On the Ontological Quality and Logical Quality of Conceptual-Modeling Grammars: The Need for a Dual Perspective

Clarke, R., Burton-Jones, A., & Weber, R. (2016). On the Ontological Quality and Logical Quality of Conceptual-Modeling Grammars: The Need for a Dual Perspective. *Information Systems Research*, *27*(2), 365-382. https://doi.org/10.1287/isre.2016.0631

Published in:
Information Systems Research

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

On the Ontological Quality and Logical Quality of Conceptual-Modeling Grammars: The Need for a Dual Perspective

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

Andrew Burton-Jones
UQ Business School, University of Queensland, St Lucia, QLD 4072, Australia, abj@business.uq.edu.au

Ron Weber
UQ Business School, University of Queensland, St Lucia, QLD 4072, Australia, uqrweber@uq.edu.au
Faculty of Information Technology, Monash University, Melbourne, VIC 3800, Australia, ron.weber@monash.edu

A core activity in information systems development involves building a conceptual model of the domain that an information system is intended to support. Such models are created using a conceptual-modeling (CM) grammar. Just as high-quality conceptual models facilitate high-quality systems development, high-quality CM grammars facilitate high-quality conceptual modeling. This paper provides a new perspective on ways to improve the quality of the semantics of CM grammars. For many years, the leading approach to this topic has relied on ontological theory. We show, however, that the ontological approach captures only half the story. It needs to be coupled with a logical approach. We explain how the ontological quality and logical quality of CM grammars interrelate. Furthermore, we outline three contributions that a logical approach can make to evaluating the quality of CM grammars: a means of seeing some familiar conceptual-modeling problems in simpler ways; the illumination of new problems; and the ability to prove the benefit of modifying existing CM grammars in particular ways. We demonstrate these benefits in the context of the Entity-Relationship grammar. More generally, our paper opens up a new area of research with many opportunities for future research and practice.

Key words: conceptual modeling; semantics; ontology; logic

1. Introduction

For many years, the quality of the semantics embodied in conceptual modeling (CM) grammars has often been evaluated by mapping their constructs to the constructs in a benchmark ontology (e.g., Gregersen and Jensen 1999, Opdahl and Henderson-Sellers 2002, Recker et al. 2011, Wand and Weber 1993, zur Muehlen et al. 2007). To the extent a one-to-one and onto mapping exists, their semantics are deemed to be ontologically clear and complete—that is, the meaning of each construct in the grammar is well defined (clarity), and the grammar has a set of constructs that can be used to represent each and every type of phenomena that occurs in the real world (completeness).

The ontological approach to evaluating the semantics of CM grammars has often provided important insights about the strengths and weaknesses of different grammars (e.g., Green et al. 2011, Recker et al. 2010). It has also produced some counterintuitive and controversial results, such as the potential problems associated with using certain kinds of grammatical constructs when generating scripts to represent a domain (e.g., Bodart et al. 2001, Gemino and Wand 2005, Burton-Jones et al. 2009). Nonetheless, predictions made using the ontological approach about the strengths and weaknesses of the semantics in a particular CM grammar have sometimes received strong empirical support (e.g., Recker et al. 2011). For these sorts of reasons, the ontological approach continues to motivate ongoing research on the merits of CM grammars and specific constructs in the grammars.

In this paper, however, we argue that the ontological approach to evaluating the semantics of CM grammars provides only a *partial* means of assessing the likely strengths and weaknesses of the grammars. We propose an alternative, complementary approach to evaluating the semantics of CM grammars—namely, the *logical* approach. The logical approach evaluates the statements expressed in a CM script against a benchmark logic to determine whether the script is consistent and precise—that is, whether the script contains any conflicting statements about its focal domain (consistency), and whether the script can answer all questions it is supposed to answer about its focal domain (precision). From an analysis of the consistency and precision of the scripts generated via a CM grammar, the strengths and weaknesses of the grammar can then be determined and solutions can be devised.

We show that the logical approach to evaluating the semantics of a CM grammar can pinpoint *all* the defects leading to inconsistency and imprecision that need to be addressed in the grammar. Moreover, it can devise solutions for the defects to ensure all scripts created using the grammar will be logically consistent and precise—an achievement that is unlikely to occur using the traditional ontological approach. In part, this reflects that the logical and ontological approaches focus on different properties of a CM grammar in the analytical approaches they use—they foreground and background different properties of the grammar. Ultimately, any defects in a CM grammar might become apparent under both approaches, but each approach tends to surface different defects and reveal different solutions for addressing them because of the particular analytical approach it uses.

On the other hand, the logical and ontological approaches differ in their levels of generality. Every ontology embeds a logic: commitment to an ontological theory entails commitment to a logical theory. Therefore, the results obtained from a logical analysis will be more general than the results obtained from an ontological analysis. If a result can be shown to hold given only the assumption of a certain logical theory, those results will hold for multiple ontologies. For example, both Bunge's (1977) ontology and Searle's (1995) ontology assume classical first-order logic, so whatever can be shown to follow from first-order logic alone will hold in both Bunge's and Searle's ontologies.

Our paper proceeds as follows. We begin by explaining the relationship between ontology and logic and why both are needed to evaluate the quality of the semantics of a CM grammar. Next, we explain the logical approach to evaluating the semantics of CM grammars and the relationship between the logical approach and ontological approach. We then show how the logical approach can be applied to identify certain types of semantic defects in a widely used conceptual modeling grammar—namely, the entity-relationship modeling (ERM) grammar. We also show how the logical approach can be used to (a) identify how the ERM grammar can be modified to overcome these semantic defects, and (b) demonstrate that no more of these types of semantic defects exist in the modified ERM grammar. We conclude by discussing some implications of the logical approach for research and practice.

2. Background: Ontology, Logic, and the Semantics of Conceptual Modeling Grammars

Our focus is on the *semantics* of CM grammars (the rules determining the meaning of the grammatical constructs) rather than their syntax (the symbols used to represent grammatical constructs and how these symbols can be combined) or their pragmatics (how grammars are used in different contexts). While the syntax and pragmatics of CM grammars are important factors affecting their overall merits (e.g., Ågerfalk 2010, Bera et al. 2014, Moody 2009), we see their semantics as primary. Ultimately, the implementation of an information system requires an unequivocal commitment to a particular set of semantics. If these semantics are not a faithful representation of someone's or some group's perception of the focal real-world domain, the efficacy of the information system is likely to be undermined (Dunn and McCarthy 1997, pp. 36–37; Sheth 1997; Wand and Weber 1995, p. 206).

We argue that ontology and logic *both* provide an important means to evaluate the semantics of CM grammars. In some cases, their contributions are unique; in other cases, their contributions are related. As background to the logical approach we propose for evaluating CM grammars, therefore, in the subsections below we briefly discuss the relationship between ontology and logic as foundations for evaluating the semantics of CM grammars.

2.1. Relationship between Ontology and Logic

The disciplines of ontology and logic have multiple schools of thought. Moreover, a longstanding, healthy debate exists about how they relate (Hofweber 2014). In short, there is no single answer to their nature and their relationship. For this reason, we take one well-accepted account, which is that ontology presupposes logic, much as physics presupposes mathematics. By this we mean that one cannot begin to answer questions of ontology without first settling questions of logic. Just as mathematics provides the very language in which physical theories are expressed, so logic provides the language in which ontological theories are expressed.

According to this account, a given ontology provides a theory about what kinds of phenomena exist in a domain. Such a theory requires a language—typically a logic—in which the theory can

be specified (Hofweber 2014). Thus, any ontology will have a logic embedded in it. Bunge's (1977, p. 14) ontology, the most widely used ontology in IS research (Fonseca 2007), takes this view explicitly. According to Bunge (1977, p. 102), a single logic should underlie all ontologies. Thus, while two distinct ontologies might disagree about the fundamental constituents of the world, they should agree on questions of logical consequence.

Because logic is in this sense prior to ontology—questions about logical consequence must be settled before we can address questions of the fundamental constituents of the world—the disciplines of logic and ontology operate at different levels of generality. Thus, logical and ontological analysis can be expected to foreground and background different properties of a script or grammar. Logicians are best equipped to evaluate a script in ways that do not depend on specific ontological commitments, giving results that hold across a variety of ontologies. On the other hand, ontologists can evaluate scripts for a more extensive, deeper reflection of their chosen ontology. In this sense, ontological and logical analysis are complementary: results depending on logic alone are broader in their application, while committing to a specific ontology as well as its underlying logic allows deeper analysis of a script.

2.2. Some Implications for Conceptual Modeling

This relationship between logic and ontology—that ontology presupposes logic—suggests that logical analysis is both fundamental to CM research and long overdue. It is fundamental because it can provide results that hold regardless of the ontology used to evaluate CM grammars. For example, IS researchers have often debated whether they should choose Bunge's ontology or Searle's ontology (Allen and March 2006, p. 5; Lemieux and Limonad 2011, p. 34), but both ontologies presuppose classical first-order logic (Bunge 1977, Ludwig 2007, Searle 1995, pp. 104–112). Thus, if we can obtain results for conceptual modeling that depend only on first-order logic and not anything specific to Bunge's or Searle's ontology, these results can be accepted by all parties to an ontological debate.

Because any ontology embeds a logic, a commitment to ontological analysis also entails a commitment to logical analysis. If CM researchers assume ontology helps them assess the semantics of

CM grammars, they are implicitly assuming that logic also helps them assess the semantics of CM grammars. Thus, ontological analysis and logical analysis should go hand-in-hand. Indeed, given that ontology presupposes logic, logical analysis should precede ontological analysis. So far, this outcome has not occurred in the IS field. As a result, IS researchers have addressed some matters of semantics through ontological analysis, such as optionality (Bodart et al. 2001, Gemino and Wand 2005), when we argue these topics are addressed more effectively through logical analysis.

To make the implications of the relationship between ontology and logic more concrete, assume that we have been asked to evaluate the semantics in an ERM script. Through the lens of logic, the script is a set of statements about a domain together with their logical entailments. This set of statements can then be evaluated for consistency and precision: does the script provide exactly one answer to each question we could ask about the domain using the script's vocabulary? Through the lens of ontology, the script is a set of statements that describes what exists in the domain. These statements can then be evaluated for descriptive accuracy: does the script's description of the domain make ontological sense, using exactly one grammatical construct for each ontological construct? Logical and ontological analysis evaluate scripts along orthogonal dimensions. A script may be ontologically clear and complete without being logically consistent or precise, or *vice versa*. Ultimately, we need both properties in high-quality scripts, and we need CM grammars capable of producing such scripts.

3. Evaluating the Semantics of Conceptual Modeling Grammars Using Logic

We are not the first to propose that logic can play a valuable role in research on the semantics of CM grammars. Table 1 provides an overview of and some examples of prior work that has been done.

Our approach differs from this prior work in two ways, however. First, our focus is not the development of languages (grammars) to support automated reasoning. Rather, our aim is to discover properties of CM grammars that support the production of CM scripts with high-quality semantics. Second, our focus is not on articulating precisely the semantics of specific constructs

Table 1 Examples of Prior Research Using Logic to Design or Evaluate CM Grammars.

Focus of Logical Analysis	References
Development of computational ontologies that support reasoning within applications.	Calvanese et al. (1998), Fillotrani et al. (2012), Franconi (2004), Hitzler et al. (2012)
Specification of the semantics of set-based constraints.	Currim and Ram (2012), Ram and Khatri (2005)
Reconstruction of existing CM grammars to improve logical precision and to facilitate reasoning with and implementation of databases.	Thalheim (1993)
Development of a grammar to support implementation of and reasoning in knowledge-intensive information systems.	Jeusfeld et al. (2009), Mylopoulos et al. (1990), Steel (1986)

in a CM grammar. Rather, our focus is on the logical analysis of CM grammars *in toto* to assess and ensure their overall semantic quality. This is an important shift in emphasis compared to the traditional approach taken in the ontological tradition of CM research. To date, researchers in this tradition have used ontological theories to assess specific constructs in a grammar (e.g., Wand et al. 1999) or even entire grammars (e.g., Wand and Weber 1993), but they have not proposed ways to ensure a grammar will have the property of always producing scripts with desirable ontological properties (such as ontological completeness or clarity).

We begin by articulating the semantic properties of CM scripts and grammars that are desirable from the perspective of a logical benchmark. Once we have these properties, we then explain how these properties complement the criteria used to undertake ontological evaluations of CM scripts and grammars.

3.1. Desirable Logical Properties of Conceptual Modeling Scripts and Grammars

The basic idea behind our approach is that a CM script can be treated as a logical theory (for similar positions, see Mylopoulos et al. 1990, p. 351; Steel 1986, p. 259). Logical analysis then involves comparing the statements (theorems) expressed in a script against a benchmark logic.

