enabled CASE tool with similar

aims. They provide an automated-reasoning engine to identify errors during the design of CM scripts. Much recent research has also investigated the design and use of ontologies on the Web. For instance, Franconi (2004, p. 76) writes that OWL Full, which is a web-based ontology language, “turned out to be very expressive—beyond first-order logic. This means that [it] can express most of the details we’d want described in an ontology, but other agents can still understand that ontology without any problems.”

In sum, our predecessors who have used the logical approach have been interested in automating the task of determining whether a particular conceptual model entails a certain conclusion (for a particular model expressed in a particular grammar). In contrast, our aim is to discover those properties of CM grammars that support production of CM scripts with high-quality semantics. Computational and automated-reasoning concerns are not our focus.² Nonetheless, despite the wide gulf between current ontological research and logical research on the semantics of CM grammars, we believe a logical approach to the evaluation of CM grammars offers important benefits. In this regard, we argue below that a joint consideration of ontology and logic is not only useful but essential.

Semantic Quality: Ontological and Logical Perspectives

Formal languages, such as CM grammars, have two parts to their syntax: a vocabulary of symbols (“terminal symbols”), and a set of production rules for combining those symbols into well-formed expressions. Corresponding to these two parts of a formal language’s syntax are two parts of a formal language’s semantics: first, a mapping from the terminal symbols to their referents; second, a recursive definition of the referent (meaning) of complex expressions in terms of the meanings of their components. These two parts can be seen as describing the respective contributions of the language’s vocabulary and its production rules to the meaning of a given well-formed expression. A complete assessment of the quality of a formal language’s semantics, therefore, must consider both elements of its semantics—vocabulary and production rules. We argue in this section that a grammar’s ontological quality depends only on the former aspect of its semantics, its vocabulary, and the interpretation thereof. We therefore develop a notion of logical quality that turns on the remaining aspect of a grammar’s semantics, its production rules, and their contribution to script meaning.

Wand and Weber’s (1993) Theory of Ontological Expressiveness (TOE) assesses CM grammars in terms of a mapping between constructs of the grammar and constructs of a benchmark ontological theory. The mapping in question is semantic: it is a mapping from grammatical constructs to their referents. A grammar is defective, according to TOE, if the semantic mapping is not one-to-one and onto. Each construct in the grammar must be mapped to exactly one construct in the benchmark ontology. Furthermore, every construct in the ontology must have exactly one construct in the grammar mapped to it. If the mapping is one-to-one, then the grammar is considered ontologically clear; if the mapping is onto, then the grammar is considered ontologically complete. As an immediate consequence of this characterization of ontological quality following TOE, we can see that ontological analysis of a CM grammar is blind to its production rules. All that matters in determining whether a grammar is ontologically clear and complete is the grammar’s vocabulary and the semantic values of the vocabulary elements. Production rules have no impact on the mapping between grammatical constructs and ontological constructs.

Ontological quality is a matter of comparison between a grammar and a benchmark ontological theory (e.g., Bunge’s 1977 theory). A theory of ontology tells us the fundamental types of phenomena that make up the real world. A grammar of high ontological quality will clearly and completely reflect this theory in its vocabulary of constructs—every type of phenomenon in the world will be represented by exactly one construct in the grammar. Similarly, logical quality, as we define it, will be a matter of comparison between a grammar and a benchmark logic. A logic tells us how logically complex propositions can be constructed from simple ones and from sub-propositional elements (objects, properties, relations, etc.). It will also tell us which of these propositions entail which other ones.

² More formally, we seek CM grammars all of whose scripts have desirable sets of entailments. This desideratum is independent of the desideratum in prior computer science research that we should be able to efficiently compute the model’s entailments.

A logic gives us a list of ways the world could possibly be. Given a set of objects, properties, and relations, a logic tells us how they can and cannot be combined in the world. If our ontology contains *employees* as a category of objects, and *being male* and *being female* as properties, then our logic might tell us that the world could contain (1) only male employees, or (2) only female employees, or (3) both male and female employees, but that it is not possible for the world to be such that both (1) and (2) are true. A logic tells us how different ways of combining objects, properties, and relations relate to each other. *Every employee has a salary* entails *every female employee has a salary*. The latter deliverances of our logical theory—relations of entailment or consequence—are nonetheless formal. On a standard understanding of what it means for logic to be formal (Beall and Restall, 2005), this means that whether one sentence entails another does not depend on the content of the particular terms supplied by the ontology. If we were to systematically replace the terms *employee*, *female*, and *has a salary* throughout the two propositions just stated, the relation of entailment would still hold: *every φ has ψ* entails *every φ who has χ also has ψ* , regardless of which predicates we substitute for φ , ψ , and χ . Because logic is formal in this way, logical analysis is as blind to a grammar's vocabulary of constructs as ontological analysis is blind to a grammar's production rules.

So far, we have an idea of what kind of structure a benchmark logical theory provides. Now we must show how to leverage that sort of theory into an account of a CM grammar's logical quality, just as TOE showed us how to leverage a benchmark ontological theory into an account of a CM grammar's ontological quality. We take a script-driven approach to analysis of grammars, based on the idea that a high-quality grammar is one that yields high-quality scripts.

As we indicated previously, a benchmark logic gives us a list of ways the world could possibly be, given a vocabulary of constructs. We can understand this list in terms of a set of questions that can be asked about the world: for any way φ that the logic tells us the world could possibly be, we can ask the question, "Is φ an accurate description of the world?" In other words, we can ask, "Is φ true?" As we argue in the next section, a high-quality script should answer all relevant questions about the domain being modeled (i.e., either "yes" or "no"); therefore, to assess a script's logical quality, we compare the list of questions answered by the script with the list of questions produced by our benchmark logic. A script of high logical quality answers all the questions on an appropriately formed list and answers them coherently. Correspondingly, a grammar of high logical quality produces only scripts of high logical quality.

In the next section, we make these general ideas more precise. We will use a notion of *information loss* to give conditions scripts and grammars must meet to exhibit high logical quality. These conditions turn on the factors just mentioned—namely, a comparison between a list of questions generated by a benchmark logic and a corresponding list of answers generated by a script (or by the scripts a grammar produces). As we will show subsequently, attending to this dimension of quality has three primary benefits:

1. It illuminates an existing debate in conceptual modeling, over optionality.
2. It surfaces new problems, analogous to optionality, hitherto unrecognized.
3. It points to guidelines for the design of CM grammars so as to avoid all problems of information loss.

