# Contents

To appear in: *Proceedings of the AAAI-92 Workshop on Enterprise Integration*, held San Jose, California, July 13, 1992.

# Applications of Logicist Knowledge Representation to Enterprise Modelling

**Benjamin N. Grosof** **Leora Morgenstern**
IBM T. J. Watson Research Center
P.O. Box 704, Yorktown Heights, NY 10598
(914) 784-7100 ;
Internet: grosof@watson.ibm.com leora@watson.ibm.com

April 17, 1992

## Abstract

The goal of the enterprise modelling efforts at IBM is to develop methods to model an organizational unit's knowledge and activities. Ultimately, enterprise modelling should support an ongoing, incremental automation process. The problem we address here is the applicability of declarative AI knowledge representation (KR) to enterprise modelling. Does KR have a useful role to play? What is it? What are its limitations?

In an initial case study in the area of purchasing, we find that much of currently non-automated guidelines information can be represented successfully using standard logicist methods. We show how to create very-high-level specifications with well-understood semantics. These specifications are useful as descriptive information in non-executable form; they help identify ambiguities, inconsistencies, and omissions in less formal guidelines information. We also show that these very-high-level specifications are partially executable using standard logicist inference methods such as rule-based programming and/or logic programming. This is useful for development of prototype software and simulations.

Our logicist approach provides a rich language and set of methods for describing and propagating constraints, especially when compared to other less formal approaches such as the Entity-Relationship model. However, we discover that most guidelines information relies heavily on common sense, and raises difficult knowledge representation challenges in temporal, default, decision-theoretic, and multiple-level reasoning that expose the limits of state-of-the-art logicist methods.

# 1 Introduction

The overall goal of the enterprise modelling efforts at IBM (involving collaboration between several different IBM units) is to develop methods to model an organizational unit's knowledge and activities. Ultimately, enterprise modelling will support an ongoing, incremental automation process. Once developed, a formal enterprise model can be used in two ways. First, in non-executable form, it enables rigorous reasoning about the enterprise, and facilitates communication among members and re-designers of the enterprise. Second, an executable specification can be used for experimentation and simulation.

Declarative / logicist AI knowledge representation offers the prospect of a *lingua franca* with strong semantics. Formal logic remains unsurpassed as a language for rigorous reasoning. It provides a very-high-level specification language that can unify such diverse areas as data typing, database semantics, rule-based programming languages, object-oriented inheritance mechanisms, KIF (DARPA Knowledge Interchange Format), conceptual graphs, and semantic networks. The specification is "very-high-level" in the sense of high-level programming languages: it is abstracted from considerations of computational implementation, such as procedural ordering. Yet the close connection between formal logic and logic programming languages means that it is relatively straightforward to make many logicist specifications executable. On the other hand, logicist representation has some drawbacks: it is often hard, or impossible, to implement inference based on expressively richer logicist representations, and the level of high-level rigor required may be too burdensome to be practical.

The problem we address here is the applicability of declarative / logicist knowledge representation (KR) methods to enterprise modelling. Do they have a useful role to play? What is it? What are their limitations?

# 2 Subject Area: Purchasing

We chose a purchasing department as the subject for an initial case study for the following reasons. First, it is a process that is relatively well-understood and easy to articulate. (By contrast, say, the hiring or sales functions involve, in their most crucial aspects, highly subjective, idiosyncratic, and implicit processes of judgments and personal bonding.) Second, it is relatively small in scope (compared to an enterprise in its entirety). Third, purchasing is a sub-enterprise which occurs in virtually all business enterprises. Fourth, we note that many of the concepts that arise within the context of purchasing, such as contracts, leasing, confidentiality, authorization, and co-ordination of discussion and decision-making, occur in many other departments as well. Thus, much of the work that we do for this project should carry over into other endeavors as well. Fifth, this is a serious, non-toy problem, involving many subtle, complex issues.

We chose to use enterprise information that is currently not automated. This choice separates the issue of the descriptive use of logicist methods from the more complex and less direct issue of their use for implemented automation. The most formal part of the enterprise information consists of a guidelines manual in the Purchasing department of a part of a major Fortune 500 company; much implicit and less formal information resides in the employees and managers of the department.

# 3 Knowledge Engineering

The first step of any effort in enterprise modelling is knowledge engineering. To formalize the purchasing department, we needed to understand what it is that the employees actually did, how they learned the ropes, what the written and unwritten rules were, and what the chain of command was. We began by speaking with several managers of the department, and then studied the department manual.