If we conceive of a script as a logical theory, we can employ two well-studied properties of logical theories—namely, *consistency* and *precision*.[1] To determine whether a script possesses these properties, we use the following approach:

- Take as input a set of phenomena that has been modeled via a CM script.
- Convert the CM script into a set of sentences in the benchmark logic.
- Use the set of sentences in the benchmark logic and the benchmark logic to generate the complete set of questions that the CM script should be able to answer.[2]
- Check via the set of sentences in the benchmark logic whether the script provides either a "yes" or "no" answer to each question.

The test for consistency and precision is then relatively straightforward. If a script gives more than one answer to one or more questions (i.e., both a "Yes" and "No" answer to a single question), it is *inconsistent*. Somewhere the script provides contradictory representations of some phenomena. For example, it shows "managers" as both a subclass of "employees" and a subclass of "contractors" (individuals who are *not* employees).

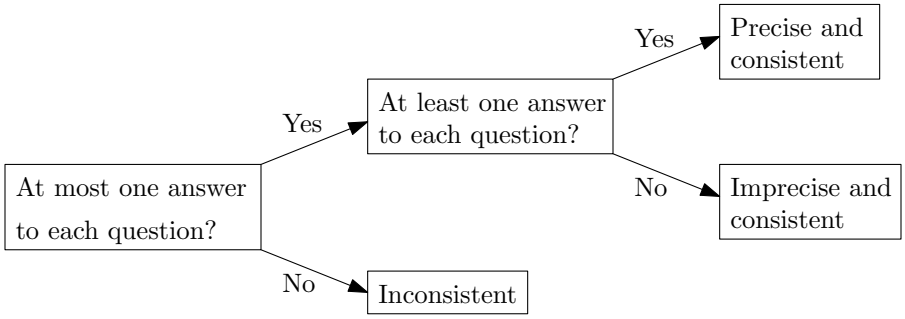
If a *consistent* script fails to answer one of the questions with a "Yes" or "No," however, it is *imprecise*. Somehow the script is ambiguous about domain phenomena. For example, it shows a class of "employees" and a class of "contractors," but it does not indicate whether these two classes can overlap. In short, when a consistent but imprecise script does answer a Yes/No question, it provides only one answer to the question, but it does not provide an answer to all Yes/No questions.

[1] We use the term "precision" to refer to what logicians call "completeness." This is because we do not wish to suggest a parallel between ontological completeness and the property we discuss here. Moreover, while the term "completeness" captures logicians' historical reason for being interested in this property, it does not capture the reason we propose conceptual modelers should be interested in it: namely, that a precise script *avoids ambiguity*.

[2] Past CM research has also used this question-answering strategy (Greenspan et al. 1986, pp. 17–18; Mylopoulos et al. 1990, p. 351). Any logic will include recursive rules for producing complex formulas (propositions) from simpler ones. Given a list of non-logical symbols (predicates, relations, names, etc.), these rules specify a list of formulas built up from these symbols plus logical symbols (connectives, quantifiers, variables, etc.). This is our list of questions: for each formula, ϕ, we have the "yes/no" answer to the question, "Is ϕ true?"

Our two logical criteria are independent of each other (Figure 1). On the one hand, an inconsistent CM script (one that provides both a "Yes" and "No" answer to at least one question) is neither precise nor imprecise. Once inconsistency exists, precision is not a relevant criterion to apply to the script. On the other hand, a consistent script may be either precise or imprecise. It will be precise only if it answers *all* questions with either a "Yes" or a "No" answer. If it fails to answer all questions, it might be consistent in relation to those questions it does answer, but nonetheless it is imprecise because it does not answer all questions.

Figure 1 Relationship between Consistency and Precision.



Note, a logical approach cannot tell us whether a script is ontologically *incomplete*—in other words, whether the script is unable to answer one or more questions that are deemed to be relevant about domain phenomena. For instance, the script fails to answer questions about managers because managers are not represented in the script in the first place, even though managers are deemed to be a relevant domain phenomenon. Incompleteness, however, is a difficult notion because identifying domain phenomena that might be relevant to stakeholders is often problematic (Pitts and Browne 2004). Indeed, discussions about the lay meaning of incompleteness often evoke the *pragmatic* notion of relevance, which logical analysis is not suited to uncovering.

Nor can a logical approach tell us whether a consistent script is *inaccurate*—in other words, whether a script gives an incorrect answer to one or more questions about a domain. For example, it answers the question "Are all managers employees?" with "Yes" when the correct answer is "No." Nonetheless, logical analysis can uncover inaccuracy when it detects a script is inconsistent. An inconsistent script represents a domain in a logically impossible and therefore inaccurate way.

Like incompleteness, however, inaccuracy is a difficult notion because identifying domain phenomena accurately and representing them accurately in a script are often problematic (Dawson and Swatman 1999, Hadar et al. 2014, Wastell 1996).

If the desirable logical properties of a CM *script* are consistency and precision, the desirable properties of a CM *grammar* are therefore that it *always* generates (a) consistent scripts, and (b) precise scripts. In other words, it should be impossible to use the constructs and production rules in the grammar in such a way that inconsistent or imprecise scripts are produced. Through logical analysis of the scripts that a CM grammar potentially can produce, we seek to determine whether the grammar possesses these two properties.

3.2. Relationship between Ontological Criteria and Logical Criteria for Evaluating Conceptual Modeling Grammars and Scripts

Table 2 shows the ontological criteria and logical criteria that we propose should be used as a basis for evaluating the semantic quality of CM grammars and scripts. Because the ontological criteria we propose differ slightly from the traditional ontological criteria used to evaluate the semantics of CM grammars, Appendix I provides a brief description of the traditional criteria and the ways in which the ontological criteria in Table 2 differ from these traditional criteria.

Note in Table 2 how our logical criteria complement ontological criteria as a basis for evaluating the semantic quality of CM grammars and scripts. Importantly, our logical criteria are independent of the ontological criteria. On the one hand, the properties of ontological clarity and completeness are associated with a mapping between the basic constructs of a CM grammar and the constructs of a benchmark ontology. On the other hand, the properties of logical consistency and precision are associated with a mapping between the entailments (theorems) of a script and the sentences of a benchmark logic. In other words, using the criteria of ontological clarity and completeness, the ontological approach evaluates the non-logical symbols (grammatical constructs) in a CM grammar, whereas using the criteria of logical consistency and precision the logical approach takes the non-logical symbols (grammatical constructs) as given and evaluates the ways these symbols (grammatical constructs) are combined. In conjunction with the ontological criteria, therefore, our

Table 2 Ontological and Logical Criteria for Assessing Semantic Quality

		Application to	
		CM Script	**CM Grammar**
Application of	**Ontology**	Ontologically clear	Ontologically complete Always ontologically clear
	Logic	Logically consistent Logically precise	Always logically consistent Always logically precise

Definitions related to ontology:
Ontologically clear: Each grammatical construct in a script represents only one construct in the chosen ontological benchmark (i.e., *no construct overload or excess*).
Ontologically complete: A CM grammar offers all constructs needed to produce scripts containing any construct in the chosen ontological benchmark (i.e., *no construct deficit*).
Always ontologically clear: A CM grammar ensures that in all scripts it produces each grammatical construct represents only one construct in the chosen ontological benchmark (i.e., *no construct overload or excess*).
Definitions related to logic:
Logically consistent: A CM script does *not* provide *both* a "Yes" and a "No" answer to any question the script purports to answer.[a]
Logically precise: A logically consistent CM script *does* provide *either* a "Yes" or a "No" answer to all questions the script purports to answer.
Always logically consistent: A CM grammar ensures any script it produces does not provide both a "Yes" and a "No" answer to any question the script purports to answer.
Always logically precise: A CM grammar ensures any logically consistent script it produces provides either a "Yes" or a "No" answer to any question the script purports to answer.

[a] By "all questions the script purports to answer," we mean all questions that can be formulated using only the script's vocabulary (i.e., those class, attribute, and relationship labels appearing in the script). By using a given class/attribute/relationship label, a script commits itself to modeling the named phenomenon—thus, we say it purports to answer questions about that phenomenon.

logical criteria provide a more comprehensive basis for evaluating the semantics of CM grammar and scripts.

4. Evaluating Whether the ERM Grammar Is a Logically Always-Precise Grammar

In this section, we provide a demonstration of how logic can be used to evaluate the semantics of the ERM grammar. We chose the ERM grammar because it is well known and widely used by researchers and practitioners. Nonetheless, the form of our analysis can be applied to other grammars (e.g., the Unified Modeling Language).

In our illustrative analysis, for two reasons we focus only on evaluating logical *precision* in the ERM grammar—in other words, we do not address logical consistency. First, for those aspects of the ERM grammar that we study, precision is more challenging to ensure than consistency. Second, based on our analyses, we believe the results we have obtained for precision are more interesting

and novel than those we have obtained for consistency. Third, some aspects of consistency have already been studied in the CM literature (e.g., Mylopoulos et al. 1990); thus, possible approaches that might be used to evaluate whether or prove that a CM grammar generates consistent scripts are available. In short, by focusing on precision and not consistency in our illustrative analysis, we believe we are providing a more powerful and more compelling demonstration of the value of the logical approach to evaluating CM grammars.[3]

4.1. Choice of Benchmark Logic

The first step in evaluating the logical precision of a CM grammar is to choose a benchmark logic. For three reasons, we chose FO^2, which is the two-variable fragment of first-order logic (FOL) without identity ("=").

First, a logic based on FOL provides a powerful benchmark. Relative to weaker logics (such as propositional modal logic), FOL formulas allow us to place more constraints on a domain, describing the world in finer detail. In some applications, FOL's power is an obstacle—for instance, first-order consequence is not decidable (Church 1936, Turing 1936).[4] In our case, however, it increases the significance of our results. Recall we aim to show all consistent scripts produced by a CM grammar are precise, which means they give an answer to every question one can ask about the phenomena represented in the script. Because FOL is powerful, it allows us to ask more questions than a weaker logic. Therefore, showing a grammar is always-precise relative to FOL is a greater achievement than obtaining the same outcome with a weaker logic.

Second, first-order theories (that is, collections of FOL formulas together with their consequences) have a well-studied property—what logicians call *completeness*[5]—which mirrors our notion of *precision* in CM scripts. A first-order theory is complete iff, for every FOL formula ϕ, either ϕ or

[3] Nonetheless, at the end of Appendix II, we state necessary and sufficient conditions for consistency in a script generated via our modified ERM grammar (ERM-R). In the interests of brevity, however, we do not show a proof of these conditions (instead, our formal analyses in Appendix II focus on precision).

[4] By contrast, Mortimer (1975) establishes the decidability of FO^2, and Grädel et al. (1997) proves that its satisfiability problem is NEXPTIME-complete.

[5] Recall that we have been using the term "completeness" to refer to an ontological property of CM scripts, not a logical one. See Table 2 and note 1 above.

not-ϕ (but not both) is a consequence of the theory. This definition clearly matches our definition of precision. Thus, by taking FO^2 as our benchmark logic, we open a route to demonstrating a script is precise or a grammar is always-precise. In particular, we use the following strategy: give a semantics that translates CM scripts as first-order theories, then show the result is logically complete.

Third, we do not need the full expressiveness of FOL to show how to enact a logical evaluation of the ERM grammar. For instance, in an ERM script, we do not expect to make a statement that a class of entities contains more than n members ($n \geq 1$).[6] Whereas such a statement can be made in FOL, it cannot be made in FO^2. We facilitate our work by using a more restricted version of FOL in the form of FO^2.

4.2. Restricting the ERM Grammar

To apply FO^2 effectively, we needed to restrict the ERM grammar in three ways. The reason is that we need our CM grammar to match our benchmark logic in expressive power. For example, because FO^2 cannot express the claim that a class has more than n members, we restrict ERM's ability to express this claim too. These restrictions mean that our results only constitute the beginning of a full logical analysis of the ERM grammar.

The first restriction we imposed was removing ternary and higher-order relations from the grammar's constructs. If we included these constructs in our modified grammar, we suspect, but do not prove, that our modified grammar would still be always-precise, relative to an appropriately chosen fragment of FOL. In the interests of simplicity, however, we consider only binary associations between classes.

The other two conditions we imposed were (a) prohibiting non-dichotomic properties,[7] and (b) prohibiting cardinality notation. At this time, we are unsure whether it is possible to prove an

[6] Recall that our interest here is conceptual modeling. If our interest had instead been modeling a database, then we would wish to be able to express such claims (see Currim and Ram 2006) and a different benchmark logic would be more appropriate.

[7] Following Bunge (1977, pp. 68–69), we use the term "dichotomic" for properties that take Boolean (true/false) values, such as "is 25 years old." These properties contrast with non-dichotomic properties such as "age," which takes a numerical value.

