The Knowledge Engineering Review, Vol. 17:1, 41–60. © 2002, Cambridge University Press
DOI: 10.1017/S0269888902000371 Printed in the United Kingdom

An ontology model to facilitate knowledge-sharing in multi-agent systems*

VALENTINA TAMMA and TREVOR BENCH-CAPON

Department of Computer Science, Chadwick Building, Peach Street, Liverpool, L69 7ZF, UK

Abstract

This article presents and motivates an extended ontology knowledge model which represents explicitly semantic information about concepts. This knowledge model results from enriching the standard conceptual model with semantic information which precisely characterises the concept's properties and expected ambiguities, including which properties are prototypical of a concept and which are exceptional, the behaviour of properties over time and the degree of applicability of properties to subconcepts. This enriched conceptual model permits a precise characterisation of what is represented by class membership mechanisms and helps knowledge engineers to determine, in a straightforward manner, the meta-properties holding for a concept. Meta-properties are recognised to be the main tool for a formal ontological analysis that allows us to build ontologies with a clean and untangled taxonomic structure.

This enriched semantics can prove useful to describe what is known by agents in a multi-agent system, and might facilitate the use of reasoning mechanisms on the knowledge that instantiates an ontology. These mechanisms can be used to solve ambiguities that can arise when agents with heterogeneous ontologies have to interoperate in order to perform a task.

1 Introduction

Advances in the Internet have made it possible to access huge amounts of diverse information from different places all over the world. Combining this information in order to obtain added value on a domain requires a deeper understanding of how heterogeneous knowledge sources can be integrated. The complexity of this task is high, chiefly because of the different types of heterogeneity that may affect knowledge sources and, to a lesser extent, their size.

One knowledge-engineering paradigm that has proved useful for dealing with the integration of heterogeneous knowledge is based on a multi-agent system architecture, where human and software agents interoperate and so cooperate within common application areas. Agents in a multi-agent system are characterised by autonomy, adaptability, interoperability and dynamism. These qualities are particularly useful in that they can help to facilitate open systems which are typically dynamic, unpredictable and highly heterogeneous (Jennings, 1995), as is the Internet. In these types of application domain, agent interoperability typical of the multi-agent system approach is required because the individual components that interact with agents are not known a priori. Additionally, this paradigm provides robustness and flexibility of both the interfaces between the agents that exist within

* The Ph.D. research presented in this work was funded by BT plc. We are grateful to Ray Paton for providing the medical notions necessary to model the example. We are indebted to Floriana Grasso for the stimulating discussions and the comments made throughout this Ph.D. research and, in particular, during the preparation of this article. We would also like to thank the anonymous referees for their valuable comments on an earlier version of this paper.

the Internet and those between agents and software systems, which is essential since the interfaces cannot be anticipated at design time (Jennings, 1995).

Within a multi-agent system, agents are characterised by different *views of the world* that are explicitly defined by *ontologies*, that is, views of what the agent recognises to be the concepts describing the application domain which is associated with the agent together with their relationships and constraints (Falasconi *et al*., 1996). Interoperability between agents is achieved through the reconciliation of these views of the world by a commitment to common ontologies that permit agents to interoperate and cooperate while maintaining their autonomy.

In open systems, agents are associated with knowledge sources which are diverse in nature and have been developed for different purposes. Knowledge sources embedded in a dynamic environment can join and leave the system at any time. From the ontology perspective, dealing with open systems implies that different ontologies are often the efforts of different domain experts, where each ontology is designed and maintained independently in distributed environments. In these kinds of situation, interoperation between agents is based on the reconciliation of their heterogeneous views, which is accomplished by merging or integrating the diverse ontologies associated with the agents composing the system (Sycara *et al*., 1999). The merging and integration of diverse ontologies has to be accomplished bearing in mind that since agents are highly heterogeneous, they are likely to be incapable of fully understanding each other, so that both syntactic and semantic inconsistencies can arise (because of language or ontology heterogeneity, illustrated in Section 3).

An agent architecture is defined (Wooldridge & Jennings, 1995) as

> one that contains an explicitly represented, symbolic model of the world, and in which decisions (for example about what action to perform) are made via logical (or at least pseudo-logical) reasoning, based on pattern matching and symbolic manipulation.

Therefore ontologies in multi-agent systems require a high degree of expressive power to support the application of reasoning techniques that result in sophisticated inferences such as those used in negotiation, which is motivated by the requirement for agents to solve problems arising from their interdependence upon one another (Parsons *et al*., 1998).

This article presents and motivates a knowledge model for ontologies which enriches the traditional ontology conceptual model by providing a precise characterisation of the properties that define concepts. This characterisation includes which properties are prototypical of a concept and which are exceptional, the behaviour of the property over time and the degree of applicability of properties to subconcepts (that is, the way properties are inherited by subconcepts). This enriched knowledge model aims to provide the kind of semantic information typical of formal ontology (Cocchiarella, 1991); this information can prove useful to dealing with problems of semantic inconsistency that arise when reasoning with integrated ontologies. In order to provide an example of how such a conceptual model could be implemented we have extended the usual set of facets in the OKBC frame-based model (Chaudhri *et al*., 1998) to encompass this additional information.