Manuals tend to be dry, soporific reading, but from the knowledge engineering point of view, they are often fascinating. There is so much that is *not* in the manual, that is assumed to be common sense knowledge that is possessed by all who will use the manual. For example, much of the work of employees in the Purchasing department involves the telephone (getting bids, contacting suppliers, etc.), but the telephone is never mentioned in the manual. In the next two sections, we will give more details of implicit and commonsense knowledge involved in formalizing the manual.

We began working with first order logic. However, it soon became apparent that first-order logic was not sufficiently expressive or powerful for our needs. To fully model even a small piece of the purchasing department, it becomes necessary to model plausible reasoning, vagueness, reasoning about plans, multiple agents, and communication. We discuss these in more detail in section 5.

## 4 Example Formalization

The kind of information we modelled in our case study is illustrated by the following sample page from the Purchasing guidelines manual for a Fortune 500 company unit, here called LuckyUnit. The overall manual has approximately 25 sections, on subjects such as authorization levels, bid analysis, import/export procedures, insurance, and requests for quotation. [1] This sample page deals with Supplier Qualification.

1. `All suppliers requested to supply quotes, products, or services must be qualified by LuckyUnit.`

2. `Normal financial qualification for suppliers for anticipated annual purchases by LuckyUnit of $100,000 to $30 Million shall be a satisfactory London-Rater report OR other nationally accepted rating (e.g., Paris-Rater).`

3. `Technical qualification will be determined jointly by the Buyer and division area after discussion and review of supplier proposals and capabilities.`

4. `Proposed sources for anticipated annual purchases of $30 Million or more must be reviewed with Lucky Corporate Finance and the appropriate technical area.`

5. `It is the Buyer's responsibility to ensure that all suppliers asked to quote on or supply products or services are qualified. This may require participation of other functions such as Project areas, Engineering, Finance, etc..`

Next, we give a formalization of this page, in first-order logic. We assume familiarity with basic logic; a good introduction can be found in (Mates, 1972).

**Ontology: Actions and Time, Agents and Events:**
First-order logic consists of three parts: a *language* determined by rules which

[1] Details of these guidelines are slightly *altered* (i.e., fictionalized) in this paper to protect confidentiality. These alterations do not affect this paper in any substantive way.

specify how sentences may be constructed; a *semantics*, telling us how to determine the truth values of sentences in the language, and *inference rules* which tell us how we can infer new sentences from old. Declarative knowledge representation, however, consists of more than just the random use of first-order logic. Rather, it is a *method* of using first-order logic. It includes standard theories and styles of axiomatization. A key concept in AI knowledge representation is the use of ontology: the basic predicates, functions, and associated definitions and axioms.

In our case study, it was immediately apparent that actions, events, and agents are central to the process of supplier qualification: they must be part of the ontology. Accordingly, it was necessary to incorporate a theory of action and time into our first order language. We chose McDermott's (1982) logic, (as extended by (Morgenstern, 1988). Time is isomorphic to the real numbers; the central concept is that of an interval of time. Actions are considered to be sets of intervals; intuitively, the set of intervals in which the action occurs. An event is an action restricted to a particular agent; thus, we have the action of riding a bicycle *ride(bicycle)* but the event of Debbie riding a bicycle, *Do(Debbie, ride(bicycle))*.

There are several event types of particular interest here, including requests (cf. rule 1) and decisions, discussions, and reviews (cf. rule 3). Also of interest are events such as verifying, establishing, and evaluating.

Events have a number of important properties. An event such as a discussion or a review can have one or more *subject*s. It also has an *outcome*. A useful function on events is *a-party*, which gives the set of agents involved in the particular event. *from* and *to* are useful functions for such obviously directed events as requests. We represent these functions with the functional binary predicates *a_party*, *from*, and *to*.

Commonly occurring agents include: LuckyUnit, Buyer, Finance, Engineering (these last two are departments), and the project area and technical areas of a potential supply situation.

The term "qualification" is used quite frequently in the guidelines text, though not always uniformly. In fact, a careful reading of this part of the purchasing manual reveals several different sorts of qualification. First and foremost, there is the overall qualification of a company *co* in a potential supply situation *pss*: we represent this with the binary predicate *qual*. But this overall concept of qualification rests upon more basic concepts of qualification. One checks for financial qualification, and for technical qualification as well: we represent these as *fin_qual(co,pss)* and *tech_qual(c,pss)*. We need to include a preliminary axiom relating these: (All variables in the axioms below are assumed to be universally quantified unless otherwise indicated.)

