Representing Roles and Purpose

James Fan
Ken Barker
Bruce Porter
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712 USA
{jfan, kbarker, porter}@cs.utexas.edu

Peter Clark
Knowledge Systems
Boeing Math and Computing Technologies
m/s 7L66, PO Box 3707, Seattle, WA 68124 USA
peter.e.clark@boeing.com

Abstract
Ontology designers often distinguish *Entities* (things that are) from *Events* (things that happen). It is not obvious how this division admits *Roles* (things that are, but only in the context of things that happen). For example, *Person* might be considered an Entity, while *Employee* is a Role. A Person remains a Person independent of the Events in which he participates. Someone is an Employee only by virtue of participating in an Employment Event. The problem of how to represent Roles is not new, but there is little consensus on a solution. In this paper, we present an ontology that finds a place for Roles as well as a representation that allows Roles to be related to Entities and Events to express the teleological notion of purpose.

Keywords
roles; ontologies; teleology

Background
One of the challenge problems in DARPA's Rapid Knowledge Formation project requires subject matter experts (SME's) with little training in knowledge engineering to build a knowledge base of information from a college-level textbook on cell biology. The knowledge base will be evaluated on its ability to answer a large set of questions drawn from standard test banks, such as the GRE subject exam (a graduate school admissions test) and questions from the end of book chapters. Our goal is to develop ways to help SME's succeed.

One of our chief concerns for this challenge problem is developing good ways to represent the wide variety of types of knowledge expressed in textbooks. Many knowledge engineering projects can focus on just a few types of knowledge – for example, building a knowledge base about aircraft might focus exclusively on structure and partonomy – because the questions they are intended to answer are relatively limited. However, textbook knowledge and the questions we expect to be presented are quite varied.

K-CAP'01, October 22-23, 2001, Victoria, British Columbia, Canada.

There are many types of knowledge conveyed in textbooks, of course, and this paper focuses on just one – how to represent the roles and purposes of entities – which has been problematic for knowledge engineering. Although ontologies typically distinguish *Entities* (things that are) from *Events* (things that happen), it is not obvious how this division admits *Roles* (things that are, but only in the context of things that happen).

The source of the problem lies in the distinction between intrinsic and extrinsic features. **Intrinsic** features, such as *shape* and *size*, describe an entity in isolation. In contrast, **extrinsic** features describe an entity relative to other entities and events. For example, *used to strike nails* is an extrinsic feature of a hammer because it relates a hammer to nails and striking. Efforts to represent concepts using only intrinsic features have largely failed [11], especially for representing artifacts [3].

Although the distinction between intrinsic and extrinsic properties is more spectral than black-and-white, it is important to distinguish the many cases that fall into the uncontroversial extremes because they differ in significant ways. For example, an entity's intrinsic features (such as *age*) may change over time, but they are always applicable to the entity. In contrast, extrinsic features (such as the *salary* of a person) may become completely inapplicable. Moreover, unlike intrinsic features, an entity's extrinsic features may be contradictory, such as the *salary* of a person with multiple jobs. For these reasons, most psychological research on concept representation distinguishes between an entity's extrinsic and intrinsic features [11].

From these distinctions (and others we discuss later) we draw three conclusions. First, the distinction between intrinsic and extrinsic features is important; a knowledge-based system that ignores their differences might draw incorrect inferences. Second, the roles and purposes of an entity are necessarily extrinsic features, *i.e.* they relate an entity to other entities and events. Finally, roles should be reified in any knowledge representation scheme. The representation of a

role consists of those extensional features of an entity that are due to its participation in some event.

The Difference between Roles and Entities

There has been considerable research on roles in data and knowledge modeling, as we summarize below. The research offers two key insights. First, entities and roles are not related taxonomically, at least not in any simple way; "Neither the roles of the real world nor the entities of the real world are a subset of the other" [2]. Guarino offers two criteria for distinguishing roles from entities [6]: (1) a role is "founded" and (2) a role lacks "semantic rigidity". Something is founded if it is defined in terms of relationships to other things. Something is semantically rigid if its existence is tied to its class; that is, if in ceasing to be of kind X, it ceases to be. For example, the concept *food* is a role because it meets these criteria, as follows:

- *Food* is Founded: The properties of food, such as *eaten-by* and *nutritional-value*, are extrinsic properties of the entity filling the role of food – they relate that entity to others participating in the *eating* event, such as the *eater*, and they are applicable only in that context.
- *Food* lacks Semantic Rigidity: An entity that might fill the role of food retains its identity (i.e. its primary class membership) outside the context of the role. For example, a grasshopper is food when eaten by a bird, but when it is no longer considered food, it is still a grasshopper.