The article is organised as follows: Section 2 introduces the notion of ontology while Section 3 defines the problem of sharing knowledge between heterogeneous agents. Section 4 introduces the enriched ontology model we propose, which is implemented in an OKBC-like model described in Section 5. Section 6 illustrates the expressive power of the knowledge model, while Section 7 discusses the extensions relating it to the tools of the *formal ontological analysis* described in Section 2. These tools are intended to provide guidelines for developing ontologies that are easier to integrate. An example of concept description using the knowledge model is described in Section 8, and finally, in Section 9, conclusions are drawn.

2 Ontologies

In this section we will first clarify what we mean by ontologies, because there is no universal agreement on the meaning of this term.

Ontology is the branch of philosophy which considers the nature and essence of things. Artificial intelligence takes a less abstract view and defines ontologies as an an explicit, formal specification of a shared conceptualisation (Studer *et al*., 1998). In this definition, a *conceptualisation* refers to an abstract model of some phenomenon in the world which identifies the concepts that are relevant to the phenomenon; *explicit* means that the type of concept used, and the constraints on their use, are explicitely defined; *formal* refers to the fact that an ontology should be machine-readable; and finally *shared* reflects the notion that an ontology captures consensual knowledge, knowledge that is not private to some individual but is accepted by a group (Studer *et al*., 1999).

According to Gruber (1993), an ontology comprises *concepts*, *relations*, *functions*, *instances*, and *axioms*. A *concept* represents a set or a class of entities or things within a domain. A concept C is described in terms of *properties* – the features that all instances of C have. Properties can be *intrinsic* or *extrinsic*. *Intrinsic* properties are those that inherently characterise an object, and they do not depend on any other object. *Extrinsic* properties are usually assigned by some external agent, and thus are not inherent. *Relations* and *functions* (we disregard here the difference between them) describe the interaction between concepts or properties of a concept. *Instances* are the individuals represented by the concepts, and finally *axioms* set constraints on concepts or instances.

In the remainder of this article the terms *concept* and *class* are used as synonyms, as are *properties* and *attributes*. Concepts are organised into a strict *is-a* hierarchy, so that properties are inherited from a class to its subclasses. In the remainder we use the terms subclass and subconcept as synonyms.

Concepts can be described not only in terms of their properties but also in terms of their formal properties (Welty & Guarino, 2001) as defined by *formal ontology* (Cocchiarella, 1991). Formal properties are the main tools of formal ontological analysis, which builds on and integrates previous methodologies to build ontologies, such as those introduced by Uschold and Gruninger (1996) and Gómez-Pérez (1998) and complements them by adding formal ontological tools that can help in clarifying the analysis. The task of the formal ontological analysis is to establish the ontological properties that constrain the way in which the representational primitives are used to model the domain. A full description of this methodology can be found in Welty and Guarino (2001).

The reason why such constraints are needed is that when modelling a domain there are a number of ways of modelling the same knowledge and the choice of one approach over the others is left to the experience and the background knowledge of the conceptual modeller. The formal tools of ontological analysis are some basic notions of formal ontological properties that have been studied in philosophy for centuries. By establishing which of these properties hold we can check whether the choices made in modelling the concepts included in the ontology and in structuring the concepts' hierarchy are sound.

The tools of formal ontological analysis are four, namely: *identity*, *unity*, *essence* and *dependence*.

Identity Identity is the logical relation of numerical sameness,[1] in which a thing stands only to itself. Based on the idea that everything is what it is and not anything else, philosophy has tried for a long time to identify the criteria which allow a thing to be identified for what it is even when it is cognised in two different forms, by two different descriptions and/or at two different times (Hirsch, 1982; Wiggins, 1967). This comprises both aspects of finding constitutive criteria (what features a thing must have in order to be what it is), and of finding reidentification criteria (what feature a thing has to have in order to be recognised as such by a cognitive agent). These are distinct, although equally important aspects of identity. Although the problem of *identifying* what features an entity should have in order to be what it is and recognised as such has been central to philosophy, it did not have the same impact in conceptual modelling and more generally in AI. The ability to identify individuals is central to the modelling process; more precisely, it is not the mere problem of identifying an entity in the world that is central to the ontological representation of the world, but the ability to reidentify an entity in all its

[1] Philosophy distinguishes qualitative sameness, where two objects exhibit the same properties but are different, and numerical sameness, when the objects are actually the same object.

possible forms or, more formally, reidentification in all possible worlds.[2] That is, the problem is related to distinguishing a specific instance of a concept from its siblings on the basis of certain *characteristic properties* which are unique and intrinsic to *that instance* in its whole. Intrinsic properties correspond to the modelling primitive *attributes*. Extrinsic properties represent relations between classes, thus corresponding to the modelling primitive *relationship*.