Benefit 3 is where our formal results come into play. Here is a brief map of what is to come and how our formal results fit into the picture. First, we give a formal analysis of information loss via the logical notion of *(in)completeness*. We then identify some sources of information loss, both familiar and new. Importantly, information loss has been suggested as a reason why optionality is to be proscribed in conceptual models, but the notion was only partially understood (an inevitable consequence of using ontological analysis for a task better suited to logical analysis). We devote a subsection to clarifying and simplifying the debate over optionality. On the other hand, in another subsection, we reveal that another feature of conceptual models—namely, the standard way of representing mandatory associations between classes—leads to the same sort of information loss as optionality, though this outcome has not yet been appreciated in the literature. By addressing an explicit list of issues,³ our formal results (discussed in a later section) then show an ERM-type grammar can eliminate all sources of losses of information. These

³ For reasons of space, we do not present the full list. Nevertheless, the issues we discuss in this section (concerning optionality and cardinalities for mandatory associations) are representative.

results naturally lead to guidelines for constructing grammars of high logical quality. Our primary purpose in this paper is to discuss the significance and applicability of our formal results. In this light, we suppress details of the proof.

Logical Quality: Information Loss and Completeness

Inaccuracy versus Information Loss in Scripts

A CM script aims to answer questions about the domain being modeled. It tells the user what the domain is like. One way a script can be flawed, then, is to give the wrong answer to a question. If a script says, for example, that some research assistants have no supervisor, when in fact all research assistants in the domain must have supervisors, the script is flawed because it is inaccurate. Identifying instances of inaccuracy generally requires some knowledge of the domain: we need to know both what the script says and what the truth is, so that we can identify cases where these two things are different.

Scripts can have another sort of flaw, which is our concern here. Rather than giving the wrong answer to a user's question, *a script might give no answer*. If a script is ambiguous between saying that all research assistants must have a supervisor and saying that some research assistants may lack a supervisor, then the script fails to answer the following question: *Must all research assistants have a supervisor?* The script has lost information about the domain, namely the answer to this question. We can identify instances of information loss without any prior knowledge of the domain. No matter which answer to the question is the correct one, we only need to know how to read our script to know whether it provides any answer.

We call a script that provides answers to all of a user's questions a *complete*⁴ script. We call a script *incomplete* otherwise. An incomplete script, we say, *loses information* about the domain modeled. An incomplete script is not one that falsely represents the domain (one that misinforms). Instead, it fails to inform users about some aspect of the domain. Complete scripts show no loss of information. Completeness is perfectly suited to study via logical analysis. Moreover, it is likely to be overlooked by ontological analysis, because of the feature noted above. We can determine whether a script is complete (in the sense defined here) without any prior knowledge of the domain—without performing any ontological analysis.

Ensuring scripts are complete is particularly important in light of what we know about the practice of conceptual modeling. Specifically, systems analysts often: (1) fail to elicit a complete set of semantics from users—they stop their elicitation work too soon (Pitts and Browne, 2004), (2) fail to verify scripts carefully with users (Wastell, 1996), (3) make incorrect assumptions about domain semantics, because they assume (incorrectly) that the domain is similar to other ones they have seen (Hadar et al., in press), and (4) base the design of information systems on conceptual models that have not been validated (Dawson and Swatman, 1999). For these reasons, systems analysts most likely will either use incomplete scripts to construct the information system or will complete the script themselves (possibly incorrectly) without validating it with users. Although a grammar cannot solve these human/social problems, the problems would be alleviated to some extent if we could guarantee the grammar would produce only complete scripts. Such scripts might allow systems analysts and users to better pinpoint where a script is defective because it does not model all the phenomena that are of interest to users or it models these phenomena incorrectly.

The definition of completeness we have given, therefore, needs development. Most importantly, what do we mean by “*all of a user's questions*”? Of course, we would not expect a conceptual model of one business to provide an answer to questions about a completely separate business. It would be a mistake to label a script incomplete for failing to characterize the workings of an organization it was never intended to characterize. This observation points to a basic restriction on the kind of questions we expect a script to

⁴The term “complete” is used in many distinct ways in literatures relevant to the present work. For example, Wand and Weber's (1993) TOE contains a notion of ontological completeness distinct from the logical notion introduced here. Our use of the term is inspired by, and most closely analogous to, the notion of a complete theory in first-order logic. We explain the notion of a complete theory on page 13.

answer—specifically, the question must be about the domain that has been modeled. Questions about a certain hospital are relevant in assessing a script intended to model that hospital, but questions about a manufacturing company in another country are not relevant.

This restriction does not go far enough. There are even questions about the domain that has been modeled that we would not reasonably expect a script to answer. Continuing with our example, our model of the hospital should not be expected to answer questions about, say, surgeons' shoe sizes (unless, of course, for some reason surgeons' shoe sizes were relevant to questions that stakeholders might ask about the hospital domain). In short, the questions must be framed only in terms of phenomena that have been modeled.

The point is general: a central task in conceptual modeling—perhaps *the* central task—is separating relevant from irrelevant information. We produce models of the world, abstractions from the world, in part because the world itself is “noisy” in an informational sense. Surgeons encode information about their shoe sizes by possessing particular shoes; they also encode information about their areas of specialization by performing specialized surgery. The latter bit of information is important to the hospital, while the former usually is not.

In this light, we amend our initial definition of an incomplete script and say: a script is incomplete if and only if it fails to provide an answer to any question that can be asked about the phenomena that have been modeled. Defining completeness in terms of the “modeled domain phenomena,” as we have done here, is consistent with prior conceptual modeling research, which has used a range of terms for this domain including the “domain of change,” “target system,” and “universe of discourse” (Hirschheim et al., 1995, p. 10). We next provide a characterization of relevant questions, which we will later formalize.

Relevance and Completeness

A modeler's task, then, centrally involves sorting relevant from irrelevant information. We see this task as prior to the construction of a concrete script and prior to selection of a modeling language with which to produce such a script. Briefly, we have in mind a picture of conceptual modeling as a five-step process.

1. The modeler examines the domain to determine which features of the domain (which entities, properties, etc.) are relevant and which are irrelevant.
2. The modeler produces or selects a basic vocabulary of symbols to represent the relevant features.
3. The modeler determines how those features are related in the domain (in the real world).
4. The modeler chooses a grammar appropriate to representing the relations between relevant domain features.
5. The modeler produces a script that reflects the relevant features of the domain and their interrelations, using the selected vocabulary, put together according to the rules of the selected grammar.