ERM grammar containing these constructs is always-precise. For the moment, it is easier to prove that our modified ERM grammar is always-precise if these constructs are proscribed.

4.3. Evaluation Process

To evaluate whether the restricted ERM grammar is always-precise, we followed two major steps.

First, we composed a formal semantics for ERM scripts. That is, we gave instructions for determining, for any ERM script, what the script entails, with these entailments expressed in the language of the benchmark logic. In our case, the benchmark logic is FO^2, so a formal semantics specifies a mapping from ERM scripts to FO^2 theories (equivalently, sets of sentences).

Second, once we had a formal semantics, we sought to determine whether any consistent ERM scripts are imprecise. This process is iterative. Most likely, several solution paths exist. Nonetheless, given we have a mapping from ERM scripts to FO^2 theories, we can address the question of whether any consistent ERM scripts are imprecise using the machinery of first-order logic. Specifically, the question of whether any consistent ERM scripts are imprecise reduces to the question of whether any theory in the range of the mapping is incomplete (in the logician's sense). In this regard, logical incompleteness is a well-studied property. Thus, many theorems exist that we can use to facilitate this stage of the analysis. For instance, we have many proven theorems of the form, "If a theory T has the property A, then T is (in)complete." In Appendix II we use such a theorem in our proof of ERM-R's always-preciseness.[8]

To find ways a consistent ERM script might be imprecise, we look to our semantics for ERM. When translating ERM scripts into FO^2, we encounter three types of predicate symbols: for each

[8] Specifically, in Appendix II, we use Lemmas 2 and 3 of Shoenfield (1967, p. 83), which in combination state the following: If every variable-free formula of a consistent theory T is decidable in T, and if every simply existential formula is equivalent in T to an open formula, then T is complete. Our proof centers on showing that the hypotheses of this theorem are always satisfied in ERM-R—that, for every consistent script of ERM-R, the result of mapping that script into FO^2 according to our formal semantics is a theory with the two properties listed above (namely, every variable-free formula is decidable, and every simply existential formula is equivalent in the theory to an open formula). See Appendix II for details.

class in a script, we have a unary predicate C_i; for each attribute, we have a unary predicate P_j; and for each relation, we have a binary predicate R_k. Given the established theorems our proof relies on (see note 8 above), we can safely restrict our attention to *open* formulas—that is, formulas without quantifiers ($\forall$ and $\exists$). What, then, are the open formulas in the language of T? They will be boolean combinations of the following atoms:

- $C_i x$ and $C_i y$, for all class predicates C_i;
- $P_j x$ and $P_j y$, for all attribute predicates P_j;
- $R_k xx$, $R_k xy$, $R_k yx$, and $R_k yy$, for all association predicates R_k.

Because FO^2 lacks an identity predicate and does not allow more than two variables per formula, the range of possible formulas is restricted. We need only consider combinations of the above types of atomic formulas, possibly with one existential quantifier prefixed. In doing so, we found the following problem cases:

1. $C_i x \wedge C_j y \wedge \neg R_k xy$

- If the association R_k between C_i and C_j is mandatory, we can ask whether the relation holds between *all* members of C_i and *all* members of C_j. This is the question of cardinality (§5.2.1).

2. $R_k xy \wedge R_k yx$

- If x bears a relation to y, must/can y bear that same relation to x? This is the question of symmetric and asymmetric relations (§5.2.2).

3. $R_k xx$

- Can an individual bear a given relation to itself? This is the question of reflexive relations (§5.2.3).

4. $R_k xy \wedge R_m xy$

- If x bears relation R_k to y, must/can x also bear the distinct relation R_m to the same individual y? This is the question of individuation of relations (§5.2.4).

5. $C_1 x \wedge C_2 x$

$C_3 x \wedge \neg C_4 x \wedge \neg C_5 x$

- Do classes C_1 and C_2 overlap, or are they disjoint? Do the subclasses C_4 and C_5 exhaust the superclass C_3? This is the question of overlap, disjointness, and exhaustiveness (§5.2.5).

6. $C_i x \wedge P_j x \wedge R_k xy$

- Suppose the attribute P_j and the relation R_k are both optional for the class C_i. Can a member of C_i both possess P_j and bear R_k to something? This is the problem of optionality (§5.2.6).

5. Making the ERM Grammar Always Logically Precise

Having pinpointed six defects in the ERM grammar, in this section we propose modifications to the ERM grammar that will mitigate the effects of these defects and ensure the grammar never generates imprecise scripts. We denote the modified ERM grammar as ERM-R.

We begin by discussing an important assumption that underlies the changes we made to the ERM grammar. We then discuss the six changes we made to the ERM grammar to address the defects. Finally, we summarize our recommendations for disciplined use of the ERM grammar.

5.1. The "Completeness" Assumption

We note one assumption we make before proceeding. Specifically, we assume ERM scripts are intended to present a complete rather than a partial representation of the domain they model. Formally, this assumption entails that we understand the absence of a claim as a claim of absence. For instance, if a script has no class called "students," we interpret the script as claiming the domain modeled does not include any students.

That we assume scripts are complete should not be a surprise. A script intended as only a partial representation of a domain will, necessarily, leave questions unanswered. This outcome is borne out in our formal proof. In providing a formal semantics for our modified ERM grammar, the success of our proof depends upon scripts being interpreted to include a claim that the modeled domain includes no entities other than those represented. [9]

[9] Despite our focus on whether scripts are complete, we do not claim use of partial scripts is always inappropriate. Rather, we are concerned with guidelines for producing high-quality scripts that give a complete picture of the modeled domain. In this light, precise scripts have a clear advantage over imprecise ones.

Interpreting scripts as complete should not be confused with making a "closed-world assumption" (Reiter 1978) or as conflicting with agile methods (Ambler 2002). The classical closed-world assumption involves treating everything not known to be true as known to be false. Thus, under a closed-world assumption, scripts are not subject to revision. On the contrary, we treat scripts as presenting a complete picture of the modeler's knowledge of a domain at a point in time. In this light, the absence of a class of students in a script means that, according to the modeler's understanding at a point in time, the modeled domain has no students. As the modeler's understanding of the domain unfolds, a class of students might be included in the script.

5.2. Modifications to the ERM Grammar

The six subsections below describe the specific modifications we made to the restricted ERM grammar (see §4.2 above) to achieve always-preciseness. Each subsection begins with an example of an imprecise ERM script. We show why the script is imprecise by providing a question that the script is unable to answer. In essence, we seek to provide the intuition behind our formal results (Appendix II). Because some ERM notation we use is non-standard, Table 3 provides a legend.

Note, importantly, that all of our modifications to the ERM grammar are *semantically* motivated. Some introduce (§§5.2.1–5.2.3) or eliminate (§5.2.6) syntactic elements, but these amendments to ERM's syntax are designed to improve the grammar's semantics. On the other hand, some of our modifications (§§5.2.4 and 5.2.5) leave the ERM's syntax unchanged, instead modifying the semantic rules for interpreting ERM scripts. That is, §§5.2.4 and 5.2.5 modify how one is to read a script (and hence how a competent modeler would construct a script), rather than adding or removing vocabulary elements.

Table 3 Legend for non-standard ERM notation

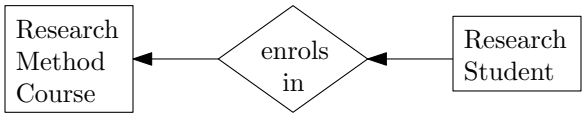
A — R → B	Arrows indicate direction of relation (A bears R to B)
A ○— P	Small circle indicates optionality (not all members of A have property P)
A_2 —⊂— A	Line with $\subset$ indicates subset (all members of A_2 are members of A)
A —∃→ R —∃→ B	Cardinality notation (see §5.2.1): Each member of A bears R to at least one member of B; equivalent to standard "$1 \dots N$" notation.
A —∀→ R —∀→ B	Cardinality notation (see §5.2.1): Each member of A bears R to each member of B.
A —S→ R —S→ B	Symmetric relation (see §5.2.2): If a member a of A bears R to a member b of B, then b must also bear R to a.
A — R	Relation symbol in attribute-type circle indicates reflexive relation (see §5.2.3): Members of A bear R to themselves.

5.2.1. New Cardinality Notation

Example possible unanswered question (Figure 2):

$\exists x \exists y[\text{ResearchStudent}(x) \wedge \text{MethodsCourse}(y) \wedge \neg\text{EnrolledIn}(x, y)]$ – Must research students enrol in *all* methods courses?

Figure 2 Problem: Inadequate Cardinality Notation.



We begin with the standard way of representing mandatory associations between classes, which fails to distinguish logically distinct ways in which two classes can be related.

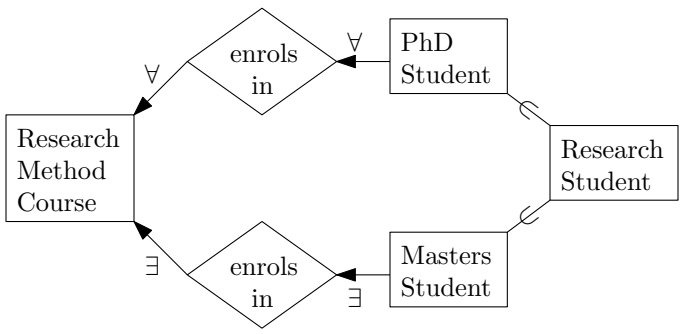
Suppose an ERM script indicates a relation holds between two classes. For example, Figure 2 shows a class *Research Student* bearing the relation *enrols in* to the class *Methods Course*. Given that the relation is mandatory for both classes (cf. §5.2.6 below for discussion of mandatory and optional associations), we can ask the following question in FO^2. Does every research student enrol in every methods course, or may a research student enrol only in some methods courses? (In symbols, we are asking whether $\forall x \forall y[\text{ResearchStudent}(x) \wedge \text{MethodsCourse}(y) \rightarrow \text{enrols}(x, y)]$ is true.) An ERM script cannot answer this question.

Our modeling grammar must be able to answer this sort of question if it is to produce precise 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*. 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*. (The "$\exists$" cardinality notation is effectively equivalent to the standard "$1 \ldots N$" notation.) 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*. Thus, in Figure 3 the *enrols in* association between *Masters Student* and *Methods Course* is $\exists$-type, whereas the *enrols in* association between *PhD Student* and *Methods Course* is $\forall$-type. (Note that "all" and "some" as used here are technical, logical notions, and they should not be confused with any vague, informal concept.)

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. Using standard cardinality notation to resolve the ambiguity 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. We would then be able only to indicate $\exists$-type cardinalities but not $\forall$-type cardinalities.

Past research in computer science has also provided formal treatments of cardinality (Thalheim 1992, Hartmann 1998, Dullea et al. 2003). Although that work and our work are similar (in relying on formal logic), the two programs of research are distinct and yet both are necessary. Our work

Figure 3 Solution: New Cardinality Notation.

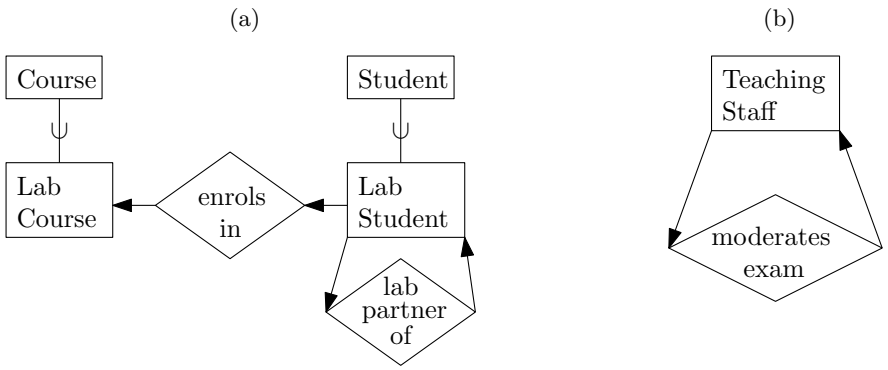


focuses on modeling organizational domains precisely, and the computer science work focuses on improved database design.

5.2.2. Symmetric and Asymmetric Relations

Example possible unanswered question (Figure 4): $\exists x \exists y[\text{LabPartner}(x, y) \wedge \neg\text{LabPartner}(y, x)]$ – If x is lab partner to y, must y also be lab partner to x?

Figure 4 Problem: Symmetric or Asymmetric Relations?



We need to be able to tell when an association between classes is symmetric. If a relation R is symmetric, then whenever a bears R to b, b also bears R to a. For example, the relation *lab partner of* in Figure 4a is symmetric: if Alice is Bob's lab partner, then Bob must be Alice's lab partner.