This notion is, of course, inherently time-dependent, since time gives rise to a particular system of possible worlds where it is highly likely that the same instance of a concept exhibits different features.[3]This problem is known as *identity through change*: an instance of a concept may remain the same while exhibiting different properties at different instants of time. Therefore it becomes important to understand which features or properties can change and which cannot (Welty & Guarino, 2001), and also the situations that can trigger such changes. If we reformulate the identity problem as *reidentification* we realise that reidentification is also affected by time; how can we reidentify the same instance at different instants of time? We face the reidentification problem in everyday life; we are able to recognise the features that permit us to distinguish an instance from others, and when intrinsic features are not available, we "attach" artificial features, that permit us to establish identity. One example is the *student ID*, which is assigned to university students, in order to identify students univocally.

Unity The notion of *unity* is often included in a more generalised notion of identity, although these two notions are different. While identity aims to characterise what is unique for an entity of the world when considered as a whole, the goal of unity is that of distinguishing the *parts* of an instance from the rest of the world by means of a *unifying relation* that binds them together (not involving anything else) (Welty & Guarino, 2001). For example, the question "is this my car?" represents a problem of identity, whereas the question "is the steering wheel part of my car?" is a problem of unity. Also the notion of unity is affected by the notion of time; for example, can the parts of an instance be different at different instants of time? This problem is also known as *individuation*.

Essence The notion of *essence* is strictly related to the notion of *necessity* (Kant, 1965). An *essential property* is a property that is necessary for an object or, in other words, a property that holds, that is true in every possible world (Lowe, 1989). Based on the notion of *essence*, Guarino and colleagues (Guarino *et al*., 1994) have introduced the notion of *rigidity*. A rigid property is a property that is necessary to all instances, a property ϕ such that: $\forall x \phi(x) \rightarrow \Box \phi(x)$.

Rigidity strictly depends on the notions of *time* and *modality* (Tamma & Bench-Capon, 2001); this point is further elaborated in Section 7.1. It is important not to confuse modal necessity with temporal permanence. Modal necessity means that the property is true in every possible world. Time is undoubtably one partition of these worlds, but temporal permanence means that the property is true in that world (time), with no information concerning the other possible worlds, and this might happen by pure chance.

Dependence In the methodology of Welty and Guarino (2001), the notion of dependence is considered related to concept properties. In this context, dependence permits us to distinguish between *extrinsic* and *intrinsic* properties based on whether they depend on objects other than the one they are ascribed to or not.

3 Problem definition

Interoperable agent systems depend upon the agents' ability to reconcile their views of the world, and to *share knowledge*. *Knowledge-sharing* denotes sharing of the same knowledge by multiple agents (be they human or software), across multiple applications, and in multiple contexts at the same time

[2] Some philosophers, e.g. Lewis (1993, 39 ff), hold that there is no such thing as trans-world identity, although objects in one world can have *counterparts* in other worlds.
[3] Here the counterpart theory does not hold, and so identity through time is always accepted.

(Uschold *et al*., 1998). This is accomplished by "combining" the ontologies associated with the agents comprising the system. Combining here means using two or more different ontologies developed for different tasks in which their mutual relation is relevant (Klein, 2001). The combined use of independently developed ontologies has been object of previous work (Fridman Noy & Musen, 1999; McGuinness, 2000; Pinto *et al*., 1999) and it is often achieved by *integrating* or *merging* ontologies. Here we follow Klein (2001) and define *integration* and *merging* as synonyms that indicate the creation of a new ontology from two or more existing ontologies with overlapping parts, which can be either virtual or physical.

Ontology integration is consists of the following steps (which may be iterated) (McGuinness *et al*., 2000)

1. find the places in the ontologies where they overlap;
2. relate concepts that are semantically close via equivalence and subsumption or instance relationships (aligning); and
3. check (at least partially) the consistency, coherence and non-redundancy of the result.

In this work we are only concerned with step 1, the coalescence of two semantically identical terms in different ontologies so that they are referred to by the same name in the resulting ontology. The coalescence of terms in diverse ontologies has to be accomplished bearing in mind that since agents are highly heterogeneous, their ontologies can be heterogeneous, so any kind of heterogeneity has to be reconciled in order to permit knowledge sharing.

Heterogeneity has long been recognised as the major obstacle to knowledge sharing and previous work has illustrated the different types of heterogeneity that can affect agents (Chalupsky, 2000; Gómez-Pérez, 1998; Klein, 2001; Neches *et al*., 1991; Visser *et al*., 1998).

In this work we take the view illustrated by Klein (2001), and we classify the different types of heterogeneity that can affect agents as follows:

- **Language heterogeneity** Language heterogeneity occurs when ontologies written in different ontology languages are integrated or combined. It can be caused by differences in the syntax, logical representation, semantics of primitives and expressivity of the languages used to represent the ontologies. For example, a language might permit the expression of disjointedness explicitly, for instance by using the statement `(disjoint A, B)`, whereas another might represent the same constraint by negation in the class declaration, e.g. `A subclass-of (NOT B)` $\wedge$ `B subclass-of (NOT A)`. More specifically, this is a case of logical representation heterogeneity (Klein, 2001). This is the kind of heterogeneity that KIF aims to resolve (Neches *et al*., 1991)