In contrast, these criteria tell us that *person* is an entity, and not a role, for the following reasons:

- *Person* is not Founded: The properties of a Person, such as *age* and *sex*, are intrinsic features. They are defined independently of other entities and events.
- *Person* has Semantic Rigidity: when a Person ceases to be a Person, she ceases to be.

Roles in Use

There would be little value in devising a complicated representation for roles if they do not occur frequently. To gauge how common roles are, we ran a simple experiment using English word lists.

We first extracted from a large online wordlist [1] nouns that end in "-ee", "-er", "-or" or "-ist". These endings, such as *employee*, *driver*, *actor* and *pianist*, are good cues for roles . We pruned this list to only those whose stems are also stems of base verb forms. The result was a list of more than $5,000$ candidate role names. To determine how many of these might actually represent role concepts, we sampled 109 at random. Based on the tests of foundedness and lack of semantic rigidity, 101 of the sampled nouns represented role concepts. Given that there are $74,577$ unique noun entries in the Collins wordlist, this experiment suggests that at least 6% of nouns may represent role concepts (at 95% confidence). The suffix filter would miss many potential roles, making this number an underestimate of roles in use.

As a second experiment, we checked a list of the most frequently used nouns in the the British National Corpus [7]. 200 of the roughly $3,000$ most frequent nouns represented role concepts, meaning that role concepts also account for 6% of the most common nouns. (Previous work [15] has established that there is considerable overlap among the more frequent words in different corpora).

A Knowledge Representation for Roles

Roles are easy to identify yet they are difficult to represent. They are not merely reified names for the participants in events. Rather, roles have their own characteristics which require that they be treated differently than entities in a knowledge representation scheme. Steimann [14] identified fifteen characteristics of roles, which we've distilled into these four:

1. Roles are created and destroyed dynamically. Because a role represents the extrinsic features of an entity due to its participation in an event, the role is created when the participation begins. If the entity stops participating, the role may cease to exist and all its properties may no longer hold.

2. A role can be transferred between entities. For example, the role of *manager* can be transferred from one person to another. Note that many of the role's features are transferred without change, while others must be re-computed in light of the new entity playing the role. For example, if a person earns a 20% bonus for being manager, then the *salary* feature must be recomputed should that role be transferred.

3. An entity may play different roles simultaneously, for example a *person* may be both an *employee* and an *employer*.

4. Entities of unrelated types can play the same role. For example, both a cracker and a grasshopper can play the role of *food*.

These four characteristics impose requirements on any knowledge representation scheme for roles. The next section assesses past approaches to representing roles in light of these requirements.

Previous Approaches to Representing Roles

According to Steimann [14], previous research produced three basic approaches to representing roles. The first approach represents a role as nothing but a label assigned to a participant in an event. For example, the *employer* role labels the agent of an *employ* event. This approach is simple, but it fails to reify roles as distinct from entities (instead combining intrinsic and extrinsic properties into a single representation of an entity), which is problematic as we discussed in Section *Background*.

Assuming that these labels can be assigned and retracted dynamically (as entities play roles and later drop them), this approach meets the first requirement ("roles are dynamic")

and the second requirement ("roles can be transferred"). The approach does not meet the third requirement ("entity can play multiple roles") because roles are not reified as predicates with arguments. Rather in this approach roles are simple propositions. Consequently the extrinsic features of an entity can clash due to the entity's participation in different events. For example, if a person has two jobs, then the two employee roles she plays will give her two different salary values. If a query about her salary is posed, then it is not clear which value should be returned. Finally, this approach meets the fourth requirement ("entities of different types can play the same role") as there are no constraints on assigning labels to entities.