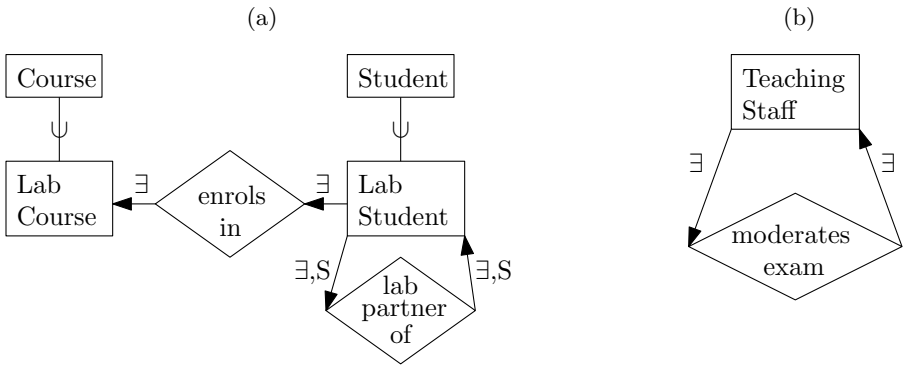
On the other hand, for an asymmetric relation R', whenever a bears R' to b, b does not bear R' to a. In Figure 4b, we have the relation *moderates exam*. This relation means that teaching staff members serve as second marker, or moderator, on each other's exams before marks are finalized. At the university represented in Figure 4b, if staff member a moderates the exam in staff member b's class, then b is prohibited from doing the same for a's class. The university imposes this constraint to protect the integrity of the moderation. Therefore, *moderates exam* is an asymmetric relation. The problem in Figure 4, however, is that symmetry and asymmetry are not distinguished.

Because we can express the statement that R is (a)symmetric in FO^2 (with the formula $\forall x \forall y[R(x, y) \rightarrow R(y, x)]$ for symmetry, or $\forall x \forall y[R(x, y) \rightarrow \neg R(y, x)]$ for asymmetry), we need to be able to distinguish symmetric from asymmetric relations in our scripts to achieve always-preciseness. We solve this problem by requiring that all relations be either symmetric or asymmetric. Moreover, they must be explicitly specified as such. Thus, the association *lab partner* in Figure

5a bears the annotation "S," indicating that it is symmetric; the association *moderates exam* in Figure 5b lacks this annotation, indicating that it is asymmetric.

Some relations are neither symmetric nor asymmetric, but we can resolve any such relation into symmetric and asymmetric component relations. If R is neither symmetric nor asymmetric, then define R_3 and R_4 as follows: a bears R_3 to b iff a bears R to b and b bears R to a; a bears R_4 to b iff a bears R to b and b does not bear R to a. Then R_3 will be symmetric, R_4 will be asymmetric, and the union of R_3 and R_4 is just the original relation R. Thus, by requiring relations to be either symmetric or asymmetric, we lose no expressive power.

Figure 5 Solution: Symmetric and Asymmetric Relations.

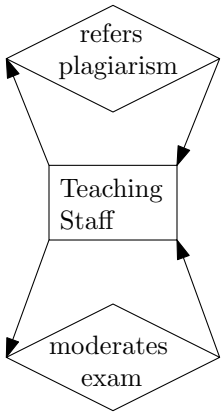


5.2.3. Reflexive Relations

Example possible unanswered question (Figure 6): $\exists x[\text{Moderates}(x, x)]$ – Can someone moderate their own exam marking?

FO^2 allows us to say that an individual bears a (binary) relation R to itself. It distinguishes this case from the case of an individual possessing a (unary) property P. For example, consider the class *Teaching Staff* in Figure 6. This class has two recursive associations on this class—namely, *moderates exam* and *refers plagiarism*. These associations represent two different tasks teaching staff perform for each other: first, staff *moderate* each other's exam marking, meaning that each staff member's marking is given to a second marker, or moderator, before being finalized; second, each teaching staff member has a designated staff member to whom all cases of plagiarism are to

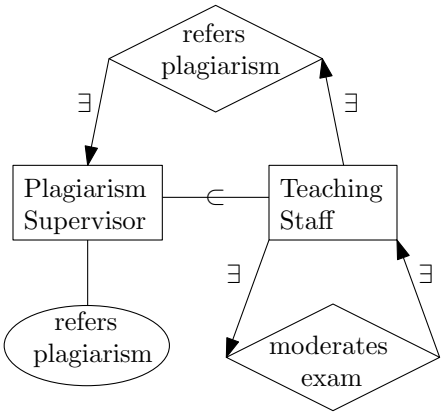
Figure 6 Problem: Reflexive Relations.



be reported. Only one of these associations represents a relation that teaching staff may bear to themselves. That is, a staff member may be in charge of handling cases of plagiarism in her own courses, in which case she will bear the relation *refers plagiarism* to herself. On the other hand, no staff member will moderate her own marking; the point of moderation is to have a second marker. ERM is unable to represent this difference.

Figure 7 shows how ERM-R solves this problem. In the revised diagram, a subclass *Plagiarism Supervisor* of *Teaching Staff* exists that has an attribute *refers plagiarism*. This representation indicates that members of this subclass report cases of plagiarism to themselves. More generally, in ERM-R diagrams, attributes that bear the same label as an association indicate that the relation denoted by the association is reflexive (on the class to which the attribute belongs). In Figure 7, no class bears an attribute labeled *moderates exam*, which indicates that the corresponding relation is irreflexive.

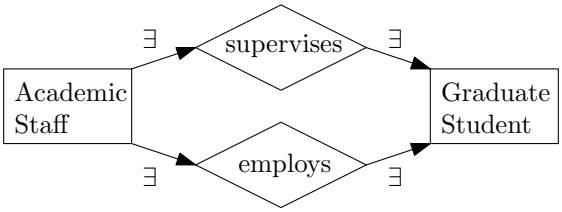
Figure 7 Solution: Reflexive Relations.



5.2.4. Individuation of Relations

Example possible unanswered question (Figure 8): $\exists x \exists y[\text{Supervises}(x, y) \wedge \text{Employs}(x, y)]$ – Can someone both supervise and employ the same person?

Figure 8 Problem: Unindividuated Relations.



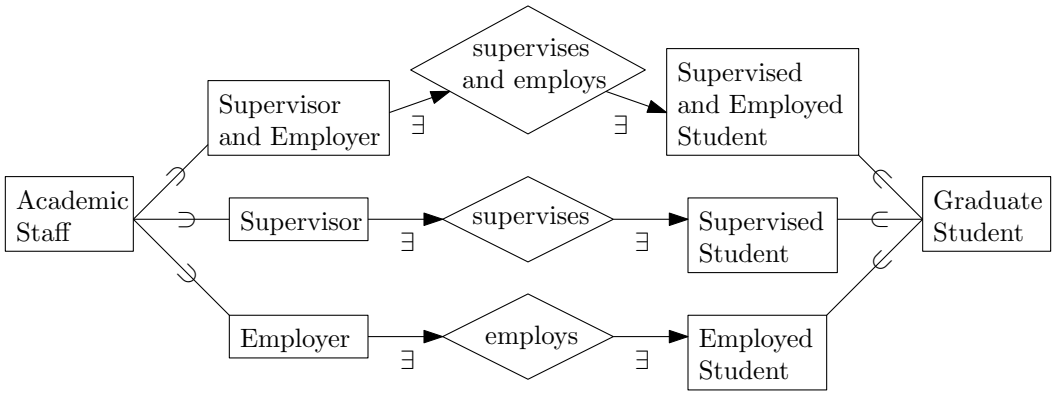
In Figure 8, two associations exist between *Academic Staff* and *Graduate Student*: both *supervises* and *employs* are ∃-type. Our question is whether the same research staff member can both supervise and employ the same research student. Figure 8 does not answer this question. It could be that all research staff must supervise at least one research student and employ at least one research student but that these students must be different (perhaps to avoid conflict of interest).[10] Nor can this ambiguity be resolved through the use of subclasses.

Our solution is to require that associations be individuated finely: a ∃-type association labeled "R_1" between A and B indicates that members of A bear R_1 *and no other relation* to members of

[10] Note that the same sort of ambiguity would not arise if we had a ∀-type association from A to B—that is, so that all members of A bear R_1 to *all* members of A, and likewise for R_2. In that case, the answer to the question is: yes, a must bear R_2 to b, because b is a member of B.

B. If we want to indicate that members of A bear R_1 and also R_2 to members of B, then we use an association labeled "R_1&R_2." If we want to indicate that members of A bear one or more of the relations R_1 and R_2 to members of B, we split A and B into the corresponding subclasses (those associated by R_1 alone, by R_2 alone, and by both R_1 and R_2), with their associations depicted explicitly. Figure 9, illustrates this approach where we introduce the new association *supervises & employs*.

Figure 9 Solution: Individuation of Relations.



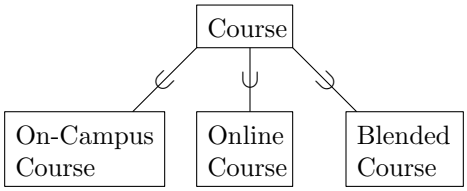
We also introduce three[11] subclasses of *Academic Staff* to ensure we express the following information. It is mandatory for each member of Academic Staff to supervise at least one Graduate Student and also to employ at least one Graduate Student. Academic Staff are subject to no further restrictions, however, on whom they may supervise or employ. In particular, it is not mandatory that they both supervise and employ any single Graduate Student. Thus, we cannot draw a mandatory association *supervises & employs* between *Academic Staff* and *Graduate Student*. Therefore, we introduce the subclass *Supervisor-Employer* for Academic Staff who supervise and employ one and the same Graduate Student and the subclasses *Supervisor* and *Employer* for, respectively, Academic Staff who supervise (employ) a Graduate Student whom they do not employ (supervise). We also introduce parallel subclasses of *Graduate Student* for the same reasons.

[11] We could have done the same job with two subclasses (as the classes *Supervisor* and *Employer* have the same members), but the extra subclass (a) does not change the script's meaning, and (b) increases clarity. We have also simplified the script by omitting indications of which subclasses are overlapping or disjoint and which subclasses are exhaustive of their superclass. See §5.2.5 below for further discussion.

5.2.5. Overlapping, Disjoint, and Exhaustive Subclasses

Example possible unanswered question (Figure 10): $\exists x[\text{OnlineCourse}(x) \wedge \text{BlendedCourse}(x)]$ – Can an online course also be a blended course?

Figure 10 Problem: Overlap, Disjointness, and Exhaustiveness of Subclasses.



Consider the ER script in Figure 10, which models the modes of course offerings in a university. The script tells us that the university offers courses in three modes: on-campus, online, and blended. It does not tell us, however, whether the three modes are disjoint (all courses are offered in one mode only) or overlapping (some courses are offered in more than one mode). Furthermore, it does not tell us whether these modes are exhaustive (are there other ways of offering a course, such as a massive open online course (MOOC)?). Such facts are expressible in FO^2, so we need to amend our fragment of ERM to ensure it is capable of generating consistent and precise scripts when questions are asked about relationships among subclasses and among subclasses and classes. Specifically, any script generated using the ERM-R must show for all subclasses represented in the script (a) which subclasses are disjoint and which subclasses are overlapping, and (b) whether subclasses are exhaustive or non-exhaustive of their superclass.

We need a solution to resolve any ambiguity about whether subclasses are disjoint or overlapping and exhaustive or non-exhaustive. Our solution is to ensure the bottom-level classes of a model (classes depicted as having no subclasses) are disjoint and exhaustive. That is, every individual of the domain belongs to exactly one bottom-level class. For other classes, we can tell whether they are disjoint or overlapping simply by checking whether they have any subclasses in common.

Different notations can be used to achieve this objective. We rely on a simple subclass notation in this paper, but other notations that explicitly denote overlapping, disjoint, and exhaustive

relationships among subclasses can also be used (e.g., Elmasri and Navathe 2014, pp. 247–249). Regardless of notation, the key objective (which we have not seen discussed explicitly in the *ontological* tradition of CM research) is to ensure *all* questions of class membership can be addressed, and this outcome can be achieved by ensuring all bottom-level classes are disjoint and exhaustive.

For example, consider the ERM-R script in Figure 11. Here we have four bottom-level classes: *Research-Only Staff*, *Research and Teaching Staff*, *Teaching-Only Staff*, and *Adjunct Staff*. Because these are bottom-level classes, they are disjoint. Furthermore, because they are the only bottom-level classes, they are exhaustive. Likewise, we can see that the two classes *Research Staff* and *Teaching Staff* are overlapping because they have the subclass *Research and Teaching Staff* in common. Also, they are not exhaustive of *Academic Staff* because that superclass has another subclass—namely, *Adjunct Staff*.

We do not suggest that the value of clarifying disjointness/overlap and exhaustiveness (or lack thereof) is unknown in the literature or in practitioner communities. Recall our central claim is not that each issue we highlight in these six subsections is a source of imprecision. Rather, our headline result is that *no other* issue results in imprecision. The lessons of this subsection are simply that an always-precise grammar must ensure that questions of disjointness and exhaustivity are answered and that the approach suggested in this section will achieve that objective.