- **Ontology heterogeneity** Ontology heterogeneity occurs when agents make different ontological assumptions about the overlapping domains. In particular, ontology heterogeneity can occur while *conceptualising* and/or *explicating* (Visser *et al*., 1998) the domain. Both conceptualisation and explication heterogeneity can be further subdivided, but such detailed classification of the possible causes of heterogeneity is out of the scope of this article. Conceptualisation heterogeneity is due to semantic differences arising from different conceptualisations of the concepts and the relations in the agents domains. For example, one agent can model the concept `Wine` into the subconcepts `White-Wine` and `Red-Wine` and then further specify the two. Therefore the concept `White-Wine` may have two subconcepts `Chardonnay` and `Riesling`; analogously the concept `Red-Wine` may be subdivided into the subconcepts `Beaujolais` and `Chianti`. Another agent might model the concept `Wine` into the subclasses `White-Wine` and `Red-Wine` only. Another example of conceptualisation heterogeneity can occur when one conceptualisation subdivides the concept `Person` into the subconcepts `Male` and `Female` whereas the other subdivides it into the subconcepts `Child`, `Teenager` and `Adult`. It concerns only concepts.

  Explication heterogeneity is due to differences in the specification of the domain. So different ontologies may share the same conceptualisation of a domain but differ in the way in which this is specified. During the conceptualisation phase the concepts describing the domain are selected, in the explication phase these concepts are made explicit. For example, let us suppose there are two

ontologies O_1 and O_2 that describe the vehicles domain, and that both model the concept *Car* by calling it `Automobile` in ontology O_1 and `Motor-car` in ontology O_2. The two concepts can be described by the same set of attributes, for example `Registration-Year`, `Maximum-Speed`. To make the problem more complex the concepts may be defined in terms of attributes that are semantically equivalent. For example `Automobile` in ontology O_1 could be described by the set of attributes `Reg-Year` and `Max-Speed`, while the concept `Motor-car` in ontology O_2 is described by the set of attributes `Registration-Year` and `Maximum-Speed`. A few research efforts have specifically addressed ontology heterogeneity, among them the KRAFT project (Preece *et al*., 2001) and OBSERVER (Mena *et al*., 2000).

Heterogeneity, and especially ontology heterogeneity, can seriously hinder attempts to share knowledge automatically between agents. In fact, in order to recognise whether two concepts from heterogeneous knowledge sources are similar, we cannot only rely on the the terms denoting them and on their descriptions, but we need to have a full understanding of the concepts in order to decide whether they are semantically related or not. As noted by McGuinness (2001), an explicit representation of the semantics of terms would be useful to understand whether two concepts are similar. By semantics of a term we refer here to a precise and unambiguous description of the meaning of the concept represented by the term. It emerges that the current ontology models are not expressive enough to provide such an explicit representation of the semantics. In fact, ontologies with the richest expressiveness currently are those organised in formal is-a hierarchies (that is, strict subclass hierarchies), including formal instance relations (those relating instances to the concepts they instantiate), whose concepts are desribed in terms of their characterising properties, which permit restrictions on the values associated with the properties and which, finally, permit the expression of logical constraints on the property values (Lassila & McGuinness, 2001). Even this kind of expressiveness does not, however, allow a full account of the semantics of the concepts described.

We realise that any attempt to provide a representation of the precise meaning of each concept will prove extremely difficult, especially if this representation is to be processable by a computer. In the attempt to find a mid-point between the quest for a richer semantics and the ontology models currently available we propose here to enrich the classic ontology model with further information aiming to give a more precise characterisation of concepts and of their properties. The enriched model we propose focuses on providing the following characterisations:

- *Attribute behaviour over time* This information aims to give a better description of attributes by characterising their behaviour over time, that is, whether they are allowed to change their value during the concept lifetime, whether the change is reversible and what triggers change.
- *Modality* This information is a qualitative description of the degree of inheritability of a concept's property by its subconcepts.
- *Prototypes* and *exceptions* This information aims to describe properties that are prototypical for a concept, that is, the properties that obtain for the *prototypical* (from a cognitive viewpoint, according to Rosch (1975)) instances of a concept. Exceptions are those properties which can be ascribed to a concept despite being highly unusual.

This work claims that an ontology model which includes this type of property characterisation might be helpful to dealing with ontological heterogeneity problems in two ways. On the one hand, the model permits the identification of formal ontology properties and therefore it supports all methodologies for building ontologies, including that by Welty and Guarino (2001), which is thought to lead to cleaner taxonomies which are easier to integrate. On the other hand, this conceptual model for ontologies facilitates a better understanding of the concepts' semantics. Currently any kind of integration is performed by hand, based on the expertise of the knowledge engineers and on the ontology documentation. Even in this case the ontology model we propose can prove useful by providing a characterisation of the properties, which can help to identify semantically related terms.

There are prospects for a semi-automatic integration of independently developed ontologies. For example, two systems such as Chimaera (McGuinness *et al*., 2000) and smart (Fridman Noy & Musen,

1999) partially deal with language heterogeneity, but a lot of work has to be done in order to permit semi-automatic integration.