To clarify, we see these as the *logical* steps involved in conceptual modeling but not necessarily as *chronological* steps. That is, we do not pretend that standard modeling practice involves explicitly following these steps in this order, nor even that these steps are performed distinctly from one another. Rather, we think that modelers must perform all these tasks in the process of producing a script.

This picture gives us a way of characterizing relevant and irrelevant questions about the domain. Relevant questions, the ones that a script must answer on pain of incompleteness, are questions about the relations between the relevant features of the domain as identified in the first and third steps. If the modeler thinks that treatment types and disease types are features of the hospital relevant to the script's purpose, then the script should provide an answer to questions about whether mental health conditions may be treated with surgery, whether a single condition may be treated with both surgery and medication, and so on. If the script in this scenario fails to provide answers to these questions, it is justifiably criticized as incomplete.

Completeness in Grammars

Completeness and incompleteness are properties of scripts. Our concern here, however, is not to give guidelines for producing high-quality scripts in arbitrary modeling grammars. Our concern is, instead, to give guidelines for producing modeling grammars that facilitate production of high-quality scripts. Therefore, we introduce a related property at the level of grammars. We say a grammar is *always-complete* if the grammar does not allow production of incomplete scripts, or, equivalently, if it only allows production of complete scripts.

At this point, one might well wonder whether completeness (of scripts) and always-completeness (of grammars) are reasonable goals. That is, they might be desirable properties but unattainable. This outcome might be the case in general, but we have some optimistic results to present here. We show below that, for an important fragment of standard CM grammars such as the entity-relationship modeling (ERM) grammar, with minor amendments to standard usage, we can produce an always-complete grammar. The fragment in question ignores cardinalities of associations (except for the new type of cardinality notation introduced in Table 2), ternary and other greater-than-binary relations, and numerical properties (e.g., salary or mass, as opposed to dichotomic or boolean properties such as pregnancy or marital status). Even if always-completeness remains elusive for the full ERM grammar with cardinalities, ternary relations, and numerical properties represented, users of the full grammar would be better off following our recommendations for the fragment isolated here. That is, it may be difficult to produce a grammar that ensures questions about numerical properties are always answered, but it is easy to ensure that questions about non-numerical properties are always answered—and, *ceteris paribus*, it is better to have as many questions answered by a script as possible.

In the following section, we discuss applications of our logical analysis of the ERM grammar to existing debates in the conceptual modeling literature. This provides a natural introduction to the modifications to the ERM grammar required for our formal results. After discussing these applications, we will provide a rigorous definition of always-completeness, and state our central formal result—namely, that our modified fragment of ERM is always-complete.

Applications: Optionality and Mandatory Associations

Before we discuss our formal results and the lessons to be drawn therefrom, consider two problems in conceptual modeling, both sources of loss of information. The first, which concerns optional attributes and associations, is an established debate that logical analysis helps us illuminate. The second, which concerns an ambiguity in mandatory associations between classes, is a new problem that logical analysis helps us unearth. In the following two subsections, we discuss each problem in turn, along with our recommended solutions to the problems, in the form of modifications to the standard ERM grammar's syntax. The upshot of our discussion in this section can be summarized as follows. Loss of information has previously been identified as a problem in conceptual modeling, specifically in connection with optionality. Nonetheless, as we argue in the first subsection, previous ontological analyses of the problems due to optionality have been unable to properly characterize this loss of information, which deficiency is addressed by our logical analysis. Applying our logical characterization of information loss from the previous section turns up further sources of loss of information (aside from optionality), notably including the problem with mandatory associations discussed in the second subsection. Furthermore, our logical analysis lets us demonstrate rigorously that the *only* sources of loss of information in our target fragment of the ERM grammar are those identified in this section and in our discussion of formal results in the next section.

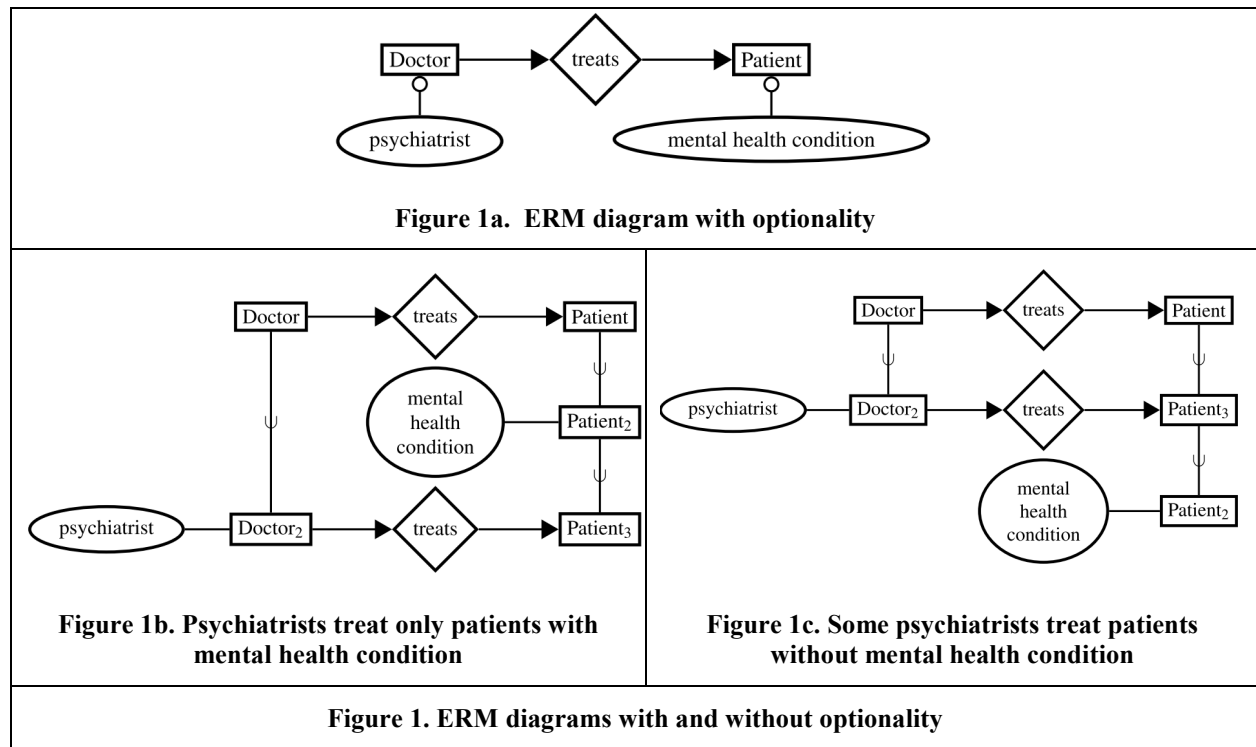
Optional Attributes and Associations

Several researchers have cautioned against using optional attributes and associations in conceptual models. Weber and Zhang (1996, p. 158) and Wand et al. (1999, p. 512) were perhaps the first to suggest that optional attributes and associations obfuscate domain semantics. Wand et al. (1999, p. 518) suggest that difficulties with optionality might arise because information about the “laws” that cover the properties of things is lost when optional attributes and associations are used. Burton-Jones et al. (2012) take this proposition as a starting point for ontological analysis of the difficulties with optionality, using