Figure 11 Solution: Overlap, Disjointness, and Exhaustiveness of Subclasses.

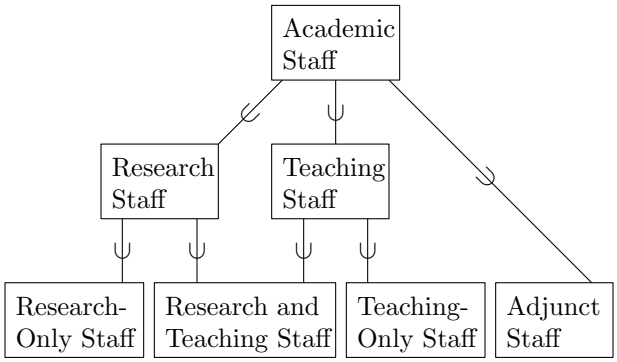
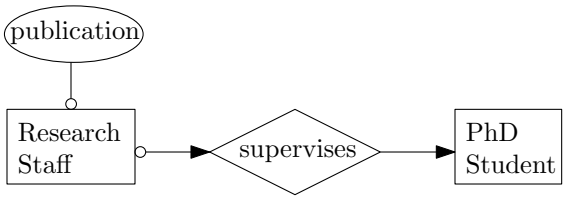


Figure 12 Problem: Optionality.



5.2.6. Optional Attributes and Associations

Example possible unanswered question (Figure 12): $\exists x[\neg \text{HasPublication}(x) \wedge \exists y \text{Supervises}(\text{x}, \text{y})]$

– Can research staff without publications supervise PhD students?

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, pp. 77–80) definition of laws. Their ontological analysis finds that some cases exist 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 understanding of and solution to this problem still eludes ontological analysis.

Logical analysis lets us characterize more precisely the problem with optionality, through the notion of precision as described in §3. When a grammar allows optional attributes or associations, it cannot be always-precise. The reason is that some scripts with optionality will be ambiguous between multiple, incompatible interpretations. *Information* about which of these interpretations is *lost* in such a script, to use Burton-Jones et al.'s terminology. In our terminology, such a script is

imprecise. 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 precise. Nonetheless, a grammar that includes optional attributes or associations will allow imprecise scripts to be generated.

To illustrate why optionality prevents always-preciseness, consider Figure 12. Here, *publication* is an optional property of the class *Research Staff*. Similarly, the relation *supervises* between the classes *PhD Student* and *Research Staff* is optional for the latter: some *Research Staff* do not *supervise* any *PhD Students*. These two instances of optionality, taken together, raise the question: Are *Research Staff* lacking a *publication* allowed to *supervise* a *PhD Student*? This question is unanswered in Figure 12's script.

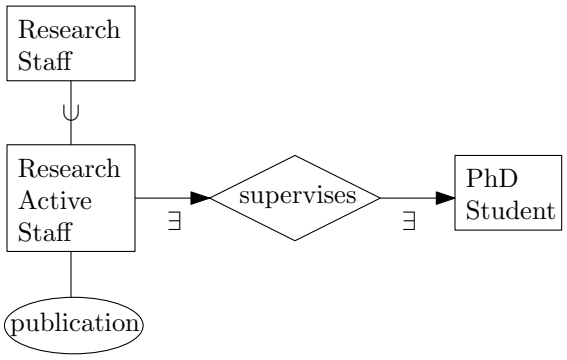
Multiple possible solutions exist to the problems that arise when optional attributes and associations are used. We could restrict the ways optional attributes/associations may be used. For example, to avoid the ambiguities just mentioned, we could prohibit optional attributes for classes with subclasses. This solution would be ungainly, however—many such restrictions would have to be introduced to eliminate all possible ambiguities. Thus, we recommend instead a general proscription against optionality. Our modified grammar ERM-R does not allow optional attributes or associations.

Without optional attributes or associations, we must use subclasses to indicate partiality—that some but not all members of a (super)class have some property or bear some relation. Thus, in Figure 13, for example, we represent the same situation represented in Figure 12. In the revised script, using subclasses instead of optionality, our previously unanswered question is now answered. We can see that the *Research Staff* with a *publication* and those who *supervise* a *PhD Student* are one and the same. Specifically, only the *Research Active Staff* have *publications*, and only they *supervise* the *PhD Students*.

In summary, our contribution to the discourse on optional properties is twofold. First, our logical analysis shows that disputes about whether the appropriate ontology has been used or a particular

ontology has been used correctly to analyse the nature of optional properties (e.g., Allen and March 2006, pp. 3–4) can be set aside. Our logical analysis shows that optional properties are problematic irrespective of the ontology used or way an ontology is used to assign them meaning. Second, our logical analysis shows that the question of whether optional properties are problematic from the perspective of generating precise semantics is not "an empirical question" nor one best resolved by "cognitive theories" (Allen and March 2006, p. 4). It is first and foremost a logical problem associated with generating imprecise CM scripts. For pragmatic reasons, conceptual modelers might still choose to use optional properties, but from the perspective of having precise semantics they should understand the potentially negative consequences of their choice.

Figure 13 Solution: Optionality Eliminated via Subclasses.



5.3. Summary and Discussion

These are the modifications to our fragment of the ERM grammar needed to produce the ERM-R grammar, for which our result holds (an always-precise grammar). Nonetheless, the focus of our logical analysis is not the individual defects *per se*. The lesson to be drawn is that the problems we identify with our targeted fragment of the ERM grammar are *the only problems* with that fragment. Moreover, the theorem proved in Appendix II tells us that addressing the problems identified in §§5.2.1–5.2.6 results in a grammar with *no further sources of imprecision*. As we presented these problems, we offered solutions to them. Our result shows that these solutions are sufficient to produce an always-precise 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.

Although other solutions exist to the six problems we identified, it is important to highlight the value of a logical approach over a purely ontological approach. In our view, for two reasons the traditional ontological approach could not have identified and resolved these problems satisfactorily. First, the common characteristic of all six problems is that inconsistencies and imprecision can arise in a script due to the way that grammatical constructs combine to make statements about a domain. The possible inconsistency and imprecision lie in the statements, not the individual constructs. Moreover, the problems with the statements are logical problems—problems of consistency and precision. Traditionally, ontological analysis has examined the clarity and completeness of individual constructs in a grammar, not statements. In cases where researchers have examine statements from an ontological perspective (as in the work of Evermann and Wand 2005a,b), the focus has naturally been on the ontological properties of the statements (such as ontological completeness and clarity) rather than logical properties. Each theory (logic and ontology) provides a lens that foregrounds some issues at the expense of others. For that reason, it is not surprising that the existing literature (which has largely taken an ontological perspective) has not identified the value of addressing all six problems identified in this paper or surfaced the more fundamental nature of some of the problems.

We also believe the ontological approach could not have addressed these issues *satisfactorily* because logic is more general than ontology. Recall that ontology presupposes logic. As a result, if researchers use an ontology that relies on first-order logic, potentially they could use that ontology to identify all six defects examined in this paper. However, doing so would amount to performing a logical analysis. The added value of the ontology lies in the specific constructs it offers over and above those offered by the logic that it presupposes. We argue that identifying and resolving these issues from a logical perspective alone is more satisfactory than doing so from the perspective of a given ontological theory. The reason is that adopting an ontological benchmark commits the

analyst to the additional claims made by that ontology. Logical analysis frees the researcher from this constraint. For instance, researchers can use the results of this paper regardless of whether they prefer Bunge's ontology or Searle's ontology. Indeed, they can use the results even if they prefer to use no ontological theory. This is not to say that ontological theories do not offer their own benefits. Logical analysis cannot offer insights into ontological completeness and clarity. Each theory offers distinct benefits—a dual perspective is needed.

6. Implications for Research

In light of the outcomes of our work, we believe that future research on the logical approach to evaluating the quality of the semantics of CM grammars and scripts could proceed in a number of directions. In the subsections below, we outline some possible topics that might be pursued in the context of the semantics, pragmatics, and syntax of CM grammars.[12]

6.1. Semantics

We have undertaken our logical analysis using a restricted subset of the ERM grammar—one that omits ternary and higher-order relations, non-dichotomic properties, and cardinality notation. Conducting a logical analysis of those aspects of the ERM grammar that we have omitted from our analysis would be valuable as a basis for determining whether an expanded ERM-R grammar is capable of ensuring precise scripts.

In the interests of brevity and our belief that our results for precision are more interesting and innovative, we have not developed the logical analysis of the ERM grammar for consistency. As restrictions on the ERM-R grammar are relaxed, however, evaluating whether an expanded ERM-R grammar is capable of ensuring consistent scripts is likely to prove more challenging. Potentially more interesting and innovative results will emerge.

Just as researchers in the ontological tradition have investigated the applicability of multiple ontological benchmarks, further research could be conducted using different logical benchmarks

[12] We recognize that semantics, pragmatics, and syntax are inextricably intertwined and researchers may therefore wish to combine ideas from each approach discussed below.

to evaluate the semantic quality of CM grammars. Whereas we used FO^2, other logics could be investigated, such as unrestricted FOL, description logic and other modal logics, and even nonstandard logics (e.g., paraconsistent logics). Use of these logics may produce different insights about the ability of CM grammars to generate consistent and precise scripts.

The relationship between the ontological and logical dimensions of the quality of semantics bears investigating. We took one fairly well-accepted view on the relationship between ontology and logic, but other views on this matter exist (Hofweber 2014). Researchers could take a different perspective from us, or even investigate whether a unifying theory of both types of semantic quality could be developed. Interesting theoretical work has also been done on the kinds of questions that can be asked about a domain that could help IS researchers to see new ways of thinking about and questioning real-world domains (Anderson and Belnap 1975, Belnap 1982).

The semantics of 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 always produce precise scripts can be assessed. The extent to which our results might impact the design of new CM grammars could also be investigated (e.g., through discussions with and working with the designers of CM grammars).

6.2. Pragmatics

Just as researchers in the ontological tradition have made progress by testing ontological predictions empirically (Burton-Jones et al. 2009), work could be done to test empirically whether different stakeholders benefit from having more consistent and more precise CM grammars and scripts. For instance, the extent to which such grammars and scripts result in better system designs and enable end-users to better query a database could be investigated.

Researchers could also examine whether the usefulness of consistent and precise CM grammars and scripts varies across contexts. In this regard, some researchers following the ontological

approach (e.g., Bodart et al. 2001) have found ontologically clear scripts are more useful when stakeholders require a more detailed understanding of a domain rather than a rough understanding of the domain. Others (e.g., Burton-Jones et al. 2012) have found that the positive effects of clearer (and more precise) scripts on stakeholder understanding weaken as the scripts become more complex.

Similar contingencies are likely to apply to precise CM scripts and always-precise CM grammars. For instance, precise scripts and always-precise grammars probably will be most useful in contexts where stakeholders are motivated to avoid misunderstandings because of their potential to result in serious negative consequences—for example, in high-reliability organizations (Roberts 1990). In contexts in which misunderstandings are less likely to occur or where they have milder effects, stakeholders may not see the benefits of having precise scripts and always-precise grammars—less-precise scripts and grammars may suffice or even be preferable because of their ease of use.

Just as researchers in the ontological tradition have designed CM methods (e.g., Evermann and Wand 2005b) and tools (e.g., Wand et al. 2008) based on ontological insights, researchers could create new methods and tools (or amend existing methods and tools) based on the results of logical analysis (such as those presented in this paper) to assist stakeholders to do higher-quality CM work. Indeed, there is a long tradition of creating tools to help modelers create models with more accurate and complete semantics (Khatri et al. 2006, Sugumaran and Storey 2006).

Creating better methods and tools seems a particularly promising avenue for research using logical analysis given the computable properties of some logics. Possible links between the logical approach outlined in this paper and the logical approach carried out in computer science research could then be pursued (e.g., Fillotrani et al. 2012).

6.3. Syntax

Researchers could express the modified ERM semantics we articulated in this paper using different syntactic forms. They could then test empirically whether some forms (e.g., those following the physics of notations, Moody 2009) are easier to understand.

In §5.2.5, we pointed out that various forms of syntax have been described in prior literature to show whether subclasses in conceptual models are overlapping or disjoint and exhaustive or non-exhaustive. We used a more parsimonious syntax that motivates a modeling approach requiring bottom-level subclasses in a level structure of classes to be disjoint and exhaustive. Empirical work could be conducted to see whether the syntax and the modeling approach it motivates leads to better representations of domain phenomena and better system designs.