The second approach, used by Sowa [13] and Uschold [8] , reifies roles and distinguishes them from entities (in that roles represent extrinsic features and entities represent intrinsic ones), then combines the two types of concepts into a single hierarchy. They can be combined in either of two ways; both are problematic [14]:

1. the roles are subtypes of entities. For example, the role *employer* would be a subtype of the entity *person*, as shown in Figure 1 (a). This becomes problematic when trying to meet the fourth requirement ("entities of different types can play the same role"). To illustrate, consider extending the hierarchy to assert that an *employer* may be either a *person* or an *organization*, as shown in Figure 1 (b). This taxonomic structure says that every *employer* is **both** a *person* and an *organization* – not what we intended. In an effort to represent the disjunction of person and organization, we create a new type, *legal-entity*, which subsumes *person* and *organization*, as shown in Figure 1 (c). Because an *employer* must be a *legal-entity*, *employer* must be a sibling of *person* and *organization*. This does not capture our original assertion that an *employer* is either a *person* or *organization*.

2. the roles are supertypes of the entities that play them. For example, *employer* would subsume *person*, as shown in Figure 1 (d). This is clearly wrong because not every person is an employer. Moreover, it fails to meet the first requirement ("roles are dynamic"), unless the subtype relationship between entities and roles is dynamic. To avoid this paradox, some knowledge representation schemes take exactly that approach [5]. (See also *qua-classes* of KL-ONE [4] and the *existence subclass* of SDM [9] and MERODE [12].) These schemes have the restriction that a role exists if and only if an entity is actually playing that role. This restriction makes it difficult to use role concepts to represent an entity's purpose, as we discuss in Section *Representing Purpose using Roles*.

The third approach represents a role as an "adjunct instance" of an entity. An adjunct instance here is a distinct instance of a role class that is coupled with the instance of an entity; the

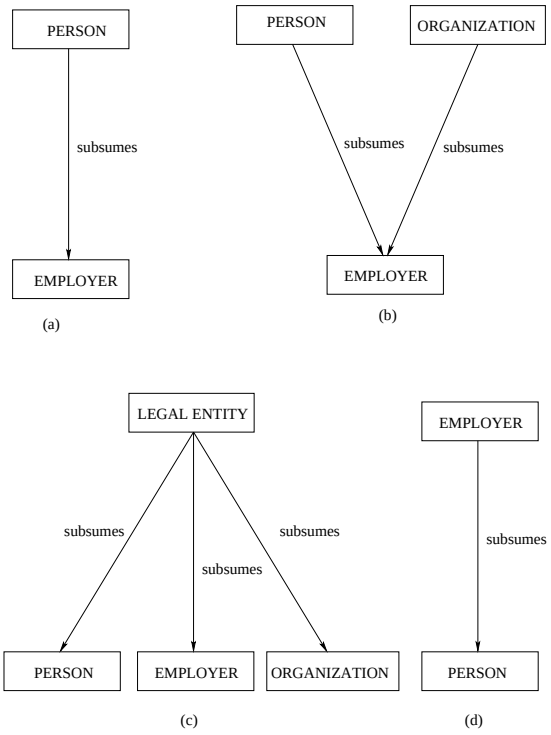

Figure 1: Taxonomy paradox. If roles and entities are combined into one hierarchy, none of the hierarchies above fully captures the intended information: an *employer* can be either a *person* or an *organization*.

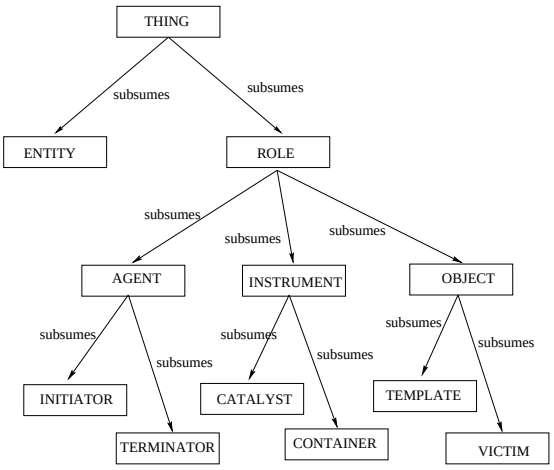

Figure 2: A partial listing of our role hierarchy. *Role* is a sibling of the top-level concept *Entity*, and it has several subtypes, such as *Agent*, *Instrument* and *Object*.

role instance does not exist independent of that entity. We adopt this basic approach, as we discuss next.