Bunge's (1977) definition of laws. Their ontological analysis finds that there are some cases where optionality leads to loss of information that cannot be categorized as information about laws. Specifically, on Bunge's definition of laws, when two classes are mutually exclusive, no lawful relation exists between the two. Nonetheless, the information that two classes are exclusive can be obscured through the use of optionality. Thus, the problem of information loss due to use of optional attributes is more general than the ontological category of laws. In short, a clear solution to this problem still eludes ontological analysis.

Logical analysis lets us more precisely characterize the problem with optionality, through the notion of completeness as described above. When a grammar allows optional attributes or associations, it cannot be always-complete. Note that the problem identified here is at the level of grammars, not scripts. As Burton-Jones et al. (2012) found, scripts with only one instance of an optional attribute or association do not suffer from loss of information. That is, some scripts containing optional attributes or associations are complete. Nonetheless, a grammar that includes optional attributes or associations will allow incomplete scripts.


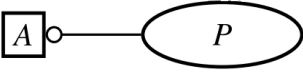
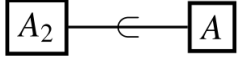

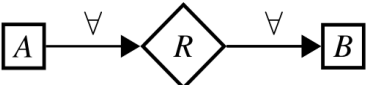
To illustrate why (but not to prove that) optionality prevents always-completeness, consider Figure 1a.⁵ This diagram shows a simple script with two optional properties, *mental health condition* and *psychiatrist*, belonging respectively to the classes *Patient* and *Doctor*, which are associated via the (non-optional) association *treats*. According to Figure 1a, all doctors treat at least one patient, all patients are treated by at least one doctor, some (but not all) patients have a mental health condition, and some (but not all) doctors are psychiatrists. Now here is a question to which the diagram provides no answer: do any psychiatrists treat patients without a mental health condition?



To see that Figure 1a does not provide an answer to this question, compare Figures 1b and 1c. Here we have two ERM diagrams without optionality (Figures 1b and 1c) that tell mutually inconsistent stories about the domain. Nonetheless, both diagrams are consistent with Figure 1a. Not only are the two diagrams inconsistent with each other, but they disagree over the answer to our question. In Figure 1b, psychiatrists only treat patients with a mental health condition. In Figure 1c, however, psychiatrists may treat patients without a mental health condition. Because these two diagrams disagree on the answer to

⁵ Our ER diagrams use standard notation, with minor exceptions noted in Table 2.

our question, but are consistent with Figure 1a, it follows that Figure 1a does not answer our question.

	Arrows indicate direction of relation (<i>A</i> bears <i>R</i> to <i>B</i>)
	Small circle indicates optionality (not all members of <i>A</i> have property <i>P</i>)
	Line with \subset indicates subset (all members of <i>A</i> ₂ are members of <i>A</i>)
	Cardinality notation (see section on Mandatory Associations)
	Cardinality notation (see section on Mandatory Associations)
Table 2. Legend for non-standard ERM notation	

Multiple possible solutions exist to the problems generated by use of optional attributes and associations. We could include a limit on the number of optional attributes/associations in a single script or attached to a single class. This solution would be ungainly, however, so we recommend instead a general proscription against optionality. Our official modification of the ERM grammar will not allow optional attributes or associations.

Mandatory Associations

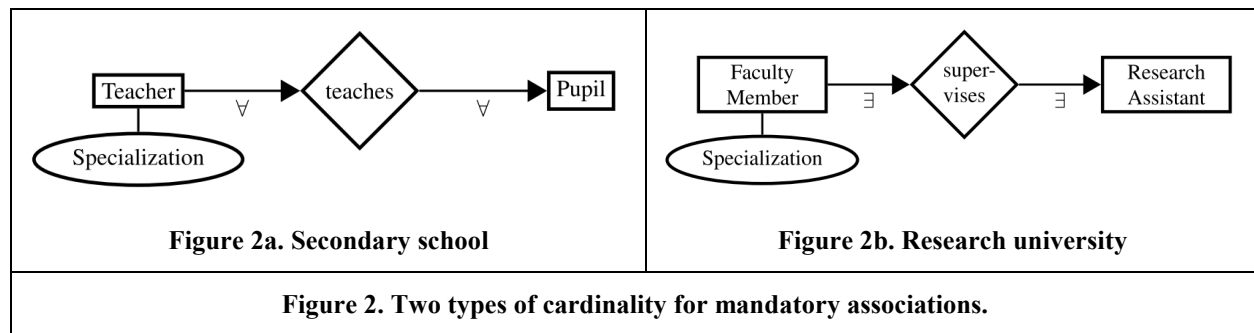
We now turn to the standard way of representing mandatory associations between classes, which fails to distinguish logically distinct ways that two classes can be related. Consider the following two situations. First, we have a small secondary school with one class of pupils in each year. We have, then, a class of teachers, each with a different specialization (science, history, mathematics, etc.), and we have a class of pupils. The teachers teach the pupils; this association between the two classes is mandatory. The second situation to consider is a research university. We have a class of faculty members, each with a different specialization, and a class of research assistants. The faculty members supervise the research assistants; we may suppose this association between the two classes to be mandatory (no faculty members lack research assistant supervisees).