7. Implications for Practice

The practical implications of our analysis are quite clear. We offer and validate (by proof) a logically "always-precise" fragment of ERM, ERM-R. Although empirical work is required to test its practical utility, we have shown that it is semantically more precise than ERM. Accordingly, we advocate the use of ERM-R over ERM for practitioners who work in areas where precise semantics are crucial. Moreover, we showed that modelers must be mindful of the six sources of imprecision we identified with the ERM grammar when they use it to prepare CM scripts.

Some practitioners may be aware of the debates on ontologies in the IS literature and have firm preferences about which should be adopted (e.g., Bunge's ontology or Searle's ontology). Other practitioners may be unaware of, or see little value in, such debates. Nonetheless, because FO^2 underwrites many ontologies (including Bunge's ontology and Searle's ontology) and depends on none of them, practitioners can adopt ERM-R and our recommendations for using ERM-R regardless of their views about ontological issues.

While the focus of our logical analysis has been the ER grammar, the problems we have identified also apply to other CM grammars. For instance, the UML grammar does not distinguish between symmetric and asymmetric associations. The ORM grammar does make this distinction (Halpin 1996), but it does not distinguish between "all" ($\forall$) associations and "some" ($\exists$) associations. Where such associations are important phenomena in the domain to be modeled, UML and ORM will not generate precise scripts. Practitioners who use CM grammars other than the ER grammar can use our results, therefore, to evaluate whether the grammar they employ is likely to generate

imprecise scripts. They may also be able to use the modifications we have made to the ER grammar as a basis for improving the precision of scripts generated via the CM grammar they employ.

Finally, while we focused this paper on implications for CM, an interesting practical question is whether the paper has broader implications for the design and use of information systems. We did not address this issue in the paper for reasons of scope, but we have conducted extensive interviews with a database designer with over 20 years of experience on this topic. Three insights stemmed from these discussions.

First, the database designer found that creating a relational database based on our solutions to all six issues described in §5 was easy. Moreover, he could attest to the practical relevance of all six issues we identified in light of his experience (e.g., in designing databases for the Department of Education in his state).

Second, rather than developing the database to reflect a particular script exactly, he would work with an application designer to decide how best to reflect the real-world semantics in the information system (while also considering other design decisions such as speed). In other words, the scripts would inform but not determine his database designs.

Third, as many systems analysts have noted (e.g., Yourdon 1989), the designer confirmed the conceptual issues we addressed in ERM-R cannot be solved through application or database design. No matter how good the application and database, users querying the database will be unable to answer certain questions (yes/no) if they do not have precise knowledge about the domain. If users have access to precise CM scripts, however, they will then be able to query the database to discern whether the data conforms with the real-world phenomena reflected in the script. In future work, we plan to carry out field work with users to discern their ability to use precise scripts as a means of detecting defects in a database.

8. Conclusions

Semantics lies at the heart of conceptual modeling. It provides the connection between our representations (scripts) and the reality they are supposed to represent. It underpins the meaning that stakeholders ascribe to representations.

Current approaches to evaluating the semantics of CM grammars and scripts rely primarily on ontology with little, if any, engagement with logic. Ontology gives us only half the picture, however, in evaluating the quality of the semantics in a grammar or script. It tells us what is in the reality we aim to represent with a CM script, but it cannot tell us how our scripts and grammars can represent that reality. A grammar with wholly adequate semantics must allow two outcomes to be achieved: (a) representation of all and only the relevant phenomena in the target domain (ontology); and (b) representation of the phenomena consistently and precisely (logic).

In this paper, we have outlined the differences between an ontological approach and a logical approach to evaluating the quality of the semantics of CM grammars and scripts. Moreover, we have identified an issue in the design of CM grammars—namely, precision—that cannot be treated adequately using a purely ontological approach. We have also characterized precision using logical analysis.

Finally, we have given a rigorous application of our logical analysis of precision to provide concrete recommendations for the redesign of and disciplined use of the ERM grammar to avoid imprecise (ambiguous) scripts. In particular, we have identified the need for a program of logical investigation into the quality of the semantics of CM grammars and scripts. We have initiated that research program both in theory and in application.

Acknowledgments

We are indebted to the Senior Editor, Associate Editor, and three Reviewers for their helpful comments on earlier versions of this paper. We also benefited from advice from Vern Bawden, Denis Duncan, Marta Indulska, Wolfgang Maass, Rachel Pottinger, Veda Storey, Wei Sun, Yair Wand, as well as participants in presentations at SIGSAND 2012 and ICIS 2013. We acknowledge financial support from the Sauder School of Business, UBC, UQ Business School, and the Australian Research Council ARC DP110104386.

References

Ågerfalk P (2010) Getting pragmatic. *European Journal of Information Systems* 19:251–256.

Allen GN, March ST (2006) A critical assessment of the Bunge-Wand-Weber ontology for conceptual modeling. *Workshop on Information Technologies and Systems*, 25–30 (Milwaukee, WI).

Ambler SW (2002) *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process* (Wiley).

Anderson AR, Belnap ND (1975) *Entailment: The Logic of Relevance and Necessity*, volume 1 (Princeton: Princeton University Press).

Belnap ND (1982) Display logic. *Journal of Philosophical Logic* 11:375–417.

Bera P, Burton-Jones A, Wand Y (2014) How semantics and pragmatics interact in understanding conceptual models. *Information Systems Research* 25(2):401–41.

Bodart F, Patel A, Sim A, Weber R (2001) Should optional properties be used in conceptual modeling? a theory and three empirical tests. *Information Systems Research* 12(4):383–405.

Bunge M (1977) *Ontology I: The Furniture of the World*, volume 3 of *Treatise on Basic Philosophy* (New York: D. Reidel).

Burton-Jones A, Clarke R, Lazarenko K, Weber R (2012) Is use of optional attributes and associations in conceptual modeling always problematic? theory and empirical tests. *International Conference on Information Systems*, 1–16 (Orlando, Florida).

Burton-Jones A, Wand Y, Weber R (2009) Guidelines for empirical evaluations of conceptual modeling grammars. *Journal of the Association for Information Systems* 10(6):495–532.

Calvanese D, Lenzerini M, Nardi D (1998) Description logics for conceptual data modeling. Chomicki J, Saake G, eds., *Logics for Databases and Information Systems*, 229–264 (Kluwer).

Church A (1936) An unsolvable problem of elementary number theory. *American Journal of Mathematics* 58(2):345–363.

Currim F, Ram S (2006) Understanding the concept of "completeness" in frameworks for modeling cardinality constraints. *Frameworks for Modeling Cardinality Constraints, 16th Annual Workshop on Information Technolgies & Systems (WITS)*, 31–36.

Currim F, Ram S (2012) Modeling spatial and temporal set-based constraints during conceptual database design. *Information Systems Research* 23(1):109–28.

Dawson L, Swatman P (1999) The use of object-oriented models in requirements engineering: A field study. *International Conference on Information Systems*, 260–273 (Charlotte, NC).

Dullea J, Song IY, Lamprou I (2003) An analysis of structural validity in entity-relationship modeling. *Data & Knowledge Engineering* 47:167–205.

Dunn CL, McCarthy WE (1997) The REA accounting model: Intellectual heritage and prospects for progress. *Journal of Information Systems* 11(1):31–51.

Elmasri R, Navathe SB (2014) *Fundamentals of Database Systems, New International Edition* (Boston: Addison-Wesley), 6th edition.

Evermann J, Wand Y (2005a) Ontology based object-oriented domain modelling: Fundamental constructs. *Requirements Engineering* 10:146–60.

Evermann J, Wand Y (2005b) Toward formalizing domain modeling semantics in language syntax. *IEEE Transactions on Software Engineering* 31(1):21–37.

Fillotrani P, Franconi E, Tessaris S (2012) The ICOM 3.0 intelligent conceptual modelling tool and methodology. *Semantic Web Journal* 3(3):293–306.

Fonseca F (2007) The double role of ontologies in information science research. *Journal of the American Society for Information Science and Technology* 58(6):786–93.

Franconi E (2004) Using ontologies. *IEEE Intelligent Systems* 19(1):76–7.

Gemino A, Wand Y (2005) Complexity and clarity in conceptual modeling: Comparison of mandatory and optional properties. *Data & Knowledge Engineering* 55:301–26.

Grädel E, Kolaitis PG, Vardi MY (1997) On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic* 3(1):53–69.

Green P, Rosemann M, Indulska M, Recker J (2011) Complementary use of modeling grammars. *Scandinavian Journal of Information Systems* 23(1):1–27.

Greenspan SJ, Borgida A, Mylopoulos J (1986) A requirements modeling language and its logic. *Information Systems* 11(1):9–23.

Gregersen H, Jensen CS (1999) On the ontological expressiveness of temporal extensions to the entity-relationship model. Chen PP, Embley DW, Kouloumdjian J, Liddle SW, Roddick JF, eds., *Advances in Conceptual Modeling*, volume 1727 of *Lecture Notes in Computer Science*, 110–121 (Springer).

Hadar I, Soffer P, Kenzi K (2014) The role of domain knowledge in requirements elicitation via interviews: An exploratory study. *Requirements Engineering* 19(2):143–59.

Halpin TA (1996) Business rules and object role modeling. *Database Programming and Design* 9:66–72.

Hartmann S (1998) On the consistency of int-cardinality constraints. Ling TW, Ram S, Lee ML, eds., *Proceedings of the 17th International Conference on Conceptual Modeling (ER '98)*, 150–163 (Singapore: Springer).

Henkin L (1967) *Logical Systems Containing a Finite Number of Symbols*, volume 21 of *Séminaire de Mathématiques Supérieures* (Montreal: Presses de l'Université de Montréal).

Hitzler P, Krotzsch M, Parsia B, Patel-Scneider PF, Rudolph S, eds. (2012) *OWL 2 Web Ontology Language Primer* (W3C Recommendation), 2nd edition, http://www.w3.org/TR/owl2-primer/.

Hofweber T (2014) Logic and ontology. Zalta EN, ed., *The Stanford Encyclopedia of Philosophy*, Fall 2014 edition, http://plato.stanford.edu/archives/fall2014/entries/logic-ontology/.

Jeusfeld MA, Jarke M, Mylopoulos J, eds. (2009) *Metamodeling for Method Engineering* (Cambridge, MA: MIT Press).

Khatri V, Ram S, Snodgrass RT (2006) On augmenting database design-support environments to capture the geo-spatio-temporal data semantics. *Information Systems* 31:98–133.

Lemieux V, Limonad L (2011) What "good" looks like: Understanding records ontologically in the context of the global financial crisis. *Journal of Information Science* 37(1):29–39.

Ludwig K (2007) Foundations of social reality in collective intentional behavior: Essays on John Searle's social ontology. Tsohatzidis SL, ed., *Intentional Acts and Institutional Facts*, 49–71 (Dordrecht, The Netherlands: Springer).

Moody DL (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.

Mortimer M (1975) On languages with two variables. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 21:135–140.

Mylopoulos J, Borgida A, Jarke M, Koubarakis M (1990) Telos: Representing knowledge about information systems. *ACM Transactions on Information Systems* 8(4):325–62.

Opdahl AL, Henderson-Sellers B (2002) Ontological evaluation of the UML using the Bunge-Wand-Weber model. *Software and Systems Modeling* 1:43–67.

Pitts MG, Browne GJ (2004) Stopping behavior of systems analysts during information requirements elicitation. *Journal of Management Information Systems* 21(1):203–26.

Ram S, Khatri V (2005) A comprehensive framework for modeling set-based business rules during conceptual database design. *Information Systems* 30:89–118.

Recker J, Indulska M, Rosemann M, Green P (2010) The ontological deficiencies of process-modeling in practice. *European Journal of Information Systems* 19(5):501–25.

Recker J, Rosemann M, Green P, Indulska M (2011) Do ontological deficiencies in modeling grammars matter? *MIS Quarterly* 35(1):57–79.

Reiter R (1978) On closed world data bases. Gallaire H, Minker J, eds., *Logic and Data Bases*, 119–140 (New York: Plenum Press).

Roberts KH (1990) Managing high reliability organizations. *California Management Review* 32(4):101–13.

Searle JR (1995) *The Construction of Social Reality* (New York: The Free Press).

Sheth A (1997) Panel: Data semantics: What, where and how? Meersman R, Mark L, eds., *Database Applications Semantics*, 601–610 (Boston, MA: Springer US).

Shoenfield JR (1967) *Mathematical Logic* (Reading, MA: Addison-Wesley), reprinted 2000 by Association for Symbolic Logic/A.K. Peters, Natick, MA.

Steel TB (1986) A minimal conceptual schema language for life, the universe, and everything. Steel TB, Meersman R, eds., *Database Semantics*, 255–284 (Elsevier Science Publishers, IFIP, North Holland).

Sugumaran V, Storey VC (2006) The role of domain ontologies in database design: An ontology management and conceptual modeling environment. *ACM Transactions on Database Systems* 31(3):1064–94.