**Axiom 0.1**

$$qual(co, pss) \supset (fin\_qual(co, pss) \wedge tech\_qual(co, pss)$$

Note that this axiom, which gives necessary conditions for qualification has a definitional flavor; many of our axioms will have this characteristic.
We then have such miscellaneous predicates, functions, and concepts as $anticipated\_annual\_purchases(LuckyUnit, co, year)$, $satisfactory(rating)$, dollar amounts, and responsibility.

Using these concepts, we began to work our way through this section of the manual.

**Formalizing the Rules in First-Order Logic:**

The first rule tells us that only suppliers qualified by LuckyUnit can be requested to supply quotes, products, or services to LuckyUnit. We immediately see that some information important to the formalization is left implicit in the text. Implicit in the concept of qualification is that it is relative to the context of a particular potential supply situation. We use the binary predicate $belongs(e, pss)$ to indicate that an event $e$ belongs to a potential supply situation $pss$.

We thus have the rule:

**Axiom 1.1:**

$$\begin{aligned} & request(e) \wedge belongs(e, pss) \wedge \\ & \quad from(e, LuckyUnit) \wedge to(e, co) \wedge \\ & \quad subject(e, s) \wedge (quote(s) \vee product(s) \vee service(s)) \\ & \qquad\qquad \supset \quad qual(co, pss) \end{aligned}$$

Rule 2 raises several interesting issues. First, the word "normal" with which rule 2 begins indicates that this rule is best treated as a default. Since at this point we are working within standard monotonic logic, we choose instead to regard our formalization as an approximation. Second, it should be noted that $anticipated\_annual\_purchases$ can hardly be considered a primitive predicate. To fully axiomatize this predicate, we would need to formalize the concept of anticipation, which in turn rests on concepts such as prediction, approximation, and expectations. Third, we note the vagueness inherent in such predicates such as *satisfactory* and *nationally_accepted*. Modulo these caveats, our formalization of rule 2 looks as follows:

**Axiom 2.1:**

$$(anticipated\_annual\_purchases(LuckyUnit, co, x) \wedge (100,000 \leq x \leq 30Million)) \supset [(rating(co, rating\_service) = value) \wedge satisfactory(value) \wedge nationally\_accepted(rating\_service) \equiv fin\_qual(c, pss)]$$

We also have:

**Axiom 2.2:**

$$nationally\_accepted(London\_rater) \wedge nationally\_accepted(Paris\_rater)$$

Rule 3 illustrates the need for the formalizers to use a hefty dose of common sense reasoning. It is clear that technical qualification of a company can come only *after* the Buyer and project area discuss and review supplier proposals and capabilities. Left implicit in the rule, however, is that the discussion and review must end in a positive outcome. Thus, we formalize rule 3 as:

**Axiom 3.1:**

$$\begin{aligned} & tech\_qual(co, pss) \equiv \\ & \exists e.\ outcome(e, yes) \wedge decision(e) \wedge consensus(e) \wedge belongs(e, pss) \wedge \\ & subject(e, \text{``}tech\_qual(@co@, @pss@)\text{''}) \\ & a\_party(e, Buyer(pss)) \wedge a\_party(e, technical\_area(pss)) \wedge \\ & after(e, e1) \wedge after(e, e2) \wedge \\ & discussion(e1) \wedge review(e2) \wedge \\ & subject(e1, \text{``}proposals(@co@, @pss@)\text{''}) \wedge \\ & subject(e1, \text{``}capabilities(@co@, @pss@)\text{''}) \wedge \\ & subject(e2, \text{``}proposals(@co@, @pss@)\text{''}) \wedge \\ & subject(e2, \text{``}capabilities(@co@, @pss@)\text{''}) \wedge \\ & a\_party(e1, Buyer(pss)) \wedge \\ & a\_party(e1, technical\_area(pss)) \wedge \\ & a\_party(e2, Buyer(pss)) \wedge \\ & a\_party(e2, technical\_area(pss)) \end{aligned}$$

[2]

*after* can be easily defined in a standard interval-based temporal logic; see, e.g. (Davis, 1990). Note here that we used quotation to allow one predicate, *subject*, to refer to a sentence (e.g., $tech\_qual(co.pss)$) or term (e.g., $proposals(co, pss)$) in the same language. We discuss the quotation construct further in Section 5.