In our examples, two different kinds of mandatory association exist between pairs of classes. In both cases, all members of a class *A* (teachers/faculty) bear a relation *R* (teach/supervise) to at least one member of a class *B* (pupils/research assistants). Furthermore, all members of the class *B* bear *R*-inverse (taught by/supervised by) to at least one member of *A*. In the secondary school situation only, however, it is also the case that all teachers teach *all* pupils: the pupils must take science *and* history *and* mathematics, and so on. Research assistants, on the other hand, need only be supervised by faculty in their home discipline and specialization. A research assistant must assist some principal investigator but not all of them. We do not expect to find the same research assistant supervised by a literary theorist and a biologist.

Our modeling grammar must be sensitive to the kind of difference identified here if it is to produce complete scripts. Therefore, we introduce a new type of cardinality notation. When class *A* is linked to class *B* via the relation *R*, we use the annotation “ \forall ” (all) at both ends of the link to indicate that all members of *A* bear *R* to *all* members of *B*, and vice versa. In a relational database, this constraint must be enforced via the instances of the corresponding relation that occur in the database. Specifically, the instances of the relation must show that *all* members of one set are related to *all* members of the other set.

Likewise, we use the annotation “ \exists ” (some) to indicate that all members of A bear R to *at least one* member of B , and vice versa (see Figure 2 for an illustration). In a relational database, this constraint again must be enforced via the instances of the corresponding relation that occur in the database. Specifically, the instances of the relation must show that *all* members of one set are related to *at least one* member of the other set. For convenience, we will sometimes refer to associations with the notation “ \forall ” as \forall -type associations, and associations with the notation “ \exists ” as \exists -type associations.

Note that restoring standard cardinality notation to our fragment of the ERM grammar, *by itself*, does not suffice to avoid the kind of ambiguity we resolve here. Standard cardinality notation allows us to say that members of class A bear R to at least one member of B by giving a cardinality of $1..n$ to the appropriate side of the association, where n is the cardinality of the class B . Likewise, we can say that members of A bear R to all members of B by giving a cardinality of n to the appropriate side of the association. To use this technique to resolve the ambiguity, however, would require specifying in advance the cardinality of the classes A and B , which in general is not possible. Without such a specification, we cannot choose a number n such that bearing R to n members of class B entails bearing R to all members of class B . Therefore, to use standard cardinality notation to resolve the ambiguity in question, we would have to add to the ERM grammar a notation for the cardinality of a class. Rather than writing a numeral corresponding to the value of n known to be the cardinality of the class B , we would need to write “ $c(B)$,” where $c(\cdot)$ is a function giving the cardinality of the class. This is equivalent to our technique of using the notation “ \forall .”



Formal Results – Proving the Completeness of an ERM Grammar

This section contextualizes and explains the significance of a formal result whose proof is available from the authors. This section presumes no technical proficiency in formal logic. Note that our formal results are part of a project of *explication* (cf. Carnap 1947, pp. 8--9). That is, our aim is to study a certain informally understood property of CM scripts (information loss, as discussed above), but we proceed by proving facts about formally defined properties of certain logical constructs. The logical constructs and their formally defined properties serve as stand-ins for the phenomena of primary interest to us. This substitution has the advantage that formally defined constructs and properties are susceptible to rigorous proof.

To understand our formal results, we need some concepts from first-order logic (FOL). Centrally, we need the notion of a *first-order theory*, which is the FOL-analogue of a CM script. Formally, a first-order theory is a set of formulas (roughly, sentences) of FOL, which is closed under entailment. We call these formulas the theorems of the theory. Where φ is a theorem of a theory T , we write $T \vdash \varphi$. When we say that theories are closed under entailment, we mean that if $T \vdash \varphi_1, \dots, \varphi_n$, and ψ is entailed by $\varphi_1, \dots, \varphi_n$, then $T \vdash \psi$. Some theories can be axiomatized. That is, we can sometimes isolate a small set of formulas Γ such that for any formula φ , $T \vdash \varphi$ if and only if φ is entailed by Γ . In our official semantics for ERM-R (our modified ERM grammar), a script is systematically translated into a set of axioms for a first-order theory. Anything that is a theorem of this theory is said to be entailed by the script.

We also need the concept of a *first-order language*. One might think of FOL as a language, and indeed some writers speak of “the language of FOL.” FOL is better thought of as a family of languages, or as a recipe for producing a language from a set of symbols of certain types. A first-order language, then, is a

formal language belonging to the FOL family of languages, produced from a set of symbols according to the FOL recipe. The symbols needed to specify a first-order language are of two kinds: *function symbols*, and *predicates*. A function symbol is to be interpreted as referring to a function from objects to objects, and a predicate as referring to a function from objects to truth values (a relation, or property). As a special case, we interpret a 0-ary function symbol as referring to an object. Given a stock of function symbols and predicates (which we refer to collectively as *non-logical symbols*), FOL gives us a set of well-formed formulas and entailment relations among those formulas: this is a first-order language.

For example, suppose we are given symbols to refer to Alice and Bob, to the properties of being a student and of being an instructor, and to the relation of teaching. In that case, FOL yields a language that lets us say the following: Alice is an instructor and Bob is a student; there are exactly three instructors; all instructors teach some but not all of the students; and so on. There are some things we would not be able to say in this formal language, such as the following: Alice and Bob have different jobs; or Carol is a student. The language simply lacks any symbols referring to Carol or to jobs. For a different reason, the language does not allow us to say, for example, that some students love only each other. This requires a second-order logic. Still, FOL is powerful enough for most applications, and certainly powerful enough for ours. In our application of FOL, we only consider first-order languages with predicates of arity 1 or 2, and without function symbols. Moreover, we focus on FO², the two-variable fragment of FOL (without identity). The formulas of FO² are just the formulas of FOL that contain at most two distinct variables.⁶

We can now say what it is for a theory to be *complete*. A theory T is complete (with respect to a first-order language L) if and only if for every formula φ in L , exactly one of φ and $\neg\varphi$ (not- φ) is a theorem of T . One helpful way of thinking about complete theories is in terms of ambiguity. An incomplete theory is ambiguous on the truth of some formula φ , because neither φ nor its negation is a theorem. We can think of L as providing us with a list of yes-or-no questions corresponding to each non-negated formula: is φ true? If $T \vdash \varphi$, then the answer is “yes.” If $T \vdash \neg\varphi$, then the answer is “no.” If $T \not\vdash \varphi$ and $T \not\vdash \neg\varphi$, then T fails to answer the question. It is better for a conceptual model to contain more information (to answer more questions). If conceptual models are translatable to first-order theories, then the best conceptual models in this regard will be translatable as complete theories. In other words, completeness of a CM script, as described earlier, maps to completeness of first-order theories, as described here.