Thalheim B (1992) Fundamentals of cardinality constraints. Pernul G, Tjoa AM, eds., *Proceedings of the 11th International Conference on the Entity-Relationship Approach (ER '92)*, 7–23 (Karlsruhe, Germany: Springer-Verlag).

Thalheim B (1993) Foundations of entity-relationship modeling. *Annals of Mathematics and Artificial Intelligence* 7:197–256.

Turing AM (1936) On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society (series 2)* 42:230–265.

Wand Y, Storey VC, Weber R (1999) An ontological analysis of the relationship construct in conceptual modeling. *ACM Transactions on Database Systems* 24(4):494–528.

Wand Y, Weber R (1993) On the ontological expressiveness of information systems analysis and design grammars. *Journal of Information Systems* 3:217–237.

Wand Y, Weber R (1995) On the deep structure of information systems. *Information Systems Journal* 5:203–23.

Wand Y, Woo C, Wand O (2008) Role and request based conceptual modeling – a methodology and a case tool. *Proceedings of CAiSE'08 Forum*, 77–80 (Montpellier, France).

Wastell DG (1996) The fetish of technique: Methodology as a social defence. *Information Systems Journal* 6:25–40.

Weber R, Zhang Y (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* (Englewood Cliffs, NJ: Prentice Hall).

zur Muehlen M, Recker J, Indulska M (2007) Sometimes less is more: Are process modeling languages overly complex? *EDOC Conference Workshop, 2007. EDOC '07. Eleventh International IEEE*, 197–204.

Appendix I: Ontological Criteria for Evaluating the Semantics of Conceptual Modeling Grammars

To evaluate the ontological clarity and completeness of a conceptual modeling (CM) grammar, the grammar's constructs are mapped against the constructs in a benchmark ontology. Wand and Weber (1993, pp. 228–233) argue three conditions undermine *ontological clarity*:

- *Construct overload*: A single grammatical construct maps to two or more ontological constructs.
- *Construct excess*: A grammatical construct does not map to any ontological construct.
- *Construct redundancy*: Two or more grammatical constructs map to a single ontological construct.

Wand and Weber (1993, pp. 226–228) argue that *ontological completeness* is undermined when *construct deficit* exists—an ontological construct is not represented by any grammatical construct.

Our notion of ontological clarity differs in two ways from Wand and Weber's notion of ontological clarity. First, we do not see construct redundancy as undermining ontological clarity. Even though two or more grammatical constructs map to a single ontological construct, the meaning of each redundant construct is still clear. Whether the redundant constructs cause confusion among users of the grammar is an issue of pragmatics and not semantics.

Second, our concept of an always ontologically clear grammar is new. The traditional view of ontological clarity focuses on creating a mapping between individual constructs in a grammar and individual constructs in an ontological benchmark. An always-clear grammar goes further to consider the grammar's production rules. In an always-clear grammar, not only is each construct clear, but also the grammar's production rules will not allow unclear combinations of constructs in the grammar to be generated. We are not aware of any research on this property of grammars to date, but it could be viewed as an extension of the line of work begun by Evermann and Wand (2005a, p. 148).

Note that ontological completeness applies only to grammars and not scripts. The reason is that we cannot know if a script includes all the grammatical constructs needed to represent all relevant ontological constructs in a domain without knowing which ontological constructs are relevant in the domain description. The latter is a pragmatic issue that lies outside the scope of ontological analysis.

Note, also, that the four ontological criteria for evaluating CM grammars are all independent of each other. The reason is that each criterion is based upon a different outcome of the mapping between the set of constructs in a CM grammar and the set of constructs in a benchmark ontology.

Appendix II: Formal Results

In this Supplementary Appendix, we provide a rigorous proof that the grammar ERM-R is always-precise. In the main paper, we used the lay term "precise" for what logicians mean by "complete" to avoid confusion with the latter term's more usual meaning in the conceptual modeling literature. However, we expect readers of this Appendix to be more familiar with the logician's terminology—and we refer here to work by logicians—so we adjust our vocabulary accordingly: wherever we refer to "completeness" in this Appendix, we mean (a) what logicians mean by "completeness" and (b) what we mean by "preciseness" in the main paper.

Syntax

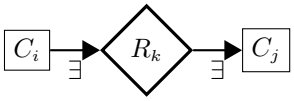
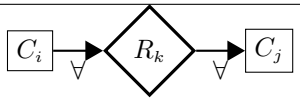
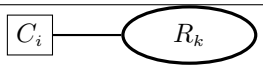
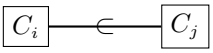
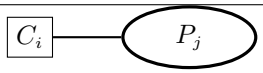
ERM-R is a visual, two-dimensional language. Nonetheless, we can translate scripts in ERM-R to a linear language (like English or FOL), which greatly simplifies the proofs to follow. We proceed by producing a linear "model-description" listing the elements of a visual ER diagram and their interconnections. This process loses none of the information contained in the ER diagram. Thus, we can use model-descriptions as substitutes for ERM-R scripts. Our scripts are always finite, so we can replace them with sentences giving instructions for constructing the grammar: "There are boxes labeled 'C_1' and 'C_2'; there is a line labeled 'R_1' from the box labeled 'C_1' to the box labeled 'C_2'; ...." By giving a syntax for the "language" of such translations—by giving rules determining what is and what is not an acceptable description of an ERM-R script—we effectively give a syntax for the original, untranslated language.

Table 4 gives a synopsis of our modified grammar ERM-R. The first column contains a list of visual constructs of ERM-R. The second column contains the translation of each construct into our language of model-descriptions. The third and fourth columns contain the informal and formal semantics intended for each construct.

We give a syntax for ERM-R by giving a syntax for model-descriptions in Backus-Naur form (BNF).

$$
\begin{aligned}
\text{< description >} ::= {} & \text{< description >; < description > |} \\
& \text{< class >} \rightarrow_{\exists} \text{< relation >< class > |} \\
& \text{< class >} \rightarrow_{\forall} \text{< relation >< class > |} \\
& \text{< class >} \subseteq \text{< class > | < class >< property >} \\
& \text{< class >< relation >}
\end{aligned}
$$

Table 4 Vocabulary elements of ERM-R.

Visual construct	Linear description	Informal semantics	Formal semantics
	$C_i \rightarrow_\exists R_k C_j$	All members of C_i bear R_k to at least one member of C_j, and vice versa.	If $x \in C_i$ then $\exists y \in C_j$ with $R_k xy$; if $y \in C_j$ then $\exists x \in C_i$ with $R_k xy$
	$C_i \rightarrow_\forall R_k C_j$	All members of C_i bear R_k to all members of C_j.	If $x \in C_i$ and $y \in C_j$ then $R_k xy$
	$C_i R_k$	Everything in C_i bears R_k to itself.	If $x \in C_i$ then $R_k xx$
	$C_i \subseteq C_j$	C_i is a subset of C_j.	$C_i \subseteq C_j$ (If $x \in C_i$ then $x \in C_j$)
	$C_i P_j$	Everything in C_i has P_j.	If $x \in C_i$ then $P_j x$

$$< \text{class} > ::= C_1 | C_2 | \ldots | C_n$$

$$< \text{relation} > ::= R_1 | R_2 | \ldots | R_m$$

$$< \text{property} > ::= P_1 | P_2 | \ldots | P_l$$

We design the syntax here for compactness rather than readability. Some brief comments on the syntax are in order. Semicolons conjoin component sentences of a description. When two classes are connected by a relation, we need to know what the relation is, its direction, and whether it is a ∀-type or ∃-type arrow. ("$\rightarrow_\exists$" or "$\rightarrow_\forall$"). Our description syntax does not allow sentences of the form, "There is a box labeled 'C_1'." This is because we assume that all classes will have some attributes—either a connection to another class via subclasshood or a relation arrow, or some intrinsic attribute of its members. Therefore, all classes will appear in one of those sentences in a model's description, making sentences of the form, "There is a box labeled 'C_1'," superfluous. From a syntactic perspective, there are no further constraints on the scripts (diagrams) we can produce: any combination of attributes, relations, and subclass relations is acceptable.

We can now define some syntactic terms that will be useful in giving a semantics below. We say that a class C_i is a *subclass* of a class C_j, and C_j is a *superclass* of C_i, and write $C_i \subseteq C_j$, iff the model-description contains, for some (finite) sequence of classes $C_1, \ldots, C_n$, the sentences "$C_i \subseteq C_1$," "$C_1 \subseteq C_2$," ..., and "$C_n \subseteq C_j$." By convention, we consider each class to be a subclass (and a superclass) of itself. We say that C_i is a *proper subclass* of C_j, and write $C_i \subset C_j$, iff C_i is a subclass of C_j and C_j is not a subclass of C_i. If C_i is a subclass of C_j but not a proper subclass, then we write $C_i = C_j$. Say a class is *basic* iff it has no (non-empty) proper subclasses and *non-basic* otherwise. Note that it is possible to have basic classes C_i and

C_j with $i \neq j$ but with $C_i = C_j$. Finally, say a class C_j is an *immediate subclass* of a class C_i iff $C_j \subset C_i$ and there is no class C_k such that $C_j \subset C_k \subset C_i$.

To deal conveniently with symmetric relations, we also add some technically superfluous sentences to our model-descriptions. If R_k is symmetric, and the model-description contains a sentence "$C_i \rightarrow_\exists R_k C_j$," then we add the sentence "$C_j \rightarrow_\exists R_k C_i$". Likewise, if the model-description contains a sentence "$C_i \rightarrow_\forall R_k C_j$," then we add the sentence "$C_j \rightarrow_\forall R_k C_i$". These additions have no material impact on the semantics of the original conceptual model, but they simplify the statement of the semantics in the next section.

Semantics

We give a semantics for ERM-R by giving a translation of any given model in ERM-R to a finitely axiomatized theory of FO^2. Because, as a fragment of FOL, FO^2 has a well-defined semantics (see, e.g., Shoenfield 1967), this translation amounts to a semantics for ERM-R.

Given a conceptual modeling script M in ERM-R, we build its translation into FOL, T_M, that we sometimes call "the (first-order) theory of M," as follows. The language $\mathcal{L}$ of the theory has as its nonlogical symbols finitely many unary predicates $C_1, C_2, \ldots$, and $P_1, P_2, \ldots$, and finitely many binary predicates $R_1, R_2, \ldots$; these are, respectively, exactly those class labels, attribute labels, and relation labels that occur in M. In general, $C_i x$ is to be intepreted as saying that x is a member of class C_i; $P_j x$ as saying that x has property P_j; and $R_k xy$ as saying that x bears relation R_k to y. A guiding principle in building T_M is that *the absence of a claim is a claim of absence*: for example, if a box C_1 does not have a circle labeled P_1 (where P_1 does occur somewhere in the script), then we interpret the script as claiming that the members of the class C_1 do not possess the property P_1. We now build the axioms of the theory T_M.

We have an axiom of exhaustivity, saying that no objects exist outside of the classes depicted in the model. It is enough to say that no objects exist outside the basic classes. So, where $C_{i_1}, \ldots, C_{i_j}$ are all of the basic classes, the exhaustivity axiom is:

$$\vdash_{T_M} C_{i_1} x \vee \ldots \vee C_{i_j} x$$

Next we add axioms of non-triviality, saying that the classes depicted are non-empty. Again, it is enough to say that the basic classes are non-empty. For each basic class C_i, we add:

$$\vdash_{T_M} \exists x C_i x$$

We also need axioms to say which of the basic classes are disjoint. For each pair of basic classes C_i and C_j with $i \neq j$, we add either a class equivalence or a class exclusion axiom. If $C_i = C_j$, then we add a class equivalence axiom:

$$\vdash_{T_M} C_i x \leftrightarrow C_j x$$

Otherwise, we add a class exclusion axiom:

$$\vdash_{T_M} \neg(C_i x \wedge C_j x)$$

Likewise, we need relation exclusion axioms, to capture the conventional constraint on relations. For each pair of distinct relation symbols R_i and R_j, we add both of the following:

$$\vdash_{T_M} \neg(R_i xy \wedge R_j xy)$$

$$\vdash_{T_M} \neg(R_i xy \wedge R_j yx)$$

Similarly, according to whether R_k is symmetric or asymmetric, we add either

$$\vdash_{T_M} R_k xy \rightarrow R_k yx$$

or

$$\vdash_{T_M} R_k xy \rightarrow \neg R_k yx$$

Now, we characterize non-basic classes by listing the subclasses of which they are composed. If C_i is non-basic, and its immediate subclasses are $C_{j_1}, \ldots, C_{j_k}$, then we add the following composition axiom:

$$\vdash_{T_M} C_i x \leftrightarrow C_{j_1} x \vee \ldots \vee C_{j_k} x$$

Thus, a non-basic class is characterized by the basic subclasses of which it is ultimately composed.