Our Approach

We built a representation of roles using the adjunct instance approach to express what an entity is designed to do (its purpose), and what an entity actually does (its role). In our representation, roles are types independent of entities. An instance of a role is played by an instance of an entity; every instance of a role exists along with an instance of an instance of an entity. The role instances are connected with the entity instances through two composition methods described in Section *Role composition*. In order to retrieve values of properties that belong to a role, we need to first retrieve the role from the entity with which it is composed, and then we can retrieve the values of properties from the role.

In keeping role concepts separate from entities, the problem arises of where in the taxonomy role concepts belong. In order to avoid the taxonomy paradox described above, we make Role a sibling of the top-level Entity concept (see Figure 2).

For our project, we are mainly concerned with general role concepts. Examples of such general roles include:

- *Agent*: the role played by an entity performing or responsible for an event. More specific Agent roles include *Initiator*, *Terminator*, *Creator*, *Interpreter*.
- *Instrument*: the role played by an entity used in some event. More specific Instrument roles include *Container*, *Catalyst* and *Connector*.
- *Object*: the role played by an entity acted upon in an event. More specific Object roles include *Template* (Object of a *Copy* event), *Idol*, *Input* and *Victim*.

Our solution is implemented in the KM language [10]. KM is a frame-based language with clear first-order logic semantics. To avoid issues of KM syntax, we will illustrate our solution with examples expressed in first-order logic.

Representing Purpose using Roles

The reification of roles (as distinct from entities) provides a convenient way to represent the teleological notion of purpose. We represent an entity's purpose as the default role(s) it plays. For example, the default role of *cereal* is *food* (i.e. to be the object eaten by people) and the default role of a *cup* is to *contain* (i.e. to be the instrument of containment). These entities are artifacts, which typically have a clear purpose, but natural entities are often ascribed a purpose, too. For example, one purpose of a human hand is to grip.

Role composition

In our approach, roles are types and instances of roles are played by instances of entities; an instance of a role requires a corresponding entity. The correspondence is established with 2 relations: *played-by* and *purpose*. When an entity is related to a role with one of these relations, we say they are composed together. (In our knowledge representation scheme [10], such compositions have inferential ramifications, which are outside the scope of this paper.)

The *played-by* composition represents that an entity is actually participating in an event. (In our knowledge representation scheme, this can be asserted to hold in a temporally bounded state.) For example, when a hammer is participating in a hammering event, the instrument role for the hammering event is *played-by* the hammer. (Equivalently, the hammer *plays* the instrument role for the hammering event.)

The *purpose* composition represents a role that the entity is intended to play, but says nothing about whether it is actually doing so. For example, the purpose of a hammer is to be the instrument of a hammering event, which is true even when the hammer is not participating in any hammering event.

Both *played-by* and *purpose* are many-to-many relations, which means that an entity can play multiple roles and a role can be played by multiple entities. Both relations are fluent, which means that an entity can dynamically acquire and relinquish roles.

It is possible and common for an entity to play a role that is the purpose of another entity. For example, the purpose of a *hammer* is to be the instrument of a *hammering* event; a

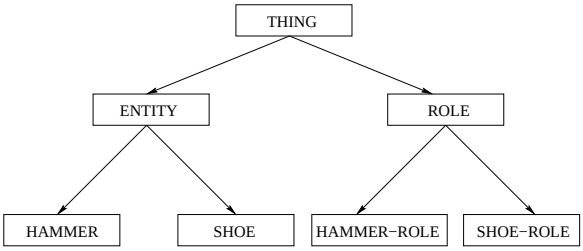

Figure 3: The duplication of the entity hierarchy in the role hierarchy caused by the promiscuous reification of the purpose of entities.

shoe might also "play the role" of a hammer – or more accurately, play the role that is the purpose of a hammer. In order to avoid this representational gymnastics, we could reify the purpose of a hammer as a *hammer-role* so that the shoe plays a hammer-role. Note that a shoe cannot "play" a hammer because hammer is an entity, not a role.