Given our modifications to the ERM grammar's syntax and our explicit semantics for ERM diagrams via translation to first-order theories, our central formal result shows that all ERM conceptual models will be translatable as complete theories (with respect to the two-variable first-order language of the theory, i.e., the language whose only non-logical symbols are those occurring in the theory's axioms). In other words, all ERM scripts produced under our modifications/guidelines will be free of information loss.

Definition. A script S is complete if and only if its translation into FO², T_S , is complete. A grammar is always-complete if and only if it only allows construction of complete scripts.

Theorem (Always-completeness). *The grammar ERM-R is always-complete.*⁷

In the statement of the theorem, the term “always-complete” is a technical term. It is a formal explication of the informal notion of always-completeness introduced earlier. A CM grammar with FOL semantics is always-complete in this sense if and only if every script produced according to that grammar is translated via the semantics to a complete first-order theory.

We must be cautious in drawing lessons for the disciplined use of the ERM grammar from the always-completeness theorem. It would be easy to misread our formal result as primarily *negative*: “Here are the problems with the ERM grammar, and here are the solutions one must adopt to fix it.” On the contrary, our result is primarily *positive*. The lesson to be drawn is that the problems we identify with our targeted

⁶ FO² is a well-studied fragment of FOL for several reasons: notably, the standard translation of modal logics (including description logic) into FOL embeds them into FO², and logical consequence in FO² is computable, unlike logical consequence in FOL. See Henkin (1967) for an early study of the proof theory of FO²; Grädel et al. (1997) examine the computability properties of FO² as well as its connection to modal logic.

⁷ A proof of the theorem is available from the authors.

fragment of the ERM grammar are *the only problems* with that fragment. Our theorem tells us that addressing the problems identified in the previous section, along with the other issues that go into the definition of ERM-R, results in a grammar with *no further sources of information loss*. As we presented these problems, we offered solutions to them. Our result shows that these solutions are sufficient to produce an always-complete grammar, but not that they are necessary. Other solutions to the problems we identify may be preferable for some particular application, and other possible solutions exist.

In sum, to achieve always-completeness for an ERM-style grammar of similar expressive power to the ERM-R grammar, it suffices to prohibit optional associations and attributes, distinguish “some” and “all” cardinalities for mandatory associations, and control for the remaining issues we survey elsewhere. Our solutions to these problems are non-unique, and furthermore some or all of these problems may fail to lead to information loss in some applications. For example, Burton-Jones et al. (2012) suggest that optionality is not problematic for scripts containing only one optional attribute or association. In such cases, our result shows that no modification to the ERM grammar is necessary to avoid information loss. Again, the force of our result is positive, because the problems we identify are all that a modeler needs to control to avoid information loss on the targeted fragment of the ERM grammar.

Conclusion

Semantics is at the heart of conceptual modeling. Semantics provides the connection between our representations (scripts) and the reality they represent. The current approach to evaluating the semantics of CM grammars relies on ontology to the exclusion of logic. Ontology only gives us half the picture in evaluating a grammar’s semantics. Ontology tells us what is in the reality we aim to represent with a CM script, but it cannot tell us about the details of how our scripts and grammars can represent that reality. A grammar with a wholly adequate semantics must be able both to represent all and only the relevant phenomena in the target domain (ontology) and to represent the phenomena precisely, without ambiguity (logic). In this paper, we have outlined the differences between an ontological approach and a logical approach to evaluating the quality of CM grammars. Moreover, we have identified an issue in the design of CM grammars—information loss—that cannot be treated adequately using a purely ontological approach. We have also given a precise characterization of information loss using logical analysis. Finally, we have given a rigorous application of our logical analysis to provide concrete recommendations for disciplined use of the popular ERM grammar to avoid information loss. In other words, we have identified the need for a program of logical investigation into the quality of CM grammars. We have initiated that research program both in theory and in application. Future research on the logical approach could proceed in many directions. Nonetheless, one way to guide such research would be to consider the main contours of research in the ontological approach over the last 25 years (since the ontological approach was first introduced in Wand and Weber’s 1988 ICIS paper), because that would open up opportunities to examine synergies between the two traditions. This approach would involve focusing on:

Theory: Just as researchers in the ontological tradition investigated the applicability of multiple ontological benchmarks, further research could be conducted using different logical benchmarks. Whereas we used FO², other logics could be investigated such as unrestricted FOL, description logic and other modal logics, or even nonstandard logics (e.g., paraconsistent logics). Moreover, the relationship between ontological and logical dimensions of quality bears investigating. Does one type of quality take precedence over the other, or are they independent? If they are independent, can we produce a unifying theory of both types of quality?

Evaluating different CM grammars: Other CM grammars could be evaluated to determine their logical quality—for instance, the different types of grammars available in the Unified Modeling Language (UML) and the various business process modeling grammars that now exist (such as the Business Process Modeling Notation (BPMN)). The extent to which and the ease with which these grammars can be modified to produce always-complete scripts can be assessed.

Empirical testing: Just as researchers in the ontological tradition have made progress by testing ontological predictions empirically (Burton-Jones et al., 2009), researchers could test the predictions from the logical approach empirically too. For instance, it would be important to test the degree to which different stakeholders benefit from more-complete scripts.

Design: Just as researchers in the ontological tradition have designed CM methods (e.g., Evermann and Wand 2005) and tools (e.g., Wand et al. 2008) incorporating ontological insights, researchers could create new methods and tools (or amend existing methods and tools) based on results of logical analysis, such as that presented here. This avenue seems particularly promising for logical analysis given the computable properties of some logics. Furthermore, it would allow us to provide closer links between the logical approach outlined here and the logical approach carried out in computer science research.

Given that we focused our work solely on the *semantics* of CM *grammars*, another natural direction for research would be to extend the approach developed here with the other dimensions of language (syntax and pragmatics) and the other aspects of the conceptual modeling process (scripts and methods) that we outlined earlier. For instance, researchers could express the modified ERM semantics that we articulated in this paper using different types of syntax and test whether some forms of syntax (e.g., those following the physics of notations, Moody 2009) are easier to understand than other forms of syntax. Likewise, from a pragmatic perspective, researchers could identify whether complete scripts and always-complete grammars are more useful in some contexts than in other contexts. After all, researchers following the ontological approach (e.g., Bodart et al. 2001) have found that it is more useful to apply ontological in contexts where a domain must be understood in detail than in contexts where a rough understanding of a domain will do. Most likely, similar contingencies apply to the value of logical principles. Over time, researchers could assemble such ontological and logical principles into CM methods that could guide practitioners on the best way of using a given CM grammar to produce suitable scripts for a given context. Overall, the ideas developed in this paper offer a rich platform for future research. We hope this paper motivates further research in the logical approach and contributes to the improvement of conceptual models in practice and our theories and methods for investigating them.