4 Enriched ontology model

The ontology model we propose comprises *concepts*, *attributes*, *relations* and *instances*. Concepts represent the entities of the domain and the tasks we want to model in the ontology. Concepts are described in terms of defining properties and are represented by associating each *attribute* with either a single value or a set of values. Concepts are organised into an is-a hierarchy, so that a concept's attributes and their values are inherited by the subconcepts. Multiple inheritance is permitted, so attributes and their values can be inherited from multiple parents. The values associated with an attribute can be restricted in order to provide a better definition of a concept (Lassila & McGuinness, 2001). Restrictions can also be defined when attributes and their associated values are inherited by subconcepts. In such a case, value restrictions model the the fact that a subconcept might be characterised by a more restrictive property. For example, the concept *Wine* may be described by the attribute *Sugar-Content* whose possible values are *Sweet*, *Medium*, *Dry*. We may define the subconcept *Dessert-Wine* which inherits the sugar content from being a kind of wine, and is characterised by the property of being either a sweet or a medium wine, which might be modelled by associating the restricted set of values *Sweet*, *Medium* to the attribute *Sugar-Content* in the concept *Dessert-Wine*.

Attributes are described in terms of their structural characteristics, such as the concepts that they are defining, their allowed values, the type of the values (string, integer and so on), and the maximum and minimum values (if attributes are numeric). Attributes are also described in terms of their behaviour over time (whether their values can change over time and under which conditions, whether this change is regular or is permitted once only, and whether the change is reversible); their behaviour with respect to inheritance, that is, a qualitative estimate of the subconcepts inheriting the attribute and its value(s); and their prototypical and exceptional values. Prototypical and exceptional values provide contextual information concerning the attribute as discussed in Section 7.4.

Relations between concepts are supported by the model: a relationship R between a concept C_1 and a concept C_2 is represented by an attribute of the concept C_1 whose value type is C_2. For example, if we consider the concepts *Wine* and *Winery*, we can define the relationship *Produces* which associates a winery with a wine. This is represented by adding to the concept description of *Winery* the attribute *Produces* whose value type is *Wine*.

Our ontology model also includes instances. Lassila and McGuinness (2001) describe ontologies including formal instance relations; we consider instances as a natural extension of ontologies enforcing a strict hierarchical structure. We model instances as concepts; however, when formal instance relations hold, the ontology includes also the content about grounding individuals and their relationships with the concept they instantiate. Finally, the ontology model supports roles. Concepts are also used to represent *roles*, which can be thought of as describing the *part played* by a concept in a context, (a more complete discussion on roles is postponed to Section 7.2). So we maintain a frame-like representation for roles as well. Roles are described in terms of their context, and the formal role relationship holds, that is, roles are related to concepts by role-of relations. It is important to note that we are not concerned with the problem of supporting a role representation in the syntax of the ontology model.

This ontology model enriches the traditional model proposed initially by Gruber (1993), in that it permits the characterisation of a concept's properties. From this viewpoint it should be more expressive. The solution of adding information characterising concepts' properties is a controversial one. We do realise that often it is not true that "more is better". However, there is a fundamental reason for enriching the ontology model with characterisations of the attributes.

An ontology is an a-priori account of the "things" that are in a domain and the relationships modelling the structure of the world seen from a particular perspective. In order to provide such an account one has to understand the concepts that are in the domain, and this involves a number of things. It involves knowing what can be sensibly said of a thing falling under a concept. This can be

represented by describing concepts in terms of their properties, and by giving a full characterisation of these properties. Thus, when describing the concept *Bird* it is important to distinguish that some birds fly and others do not. A full understanding of a concept involves more than this, however; it is important to recognise which properties are *prototypical* (Rosch, 1975) for the class membership and, more importantly, which are the permitted exceptions. There are, however, differences in how confident we can be that an arbitrary member of a class conforms to the prototype: it is a very rare mammal that lays eggs, whereas many types of well-known bird do not fly.

Understanding a concept also involves understanding how and which properties change over time. This dynamic behaviour of attributes also forms part of the domain conceptualisation and can help to identify the *meta-properties* holding for the concept. It is worth pointing out that we are not referring to the dynamic evolution of ontologies, but to the fact that attributes of a concept might change their value(s) during the existence of the concept. This information concerns the concept and should be reflected by its ontological definition. It might be argued that this kind of of knowledge has not an *ontological* nature, but rather an *epistemic* one, and to some extent we do agree with this criticism. But ontologies, especially when considered in the context of multi-agent systems, should provide an actual account of the agent's view of a domain. Communication in agent system rely heavily on the agents' ability to share knowledge, and we believe that the ability of ontologies to facilitate sharing and reuse of knowledge and reasoning with ontological knowledge and its instances can be improved if the formal meta-level of the description is complemented by a richer concept description.

When we consider ontologies from a pure philosophical perspective, they are an a-priori description of what constitutes necessary truth in all possible worlds (Kripke, 1980). It is this formal stance on ontologies that makes it possible to add to ontologies a meta-level of description and thus to reason about meta-properties (Welty & Guarino, 2001).

Our view on ontologies, especially when these make an agent's conceptualisation of the domain explicit, is that ontologies should provide sufficient information to enable agents to have a full understanding of a concept as it is in the domain (that is, in the real world), and thus they are fully grounded in the agent's reality, but should also enable knowledge engineers to perform a formal ontological analysis on these concepts. If ontologies are seen in this perspective, then the boundary between what is to be considered ontological knowledge and what is epistemic knowledge becomes blurred.