[2] see (Davis, 1990), Chapter 8 for a complete discussion of the @ quasi-quote operator, used for quantifying into quoted contexts.

Rule 4 is formalized in a similar manner. We have:

**Axiom 4.1:**

$$(anticipated\_annual\_purchases(LuckyUnit, co, x) \wedge (x > 30Million)) \supset$$
$$\exists e.\ belongs(e, pss) \wedge review(e) \wedge$$
$$a\_party(e, Finance) \wedge a\_party(e, technical\_area(pss))) \wedge$$
$$subject(e, \text{``}qual(@co@, @pss@)\text{''}))$$

Since both the Finance department and the relevant technical area are involved in the review, the subject of the review seems to be general qualification.

Note that the natural language text for this rule merely specified that there be a review for any proposed source of purchasing. There is no implicit assumption here that the review must end in a positive outcome. Indeed, it will sometimes be the case that reviews end negatively. (In that case, presumably, the proposed source would not become an actual source. We did not formalize statements of this level of conjecture.)

Rule 5 is unique in this section of the manual in that it seems primarily concerned with control-level information. Certainly, a first step would be to define a predicate *Responsible* and to write down:

**Axiom 5.1:**

$$Responsible(Buyer(pss), \text{``}qual(@co, @pss)\text{''})$$

Reducing the predicate *Responsible* to more primitive predicates is a complex task. Rule 5 can be read in at least two ways: the rule might be saying that the buyer is responsible - i.e. if anything goes wrong, the buyer will suffer the consequences in some way - or, the rule might just be a comment on the preceding rules: i.e., the buyer is the person who will have to *enforce* rules 1 through 4. This rule would then be interpreted as giving information about the control structure of the organization. While such information is useful in the modelling of an enterprise, it is beyond the scope of this study.

# 5 Representational Challenges and Limitations of Logicism

In the course of our case study, we discovered that even simple guidelines information raises difficult knowledge representation challenges in temporal, default, decision-theoretic, and meta-level reasoning. These difficulties exposed the limits of current logicist methods. In the last section, we mentioned some of these difficulties in passing. In this section, we give some more examples to give the flavor of these problems.

**Common Sense is Critical-Path:** Some in (and out of) AI think that logicists are a bit prissy or overly fussy in worrying about the semantics of common sense reasoning. But common sense reasoning turns out to be essential for capturing in complete and precise form the guidelines information we studied. We find that the difficulties posed by common sense knowledge and reasoning for AI knowledge representation turn out to be "critical path" in this application.

**Temporal:** Consider the following sample guidelines sentence information: `All bids will be analyzed at the same time.` Clearly, this is not meant literally. Instead, it means something like: there exists a bounded time period, in which the analyses for each bid are sequentially interleaved, and no decision is taken on any until the end of the period, which only comes after each analysis is complete. This is rather complex to represent; some temporal formalisms cannot do it at all; others may require resorting to second-order logic or set theory.

In addition, the guidelines information is pervaded by implicit persistence and/or frame assumptions; it is assumed that most features about the world remain the same when actions are performed (McDermott, 1982). This is a default assumption; thus, temporal reasoning is injected with a non-monotonic aspect.

**Default:** Many concepts employed in the guidelines information, both explicit and implicit, involve logical non-monotonicity and a need for default reasoning. For example, the concept of a contract being in force is a default presumption. It may be overridden by a violation or emergency. "Normal" procedures are really defaults; there may be exceptions, such as direct intervention by higher management. And, as discussed in the previous paragraph, the pervasive concept of temporal persistence requires non-monotonicity for proper representation. In addition, rules are usually modifiable; the guidelines may be augmented or revised. Again, this is best represented within a default logic (Reiter, McCarthy, McDermott and Doyle, 1980).

**Decision-Theoretic:** The guidelines information often explicitly uses expressions such as "sufficient justification" or "give consideration". Descriptions of planning processes implicitly rely on criteria of satisficing rather than perfect optimality.

To adequately represent these descriptional fragments we need to include concepts such as partially-constrained weights of probabilistic evidence, utility, and cost-benefit analysis. In addition, there is a need to represent the vagueness and open-ended ambiguity inherent in the ideas of "sufficient" and of satisficing. There is likewise a need to formalize such decision processes.