Although we *could* reify hammer-role, that leads to a potential problem. Reifying the purpose of entities promiscuously will result in duplication of the entity hierarchy in the role hierarchy (Figure 3). The duplication problem is inherit in any representation of purpose. In practice, however, it is not a serious issue because most roles need not be reified. Our criteria is to reify only those roles, such as *container*, that are likely to be played by many different kinds of entities, not just those entities whose purpose is to play the role.

Non-reified roles are specialized instances of generic roles, and they are left unnamed. Specialization is accomplished through the addition of properties or constraints on an instance of the generic roles. As an example of composition, the purpose of a hammer might be represented as follows:

$$\forall x\ isa(x, Hammer) \rightarrow$$
$$\exists y, z\ \ isa(y, Instrument) \wedge$$
$$isa(z, Hammering) \wedge purpose(x, y) \wedge in-event(y, z)$$

The Skolem variable y is an example of a non-reified role.

By using a combination of *purpose* composition and non-reified role concepts, we can avoid the problem of duplicating the entity hierarchy in the role concept hierarchy. For example, a representation of using my shoe as a hammer would be:

$$\exists p, h\ \ isa(myShoe, Shoe)$$
$$\wedge isa(h, Hammer) \wedge plays(myShoe, p) \wedge purpose(h, p)$$

The Skolem instance p is a non-reified role denoting the purpose of a hammer. It is used to express that $myShoe$ plays that role. That is, $myShoe$ plays the role which is the purpose of a hammer.

Conclusion

The distinction between *entities* (things that are) and *events* (things that happen) is clear and common in ontologies, but it's decidedly less clear how to handle *roles* (things that are, but only in the context of things that happen). Although roles are often confused with entities, and mixed together in a single hierarchy, we draw from the data modeling literature an operational distinction between them. Using this distinction we determine that roles are frequently used in English text, accounting for more that 6% of the most common nouns. We describe our representation in which roles are reified, and instances of roles are composed with the entities that participate in them. Finally, we show how this representation can be easily extended to include the teleological notion of the purpose of entities.

Acknowledgments

We wish to thank Charles Benton, Paul Navratil, Art Souther, Dan Tecuci, John Thompson, and Peter Yeh for their insightful comments and suggestions. Support for this research is provided by a contract from Stanford Research Institute as part of DARPA's Rapid Knowledge Formation project. This material is based upon work supported by the Space and Naval Warfare Systems Center - San Diego under Contract No. N66001-00-C-8018.

REFERENCES

1. *Collins English Dictionary*. William Collins Sons Co. Ltd., 1979.

2. C. Bachman and M. Daya. The role concept in data models. In *Proceedings of the 3rd International Conference on VLDB*, pages 464–476, 1977.

3. R. Barr and L. Caplan. Category representations and their implications for category structure. *Memory and Cognition*, 15:397–418, 1987.

4. R. Brachman and J. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216, 1985.

5. J. Odell C. Bock. A more complete model of relations and their implementation: Roles. *Journal of Object-Oriented Programming*, 11:51–54, 1998.

6. N. Guarino. Attributes and arbitrary relations. *Data and Knowledge Engineering*, 8, 1992.

7. A. Kilgarriff. BNC database and word frequency lists.

8. S. Moralee M. Uschold, M. King and Y. Zorgios. The enterprise ontology. *The Knowledge Engineering Review*, 13, 1998.

9. D. McLeod and M. Hammer. Database description with SDM: A semantic database model. *ACM Transactions on Database Systems*, 6(3):351–386, 1981.

10. B. Porter and P. Clark. KM - the knowledge machine: Reference manual. Technical report, University of Texas at Austin, 1998.

11. E. Smith and D. Medin. *Categories and Concepts*. Harvard University Press, Cambridge, MA, 1981.

12. M. Snoeck and G. Dedene. Specialization and role in object oriented conceptual modeling. *Data and Knowledge Engineering*, pages 171–195, 1996.

13. J. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. AddisonWesley Publishing Company, New York, 1984.

14. F. Steimann. On the representation of roles in object-oriented and conceptual modelling. *Data and Knowledge Engineering*, 35(1):83–106, 2000.

15. S. Delisle T. Copeck, K. Barker and S. Szpakowicz. More alike than not–an analysis of word frequencies in four general-purpose text corpora. In *Proceedings of the Fourth Conference of the Pacific Association for Computational Linguistics*, pages 282–287, 1999.