5 Implementing the ontology model

In this section we describe how we have implemented the ontology model discussed in Section 4 in a frame-based knowledge model. The knowledge model we use is inspired by OKBC (Chaudhri *et al*., 1998), and therefore supports a frame-based representation of knowledge. It is worth noting at this point that we have used the OKBC model only as a support for the proof of concepts and that there are some differences between the model we propose and the OKBC knowledge model. Our aim is not to build a new knowledge model for ontologies but to support a semantically enriched description of attributes when defining concepts in ontologies.

In frame-based systems *frames* represent objects, classes of related objects or general concepts (predicates) (Karp, 1993). Frame-based systems might recognise just one type of frame or two types of frame, such as *class frames* and *instance frames*, where the former represent classes, or sets of things sharing the same features, while the latter represent particular instances of things. Typically, frames are organised into a hierarchy that can be single or multiple. A parent frame is more general than its children, and a collection of frames in one or more inheritance hierarchies is a *knowledge base*.

Frames are described by components called *slots*, which model attributes or properties of the things represented by the frame. Slots can also represent relationships between the frame where the slot is defined and another frame. Slots are associated with values; restrictions on allowable values may also be part of a slot definition. Besides name, values and value restrictions, slot definitions have other components which specify other characteristics of the frame. These components are called *facets*. Slot

definitions propagate down the inheritance hierarchy, thus name, values, value restrictions and facets are inherited from parent to child frames.

The knowledge model implementing the ontology model described in Section 4 is frame-based. This model focuses on concepts, therefore all frames are class frames, also called *classes*. *Classes* correspond to concepts and to roles (see Section 7.2) that these concepts can play, hierarchically organised into a multiple inheritance hierarchy, linked by is-a links. Inheritance with exception is permitted, therefore the value associated with a slot inherited from a more general frame can be overridden in a more specific frame (Touretzky, 1986). Classes are described in terms of *slots*, or attributes, whose values can either be sets or single values. A slot is described by a name, a domain (the class frame to which the slot is associated), a value type (the kind of values it may take, such as string, integers and so on) and by a set of additional constraints, here called *facets*. Facets can contain the documentation for a slot, constrain the value type or the cardinality of a slot and provide further information concerning the slot and the way in which the slot is to be inherited by the subclasses.

In the following sections we describe in more detail our knowledge model. This description is not meant to be exhaustive, but just to give an example of how the enriched semantic conceptual model could be implemented in an OKBC-like knowledge model. For this reason, we describe in the detail only the suggested extensions to OKBC.

5.1 The additional facets

The extended semantics has been represented by additional facets augmenting those already provided by OKBC. These facets, as those in OKBC, are not mandatory, but they can be optionally filled in. These facets are not mapped by corresponding slots into slot frames since the information encompassed in the additional facets makes sense only when a slot is associated with a frame, whereas it is undefined when the slot is considered on its own.

Class type The facet:class-type has been added to the OKBC facets to specify whether the class that is being defined is a concept or a role. This facet can take two possible values, `concept` and `role`, which are used to change the meaning of some of the frame facets.

Value label The value associated with the facet:value-label of slot S of frame F is one or more elements from the set of keywords and keyphrases: {`Inherited`, `Inherited with exceptions`, `Distinguishing`, `Value`}. If the value or set of values associated with the facet:value-label of slot S of frame F is `Inherited` this means that the value associated with S has been inherited unchanged from some super class, whereas if it is `Inherited with exceptions` then the slot inherits the value from its parent frame, with some modifications (for example restriction on the domain). If the slot value is labelled through the facet:value-label as `Distinguishing` this means that it is a value that differentiates among siblings with a common super class. For example, let us suppose that the class *Wine* is described by the slot *Colour*, whose allowed values are {*Red*, *White*}. Let us define two classes, *Red-Wine* and *White-Wine*. Both classes are subclasses of *Wine* and therefore they inherit the slot *Colour*, but this is inherited with value restrictions, therefore the value associated with this slot is *Red* for the class *Red-Wine* and *White* for the class *White-Wine*. Since the slot *Colour* permits us to distinguish between red wine and white wine, it is labelled `Distinguishing`.

If the slot value is labelled `Value` it means that the value is neither prototypical nor inherited or distinguishing.

It should be noted that inherited and distinguishing values are incompatible in the same concept description, that is, a value is either inherited or distinguishing but cannot be both. On the other hand a value can be prototypical and inherited. Distinguishing values become inherited for subclasses of the class. Of course also for distinguishing values, it may be that only inheritance does not concern the whole range of values, but only a subrange.