**Multiple Levels of Representation:** A key concept in logic is that of denotation. Denotation is a formal referring relationship. Quotation is a common formal syntactic device for representing such denotation. E.g., $tech\_qual(Acme, pss103)$ stands for a truth value, either true or false, but its quoted version, "$tech\_qual(Acme, pss103)''$ stands for the (unquoted) sentence itself.

Denotation is important for representing multiple levels of abstraction. For example, we find that it is useful sometimes to treat the description of the result of a decision process, e.g., in rule 3, that a supplier is technically qualified, as on a different semantic level than the description of the corresponding decision process. There, it was useful to represent the decision procedure as an object that has a subject attribute, whose value is the description of the result, not the result itself.

Multiple levels of representation and quotation are important for representing communication between agents, as well.

The limited use of quotation present in this paper, is fairly simple to implement and poses no fundamental logical difficulties. Unlimited use of quotation, however, which is needed for greater expressivity can result in paradox. There are techniques for dealing with such problems, but they are not trivial (cf. (Morgenstern, 1988)).

**Current Limitations of Logicist KR:** Current logicist methods for dealing with these difficult representational issues (temporal, default, decision-theoretic, multiple-level) are still immature and unstandardized. Solutions often exist at only the theoretical level, and are not practical to implement. Moreover, logicist methods for *integrating* solutions to different knowledge representation problems do not yet exist. For example, there are techniques for doing non-monotonic reasoning, and techniques for doing multiple-agent reasoning. However, research in multiple-agent non-monotonic reasoning is still in its infancy.

## 6 Uses and Executability

Our logicist method yields a very-high-level specification (cf. Section 1). In its non-executable form, this specification is useful for supporting communication, revision, and rationalization of enterprise procedures and activities. Its rigor helps to identify ambiguities, inconsistencies, and omissions in current natural language guidelines information. Its conciseness and semantics facilitate readability, verifiability, and knowledge acquisition by humans. Its ontology (situations, events, agents) provides a basis for the ontology of lower-level specifications closer to implementation.

Much of our specification is executable almost directly, via logic programming and other logic-based AI inference techniques (such as rule-based expert system technology). For example, rule 1 is easily massaged into Horn-clause form, as three rules, one per disjunct (quote, product, or service). These rules are executable either in backward or forward directions of inference.

In addition, one can use the executable form to develop prototype software to automate enterprise activities, or to develop simulations to understand and support revision of existing activities. Note that for simulation purposes, the fact that our methods may achieve only partial coverage is acceptable, since one can interface with human users (or other programs) for the uncovered remainder.

## 7 Conclusions and Future Work

We have demonstrated that first-order logic can be used to formalize a small but significant chunk of an enterprise.

We plan to extend this work in several directions. First, we plan to do more exploratory case studies, expanding both the depth and breadth of this research. Our agenda includes looking at the purchasing department in more depth, as well as examining different enterprises.

Second, we plan to go beyond first order logic. In particular, we plan to use a non-monotonic formalism such as Circumscription (McCarthy, 1980) in our formalization. Moreover, we intend to define some standardized logical constructs ("syntactic sugar") to ease the syntactic awkwardness of raw predicate calculus.

Finally, we believe that the the formal representation of enterprises will be a useful tool for the development of applications programs. In particular, formal specifications can be used as a basis for a program that generates plans and interactively steps a user through them (e.g., through the supplier qualification process). Much work needs to be done in creating the bridge between the formal specification and the applications program.

# References


Davis, Ernest. (1990) *Representations of Knowledge for Commonsense Reasoning,* Morgan Kaufmann, San Mateo, 1990

Mates, Benson. (1972) *Elementary Logic, 2nd ed.* Oxford University Press, Oxford, 1972

McCarthy, John. (1980) "Circumscription - A Form of Non-monotonic Reasoning," *Artificial Intelligence 13,* 1980

McDermott, Drew and Jon Doyle. (1980) "Non-monotonic Logic I," *Artificial Intelligence 13*, 1980

McDermott, Drew. (1982) A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

Morgenstern, Leora. (1988) *Foundations of a Logic of Knowledge, Action, and Communication.* PhD thesis, Computer Science Dept., New York University, 1988.

Reiter, Raymond. (1980) "A Logic for Default Reasoning," *Artificial Intelligence 13*, 1980