Acknowledgements

The research was supported by funds from Australian Research Council Discovery Grant DP 110104386. We also thank the ICIS review team for their constructive advice.

References

- Agarwal, R., P. De, A. P. Sinha, and M. Tanniru: 2000, "On the Usability of OO Representations," *Communications of the ACM* 43(10), 83.
- Agarwal, R., A. P. Sinha, and M. Tanniru: 1996, "Cognitive Fit in Requirements Modelling: A Study of Object and Process Methodologies," *Journal of Management Information Systems* 13(2), 137–64.
- Aguirre-Urreta, M. and G. Marakas: 2008, "Comparing Conceptual Modeling Techniques: A Critical Review of the EER vs. OO Empirical Literature," *The DATA BASE for Advances in Information Systems* 39(2), 9–32.
- Austin, J.: 1962, *How To Do Things With Words*. Clarendon, Oxford.
- Beall, J. and G. Restall: 2005, "Logical Consequence," in E. N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*, fall 2009 edition. URL = <<http://plato.stanford.edu/archives/fall2009/entries/logical-consequence/>>.
- Bodart, F., A. Patel, A. Sim, and R. Weber: 2001, "Should Optional Properties Be Used in Conceptual Modeling? A Theory and Three Empirical Tests," *Information Systems Research* 12(4), 383–405.
- Boehm, B.: 1981, *Software Engineering Economics*. Prentice Hall, Englewood Cliffs, NJ.
- Bunge, M.: 1977, *Ontology I: The Furniture of the World (Treatise on Basic Philosophy 3)*. D. Reidel, New York.
- Burton-Jones, A., R. Clarke, K. Lazarenko, and R. Weber: 2012, "Is Use of Optional Attributes and Associations in Conceptual Modeling Always Problematic? Theory and Empirical Tests," in *International Conference on Information Systems*, pp. 1-16. Orlando, Florida.
- Burton-Jones, A., Y. Wand, and R. Weber: 2009, "Guidelines for Empirical Evaluations of Conceptual Modeling Grammars," *Journal of the Association for Information Systems* 10(6), 495–532.
- Calvanese, D., M. Lenzerini, and D. Nardi: 1998, "Description Logics for Conceptual Data Modeling," in J. Chomicki and G. Saake (eds.), *Logics for Databases and Information Systems*, pp. 229–64. Kluwer.
- Carasik, R., S. Johnson, D. Patterson, and G. Von Glahn: 1990, "Towards a Domain Description Grammar: An Application of Linguistic Semantics," *ACM SIGSOFT Software Engineering Notes*

15(5), 28–43.

- Carnap, R.: 1947, *Meaning and Necessity*. University of Chicago Press, Chicago. 2nd edition 1956.
- Chen, P. P.: 1976, “The Entity-Relationship Model: Toward a Unified View of Data,” *ACM Transactions on Database Systems* 1(1), 1–36.
- Dawson, L. and P. Swatman: 1999, “The Use of Object-Oriented Models in Requirements Engineering: A Field Study,” in *International Conference on Information Systems*, pp. 260–73. Charlotte.
- Eriksson, O. and P. Ågerfalk: 2005, “Information Modelling Based on Semantic and Pragmatic Meaning,” in K. Siau (ed.), *Advanced Topics in Database Research*, Vol. 4. Idea Group Publishing.
- Evermann, J. and Y. Wand: 2005, “Ontology Based Object-Oriented Domain Modelling: Fundamental Constructs,” *Requirements Engineering* 10, 146–60.
- Evermann, J. and Y. Wand: 2006, “Ontological Modeling Rules for UML: An Empirical Assessment,” *Journal of Computer Information Systems* 46(5), 14–29.
- Fettke, P.: 2009, “How Conceptual Modeling Is Used,” *Communications of the AIS* 25(1), 571–92.
- Fillotrani, P., E. Franconi, and S. Tessaris: 2012, “The ICOM 3.0 Intelligent Conceptual Modelling Tool and Methodology,” *Semantic Web Journal* 3(3), 293–306.
- Franconi, E.: 2004, “Using Ontologies,” *IEEE Intelligent Systems* 19(1), 76–7.
- Frederiks, P. J. M., and T.P. van der Weide: 2006, “Information Modeling: The Process and the Required Competencies of its Participants,” *Data & Knowledge Engineering*, 58 (1), 4–20.
- Gemino, A.: 2004, “Empirical Comparisons of Animation and Narration in Requirements Validation,” *Requirements Engineering* 9(3), 153–68.
- Gemino, A. and Y. Wand: 2003, “Evaluating Modeling Techniques Based on Models of Learning,” *Communications of the ACM* 46(10), 79–84.
- Genero, M., G. Poels, and M. Piattini: 2008, “Defining and Validating Metrics for Assessing the Understandability of Entity-Relationship Diagrams,” *Data & Knowledge Engineering* 64, 534–57.
- Ginzberg, M.: 2012, “A Business Dean’s Perspective on the IS Field,” *The DATA BASE for Advances in Information Systems* 43(2), 7–10.
- Grädel, E., P. G. Kolaitis, and M. Y. Vardi: 1997, “On the Decision Problem for Two-Variable First-Order Logic,” *Bulletin of Symbolic Logic* 3(1), 53–69.
- Guizzardi, G.: 2007, “On Ontology, Ontologies, Conceptualizations, Modeling Languages, and (Meta)Models,” in O. Vasilecas, J. Eder, and A. Caplinskas (eds.), *Databases and Information Systems IV (Selected Papers from the 7th International Baltic Conference)*, pp. 18–39. IOS Press, Amsterdam.
- Hadar, I., P. Soffer, and K. Kenzi: in press, “The Role of Domain Knowledge in Requirements Elicitation via Interviews: An Exploratory Study,” *Requirements Engineering Journal*.
- Harel, D. and B. Rumpe: 2004, “Meaningful Modeling: What’s the Semantics of ‘Semantics’?,” *Computer* pp. 64–72.
- Henkin, L.: 1967, *Logical Systems Containing a Finite Number of Symbols (Séminaire de Mathématiques Supérieures 21)*. Presses de l’Université de Montréal, Montreal.
- Hirschheim, R., H. Klein, and K. Lyytinen: 1995, *Information Systems Development and Data Modeling: Conceptual and Philosophical Foundations*. Cambridge University Press, Cambridge.
- Iivari, J., J. Parsons, and Y. Wand: 2006, “Research in Information Systems Analysis and Design: Introduction to the Special Issue,” *Journal of the Association for Information Systems* 7(8), 509–13.
- Kent, W.: 2000, *Data and Reality*. 1st Books Library. Originally published by North Holland, 1978.
- Krogstie, J., G. Sindre, and H. Jorgensen: 2006, “Process Models Representing Knowledge for Action: A Revised Quality Framework,” *European Journal of Information Systems* 15, 91–102.
- Lindland, O., G. Sindre, and A. Sølvberg: 1994, “Understanding Quality in Conceptual Modeling,” *IEEE Software* 11(2), 42–9.
- Lukyanenko, R. and J. Parsons: 2011, “Information Loss in the Era of User-Generated Data,” in *SIG Information Quality Workshop*, pp. 1–4. Shanghai.
- Lyytinen, K.: 1987, “Two Views of Information Modeling,” *Information and Management* 12, 9–19.
- Maes, A. and G. Poels: 2007, “Evaluating Quality of Conceptual Modelling Scripts Based on User Perceptions,” *Data & Knowledge Engineering* 63(3), 701–24.
- Moody, D. L.: 2005, “Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models: Current State and Future Directions,” *Data & Knowledge Engineering* 55, 243–76.
- Moody, D. L.: 2009, “The ‘Physics’ of Notations: Towards a Scientific Basis for Constructing Visual Notations in Software Engineering,” *IEEE Transactions on Software Engineering* 35(5), 756–78.
- Moody, D. L. and G. G. Shanks: 2003, “Improving the Quality of Data Models: Empirical Validation of a