Value prototypes The facet :VALUE-PROTOTYPES of slot S of frame F specifies which values of slot S are considered *prototypical* in the context specified by the frame F, that is, the set of those values that are normally (in the conception of the ontology designers) associated with the slot S when this is

describing the concept at frame F. This enables the ontology designers to express what is believed to be normal from their perspective. Therefore the values associated with the slot S at frame F are those describing any prototypical instance of the class, but exceptions are permitted with a degree of credibility expressed by the slot :MODALITY (see also the facet :VALUE-EXCEPTIONS). For example, let us suppose we are modelling the concept `Blood-Circulation`, which is described by a slot `Blood-Pressure` whose allowed value is one or a set of pairs (*s*, *d*), where *s* is the value of the *systolic pressure* and *d* is the value of the *diastolic pressure*. The prototypical values (the possible values of blood pressure for a healthy individual over 18) are between 90 and 130 for systolic pressure and between 60 and 85 for diastolic pressure. This notion of prototypical values is related to the analogous notion in cognitive science (Rosch, 1975) and is discussed, together with the notion of exception, in Section 7.4.

Value exceptions The facet :VALUE-EXCEPTIONS of slot S of frame F specifies which set of possible values of slot S are to be considered exceptional, that is, those values that are permitted in the concept description because they are in the domain, but deemed exceptional from a common-sense viewpoint. The exceptional values are those which are possible but highly unlikely. The values that this facet can take are therefore a subset of the values associated with the slot S. Not all values that are not prototypical are exceptional. Let us consider again the blood pressure example, exceptions are those values registered for people affected by conditions such as hypertension or hypotension and so are in the range of values for the slot `Blood-Pressure` but outside the range determined by the prototypical values. That is, exceptional values for systolic pressure are those in the range of the slot that are smaller than 90 and greater than 130, whereas for diastolic pressure, the exceptional values are those smaller than 59 and greater than 85.

Value modality The facet :VALUE-MODALITY of slot S of frame F denotes the degree of confidence in the fact that the slot is associated with a some specified values. It describes the class membership conditions and the degree of inheritability of the property modelled by the pair slot-value(s). This value makes sense only for class descriptions and not for instance descriptions.

The value associated with this facet is a nonnegative integer between 1 and 7. Each of these numbers is associated with a specific meaning. The possible values associated with this slots are reported below together with an example showing cases in which each of the values apply:

1. `All` Let us assume we have a frame `Person` which is described by the property *has fingerprints* modelled by associating the value `Yes` to the slot :HAS-FINGERPRINTS. The property of having fingerprints is inherited by all subclasses, that is, all the subclasses of `Person` (such as `Child`, `Teenager`, `Adult` and so on) have fingerprints. This kind of information is described by associating the filler `All` with the facet :VALUE-MODALITY when describing the slot :HAS-FINGERPRINTS associated with the value `Yes`.
2. `Almost all` A classic example given in knowledge representation of a property which holds for *almost all* the subconcepts of the concept which is being described is a mammal's ability to give birth to live young. In fact all species of mammals give birth to live young with the exceptions of a particular order, called *monotremes*, which do not. If we were to model this situation, the slot :ABILITY-TO-GIVE-BIRTH-TO-LIVE-YOUNG would be associated with value `Able` and would be described by value `Almost all` associated with the facet :VALUE-MODALITY.
3. `Most` The filler `Most` is to be used in those cases where the majority of subclasses inherit the property. For example, let us suppose to consider the concept *Cat*. The majority of cats have short hair, although there is a considerable number of cat varieties which have long hair. If we chose to model the concept by associating with it the slot :HAS-SHORT-HAIR associated with the value `Yes`, then such a slot would be described by associating the filler `Most` to the facet :VALUE-MODALITY.
4. `Possible` In some cases, however, we might not have any information concerning the degree of applicability of a property to the subconcepts. For example, let us consider the concept *university professor*. In some countries, e.g. Italy, it is not always the case that in order to be a professor one has to be awarded a Ph.D. On the other hand, in some other countries, e.g. the UK or the United States, it is often the case that a professor has a PhD. If we had to model the concept *professor* the

property *has a phd* would be described by associating the value `Possible` to the modality facet, in that it is possible that the property holds for some of the subclasses of the concept, but we do not have information concerning the specific subclasses for which the property holds and neither do we know for how many of them.

5. `A Few` This has the opposite semantics of `Most`.
6. `Almost none` This has the opposite semantics of `Almost all`.
7. `None` This has the opposite semantics of `All`.

It is worth noting that the value `None` associated with this facet is tantamount to negation. The value `None` as possible filler for the slot value-modality makes sense especially in the context of inheritance of distinguising properties, that is, a property that is distinguishing may be inherited by all subconcepts of the concept for which the property is defined. However, such property does not hold for all subconcepts. Such a filler has also been added to the model under the hypothesis that such a model will be used to support semi-automatic inconsistencies resolution. In situations different from those described above, it would make no sense for a knowledge engineer to include in the concept description a property whose degree of applicability to subclasses is `None`.

Value change frequency The facet :VALUE-CHANGE-FREQUENCY of slot S of frame F specifies whether and how often the value of slot S changes during the lifetime of the concept which is represented by the frame F. The value associated with this slot is an element of the set: {`Regular`, `Once only`, `Volatile`, `Never`}.