Finally, we add characterization axioms, giving the properties attributed to members of each basic class. For each basic class C_i, the characterization axiom is:

$$\vdash_{T_M} C_i x \rightarrow \Phi_i x \wedge \neg \Psi_i x,$$

where $\Phi_i x$ and $\Psi_i x$ are to be defined as follows.

To define the formulas $\Phi_i x$ and $\Psi_i x$ that do the real work of characterizing a basic class C_i, we must look at the relations and properties attributed by the model to members of C_i. $\Phi_i x$ will be a conjunction listing the properties that members of C_i must have, and $\Psi_i x$ will be a disjunction listing the properties that members of C_i lack. We build $\Phi_i x$ and $\Psi_i x$ by going through all the nonlogical symbols of $\mathcal{L}$ other than class symbols and checking whether and how they apply to the class C_i in the model.

For each predicate symbol P_k, if the model-description contains a sentence "$C_j P_k$," where C_j is any superclass of C_i, then add $P_k x$ as a conjunct of $\Phi_i x$; otherwise, add the same formula as a disjunct of $\Psi_i x$.

For each relation symbol R_k, we have several cases to check. If the model-description contains a sentence "$C_j R_k$," where C_j is any superclass of C_i, then add $R_k xx$ as a conjunct of $\Phi_i x$; otherwise, add the same formula as a disjunct of $\Psi_i x$.

If the model-description contains a sentence "$C_j \rightarrow_\forall R_k C_g$," where C_j is any superclass of C_i, then add $\forall y(C_l y \rightarrow R_k xy)$ as a conjunct of $\Phi_i x$; otherwise, add the same formula as a disjunct of $\Psi_i x$. If the model-description contains a sentence "$C_g \rightarrow_\forall R_k C_j$," where C_j is any superclass of C_i, then add $\forall y(C_g y \rightarrow R_k yx)$ as a conjunct of $\Phi_i x$; otherwise, add the same formula as a disjunct of $\Psi_i x$.

If the model-description contains a sentence "$C_j \rightarrow_\exists R_k C_g$," where C_j is any superclass of C_i, and it does not contain any sentence "$C_j \rightarrow_\forall R_k C_h$," where C_h is a superclass of C_g, then add $\exists y(C_g y \wedge R_k xy)$ and $\exists y(C_g y \wedge \neg R_k xy)$ as conjuncts of $\Phi_i x$; otherwise, add the first formula as a disjunct of $\Psi_i x$.

If the model-description contains a sentence "$C_g \rightarrow_\exists R_k C_j$," where C_j is any superclass of C_i, and it does not contain any sentence "$C_h \rightarrow_\forall R_k C_j$," where C_h is a superclass of C_g, then add $\exists y(C_g y \wedge R_k yx)$ and $\exists y(C_g y \wedge \neg R_k yx)$ as conjuncts of $\Phi_i x$; otherwise, add the first formula as a disjunct of $\Psi_i x$.

This completes the definition of $\Phi_i x$ and $\Psi_i x$, and thereby also completes the definition of T_M.

Always-preciseness

To prove the following result, we extend T_M to a new theory, T'_M, as follows. Take an arbitrary basic class—say, C_1. We get T'_M by adding a constant c to the language of T_M, and the axiom $\vdash_{T'_M} C_1 c$, which we call the axiom for c. We will deal with T'_M in what follows rather than with T_M. This is a legitimate move because,

for any sentence ϕ in the language of T_M (i.e., any sentence in T'_M not involving the new constant c), $\vdash_{T_M} \phi$ iff $\vdash_{T'_M} \phi$. In other words, T'_M is a *conservative extension* of T_M.

LEMMA 1. *T'_M is a conservative extension of T_M.*

Proof. By the non-triviality axiom for C_1, we have $\vdash_{T_M} \exists x C_1 x$. The lemma therefore follows by a theorem on functional extensions (Shoenfield 1967, pp. 55–56). □

THEOREM 1 **(Always-Preciseness)**. *Any consistent conceptual modeling script in ERM-R is precise: for any script M, the theory T_M is complete in FO^2.*

Proof. By Shoenfield (1967, p. 83, Lemmas 2 and 3), it suffices to show that every variable-free formula of T'_M is decidable in T'_M and that every simply existential formula (formula of the form $\exists x A$, with A open) is equivalent in T'_M to an open formula. FO^2 can be axiomatized to recover classical consequence (Henkin 1967)—i.e., if Γ is a set of formulas of FO^2 and ϕ is a formula of FO^2, and ϕ is derivable from Γ in FOL, then ϕ is derivable from Γ in FO^2. Therefore, in showing that each simply existential formula of FO^2 is equivalent in T'_M to an open formula of FO^2, we will not worry about intermediate formulas in FOL but not in FO^2 (e.g., through changes of bound variables).

The only variable-free formulas in $\mathcal{L}'$, the language of T'_M, are boolean combinations of formulas of the form $P_k c$, $C_k c$, and $R_k cc$. Therefore, it is enough to show that each such atomic formula is decidable in T'_M. First, each $C_k c$ is true if C_k is a superclass of C_1 by the axiom for c and the composition and class equivalence axioms. If C_k is not a superclass of C_1, then $C_k c$ is false by the composition and class exclusion axioms. Once we have the truth values of $C_k c$, we can determine whether $P_k c$ is true by the characterization and composition axioms.

Finally, take $R_k cc$. We have either $\vdash_{T'_M} \phi$ or $\vdash_{T'_M} \neg\phi$ according to whether the model-description contains a sentence "$C_j R_k$," where C_j is any superclass of C_1, by the axiom for c, the composition axioms, and the characterization axioms.

Now we show that every simply existential formula $\exists x A$ is equivalent in T'_M to an open formula. First, we assume that A is in disjunctive normal form (i.e., A is a disjunction of conjunctions of literals). Because the existential quantifier distributes over disjunction, we can assume without loss of generality that A is a conjunction of literals. We can further assume that every conjunct of A contains an occurrence of x, because $\exists x(B \wedge C)$ is equivalent to $\exists x B \wedge C$ if x does not occur in C. Now, we will show that $\exists x A$ is equivalent in

T'_M to a boolean combination of formulas in the set $\chi = \{C_i y | 1 \leq i \leq l\}$. (Once we know what classes y can belong to, we know everything there is to know about what other properties it can have.)

A is equivalent to a conjunction of the formulas A_x and A_y, where, A_x is the conjunction of those literals in A in which no variable other than x occurs, and A_y is the conjunction of those literals in A in which x and y both occur. We will build a boolean combination of formulas in χ by using A_x to restrict our attention to a subset of the original conceptual model, and then using that subset plus A_y to state a constraint on y.

To begin, focus on A_x. We will use this to determine the classes to which x might belong. We do this by showing that A_x (i.e., any conjunction of literals in each of which x occurs, but no other variable occurs) is equivalent to a disjunction $C_{i_1}x \vee \ldots \vee C_{i_j}x$. We proceed by induction on the number of literals in A_x.

First, suppose A_x is a literal. Then there are five possibilities: A_x is $C_k x$, $P_k x$, $R_k xx$, $R_k xc$, $R_k cx$, or the negation of one of these formulas. The first case is trivial. In the remaining cases, the exhaustivity, characterization, composition, class exclusion, and class equivalence axioms, plus the axiom for c, give us our result.

Now, for induction, assume that the claim holds for any formula of length k or less ($k \geq 1$), and that A_x contains $k+1$ literals. Then A_x is equivalent to $B \wedge D$, where B is a literal, and D is of length k. Then D is equivalent to a disjunction $C_{i_1}x \vee \ldots \vee C_{i_j}x$ by the induction hypothesis, and B to a disjunction $C_{g_1} \vee \ldots \vee C_{g_h}$. So A_x is equivalent to $(C_{g_1} \vee \ldots \vee C_{g_h}) \wedge (C_{i_1}x \vee \ldots \vee C_{i_j}x)$, which is equivalent to $(C_{g_1} \wedge C_{i_1}x) \vee \ldots \vee (C_{g_h} \wedge C_{i_1}) \vee \ldots \vee (C_{g_h} \wedge C_{i_j}x)$. But any formula of the form $C_g x \wedge C_i x$ will be equivalent either to $\perp$ by the composition and class exclusion axioms, or to $C_g x$ by the composition and class equivalence axioms.

So we can conclude that $\exists x A$ is equivalent in T'_M to $\exists x((C_{g_1}x \vee \ldots \vee C_{g_h}x) \wedge A_y)$. This in turn is equivalent to $\exists x(C_{g_1}x \wedge A_y) \vee \ldots \vee \exists x(C_{g_h}x \wedge A_y)$, so it is enough to show that $\exists x A$ is equivalent to a boolean combination of formulas in χ when A_x is an atomic formula of the form $C_i x$.

Now consider A_y. This is a conjunction of literals, each of which contains an occurrence of x and an occurrence of y. Therefore, each literal must be either $R_k xy$, $R_k yx$, or their negations. Note, first, that by the relation exclusion axioms, $R_k xy$ entails $\neg R_g xy$ and $\neg R_g yx$ for $g \neq k$; furthermore, either $R_k xy$ entails $R_k yx$ or it entails $\neg R_k yx$. Therefore, we may assume that A_y is either a single unnegated atomic formula $R_k xy$ or $R_k yx$, or it is a conjunction of negated atomic formulas. In the latter case, we can regard A_y as a

conjunction $B_y \wedge D_y$, where B_y is a conjunction of formulas of the form $\neg R_k xy$, and D_y is a conjunction of formulas of the form $\neg R_k yx$.

By the characterization axiom for C_i (or by the composition axiom, and the characterization axioms for the component subclasses of C_i), we can show that $\exists x A$ is equivalent to $(C_{i_1} y \vee \ldots \vee C_{i_k} y)$, where the various C_{i_g} are specified as follows. We can use A_y to produce a list of classes to which y might belong. If A_y is of the form $R_k xy$, then the list will consist of those classes C_j such that the model-description contains "$C_a \rightarrow_\exists R_k C_b$" or "$C_a \rightarrow_\forall R_k C_b$," where C_a is a superclass of C_i (the class to which x belongs) and C_b is a superclass of C_j. In this case, $\Phi_i x$ and $\Psi_i x$ (or $\Phi_j x$ and $\Psi_j x$ for the basic subclasses of C_i) were constructed to entail that $\exists y(C_j y \wedge R_g xy)$; and $\Phi_j x$ and $\Psi_j x$ for all other classes C_j were constructed to entail the negation of this formula. The list is similar if A_y is of the form $R_k yx$. Suppose instead that A_y is of the form $B_y \wedge D_y$, as described at the end of the previous paragraph. Then we produce a list for B_y and a list for D_y, and take the intersection of the two lists. The list for B_y will consist of those classes C_j such that the model-description contains no sentence "$C_a \rightarrow_\forall R_{k_g} C_b$," where C_a is a superclass of C_i and C_b is a superclass of C_j, for $1 \leq g \leq h$. In this case, $\Phi_i x$ and $\Psi_i x$ (or $\Phi_j x$ and $\Psi_j x$ for the basic subclasses of C_i) were constructed to entail that $\exists y(C_j y \wedge \neg R_g xy)$; and $\Phi_j x$ and $\Psi_j x$ for all other classes C_j were constructed to entail the negation of this formula. The list for D_y is built similarly to the list for B_y. If either of these lists is empty, or if they do not overlap, then $\exists x A$ is equivalent in T'_M to $\perp$. $\square$

The following fact is not difficult to verify. Say a script M is inconsistent iff T_M is inconsistent. For convenience, write $Cl(A)$ for the $\subseteq$-closure of a class A (in a given script M): $Cl(A)$ is the smallest set of classes including A and every class in M which can be reached by a finite number of $\subseteq$-steps from A. For example, if M includes the sentences "$A \subseteq B$" and "$B \subseteq C$," then B and C are both members of $Cl(A)$.

PROPOSITION 1. *A script M is inconsistent iff at least one of the following conditions holds:*

1. *For some classes A and B, there are classes $C \in Cl(A)$ and $D \in Cl(B)$ such that M contains both "$A \rightarrow_\exists R_k B$" and "$C \rightarrow_\forall R_k D$" for some k.*

2. *For some classes A and B, there are classes $C_1, C_2 \in Cl(A)$ and $D_1, D_2 \in Cl(B)$ such that M contains both "$C_1 \rightarrow_\forall R_i D_1$" and "$C_2 \rightarrow_\forall R_j D_2$" for some $i \neq j$.*

Condition 1 entails that M is inconsistent because our semantics for "$A \rightarrow_\exists R_k B$" entails that each member of A bears R_k to some *but not all* members of B. Condition 2 entails that M is inconsistent because of our requirement that the relations R_k be mutually exclusive.