- Quality Management Framework,” *Information Systems* 28(6), 619–50.
- Nelson, H. and D. Monarchi: 2007, “Ensuring the Quality of Conceptual Representations,” *Software Quality Journal* 15, 213–33.
- Parsons, J. and Y. Wand: 2008, “Using Cognitive Principles to Guide Classification in Information Systems Modeling,” *MIS Quarterly* 32(4), 839–68.
- Pitts, M. G. and G. J. Browne: 2004, “Stopping Behavior of Systems Analysts During Information Requirements Elicitation,” *Journal of Management Information Systems* 21(1), 203–26.
- Rockwell, S. and A. Bajaj: 2004, “COGEVAL: Applying Cognitive Theories to Evaluate Conceptual Models,” *Advanced Topics in Database Research* 4, 255–82.
- Scheer, A.-W. and F. Habermann: 2000, “Making ERP a Success,” *Communications of the ACM* 43(4), 57–61.
- Schuette, R.: 1999, “Architecture for Evaluating the Quality of Information Models – A Meta and an Object Level Comparison,” in J. Akoka, M. Bouzeghoub, I. Comyn-Wattiau, and E. Metais (eds.), *International Conference on Conceptual Modeling (ER ’99)*, pp. 490–505. LNCS 1728, Paris, France.
- Selic, B.: 2003, “The Pragmatics of Model-Driven Development,” *IEEE Software* 20(5), 19–25.
- Shanks, G., E. Tansley, and R. Weber: 2003, “Using Ontology to Validate Conceptual Models,” *Communications of the ACM* 46(10), 85–9.
- Siau, K. and M. Rossi: 2011, “Evaluation Techniques for Systems Analysis and Design Modelling Methods – A Review and Comparative Analysis,” *Information Systems Journal* 21, 249–68.
- Siau, K. and X. Tan: 2005, “Improving the Quality of Conceptual Modelling Using Cognitive Mapping Techniques,” *Data & Knowledge Engineering* 55, 343–65.
- Stark, J. and W. Esswein: 2012, “Rules from Cognition for Conceptual Modelling (ER 2012),” in P. Atzeni, D. Cheung, and S. Ram (eds.), *International Conference on Conceptual Modeling*, pp. 78–87. LNCS 7532, Florence, Italy.
- Teo, H.-H., H. C. Chan, and K. K. Wei: 2006, “Performance Effects of Formal Modeling Language Differences: A Combined Abstraction Level and Construct Complexity Analysis,” *IEEE Transactions on Professional Communication* 49(2), 160–75.
- Utting, M. and B. Legeard: 2007, *Practical Model-Based Testing: A Tools Approach*. Morgan Kaufmann.
- Wand, Y.: 1996, “Ontology as a Foundation for Meta-Modelling and Method Engineering,” *Information and Software Technology* 38, 281–7.
- Wand, Y., D. Monarchi, J. Parsons, and C. Woo: 1995, “Theoretical Foundations for Conceptual Modelling in Information Systems Development,” *Decision Support Systems* 15, 285–304.
- Wand, Y., V. C. Storey, and R. Weber: 1999, “An Ontological Analysis of the Relationship Construct in Conceptual Modeling,” *ACM Transactions on Database Systems* 24(4), 494–528.
- Wand, Y. and R. Weber: 1988, “An Ontological Analysis of Some Fundamental Information Systems Concepts,” in *Proceedings of the 9th International Conference on Information Systems*, pp. 213–25. Minneapolis, MN.
- Wand, Y. and R. Weber: 1993, “On the Ontological Expressiveness of Information Systems Analysis and Design Grammars,” *Journal of Information Systems* 3, 217–37.
- Wand, Y. and R. Weber: 2002, “Information Systems and Conceptual Modelling – A Research Agenda,” *Information Systems Research* 13(4), 363–76.
- Wand, Y., C. Woo, and O. Wand: 2008, “Role and Request Based Conceptual Modeling – A Methodology and a CASE Tool,” in *Proceedings of CAiSE’08 Forum*, pp. 77–80. Montpellier, France.
- Wastell, D. G.: 1996, “The Fetish of Technique: Methodology as a Social Defence,” *Information Systems Journal* 6, 25–40.
- Weber, R. and Y. Zhang: 1996, “An Analytical Evaluation of Niam’s Grammar for Conceptual Schema Diagrams,” *Information Systems Journal* 6(2), 147–70.
- Yourdon, E.: 1989, *Modern Structured Analysis*. Prentice Hall, Englewood Cliffs, NJ.