If the value of the slot is `Regular` it denotes that the change process is continuous, for instance the age of a person can be modelled as changing regularly. If the facet value is to `Once only` it means that only one change over time for the value of slot S is possible, while if the value of the slot is `Never` it specifies that the value of the slot S is set only once and then it cannot change again, for example a person's date of birth once set cannot change again. Finally `Volatile` means that the change process is discrete and can be repeated at irregular intervals; thus the attribute's value can change more than once, as for example people can change job more than once.

Value-change-events The :VALUE-CHANGE-EVENTS slot specifies the conditions under which the values associated with slot S change. It is associated with a set of quadruples

$$\{((E_j, S_j, V_j), R_j) \mid j = 1, \ldots, m\}$$

where E_j is an event, S_j is the one of the possible values associated with the slot (which models one of the possible states of the property represented by the pair slot-value), V_j defines the event validity and R_j denotes whether the change is reversible or not. The semantics of this facet is explained in Section 7.1. If the class describes a role, that is, the facet :CLASS-TYPE is associated with the value `role`, then the facet :VALUE-CHANGE-EVENTS defines the conditions regulating the acquisition and the relinquishing of a role. This point is further discussed in Section 7.2.

6 Expressive power of the knowledge model

The knowledge model presented in the previous section can accommodate all the modelling primitives which we considered necessary to write ontologies. As we have already mentioned, concepts are represented by classes, which are described in terms of attributes or properties. Properties are described by *slots-values* pairs. In this knowledge model slots are used to describe both *intrinsic* and *extrinsic* properties. We follow Welty and Guarino (2001) and consider an intrinsic property something inherent to an individual, not dependent on other individuals, such as having a heart or having a fingerprint. An extrinsic property is one that is not inherent and has relational nature, such as "being a friend of John'. So slot-value(s) pairs are used to describe properties holding for a class, in turn properties are described by a set *facet-filler* pairs that characterise the properties.

Properties can be divided into prototypical, necessary, distinguishing, inherited and simple value assignments. As already mentioned in Section 5, concepts are hierarchically organised according to an is-a relationship that permits property inheritance from ancestors to descendants. The properties that

are inherited from an ancestor are labelled as "*inherited*". However, inherited properties can be overruled in the more specific concept in order to accommodate inheritance with exceptions. The properties that have been overruled are labelled as "*distinguishing*", since they allow us to distinguish between siblings of the same parent concept. A property is to be considered necessary if it is essential to all instances of the concept, while it is prototypical if it holds for the prototypical instances of the concept only. The notion of essential property relates to the idea of necessary condition while prototypical properties permit us to identify prototypes, discussed in Section 7.4. Finally, a property labelled value assignment associates a value to an attribute in order to describe a specific feature of the instances of the concept, such as hair colour = brown.

Roles, already defined in Section 5, are also supported in this knowledge model; they are represented as concepts but the facet :CLASS-TYPE is set to `role`, so that we are able to distinguish them from a concept definition. As for the rest, a role has exactly the same definition as a concept since roles are described in terms of attributes that are typical of a role and are organised into an is-a hierarchy totally analogous to the one defined for concepts, where the inheritance of properties through the role hierarchy is permitted. Most of the considerations we made for concepts hold for roles as well, therefore we can consider prototypical properties for roles, distinguishing properties. What distinguishes a role from a concept is that the role holds during a specific span of time in which some property holds. For example, the role Ṣtudent' is applicable only if the property of being registered to a university holds. Therefore, the behaviour of properties over time permits us to model the acquisition and relinquishment of a role. Roles and their properties are discussed below in Section 7.2.

7 Discussion

In order to use the enriched conceptual model knowledge engineers have to provide more details concerning the concepts than if they were using a traditional OKBC-like knowledge model; they are thus guided in performing the ontological analysis, which is usually demanding. Furthermore, the enriched knowledge model forces knowledge engineers to make ontological commitments explicit, that is, make explicit agreement on the meaning of the terms used to describe a domain (Guarino, 1998). Knowledge-sharing is possible only if the ontological committment of the different agents is made explicit. Indeed, real situations are information-rich events, whose context is so rich that, as has been argued by Searle (1983), it can never be fully specified. Many assumptions about meaning and context are usually made when dealing with real situations (Rosch, 1999). These assumptions are rarely formalised when real situations are represented in natural language but they have to be formalised in an ontology since they are part of the ontological commitments that have to be made explicit. Enriching the semantics of the attribute descriptions with things such as the behaviour of attributes over time or how properties are shared by the subclasses makes some of the more important assumptions explicit.

The enriched semantics is essential to solve the inconsistencies that arise either while integrating diverse ontologies or while reasoning with the integrated knowledge. By adding information on the attributes we are better able to measure the similarity between concepts, to disambiguate between concepts that *seem* similar while they are not, and we have the means to infer which property is likely to hold for a concept that inherits inconsistent properties.

A possible disadvantage of such a semantically enriched knowledge model is the large number of facets that need to be filled when building ontologies. We realise that this can make building an ontology from scratch even more time-consuming but we believe that the outcomes balance the increased complexity of the task. Indeed, in order to fill the additional facets knowledge engineers need to have a full understanding not only of the concept they are describing but also of the context in which the concept is used. Arguably, they need such knowledge if they are to perform the modelling task thoroughly.

The evaluation of the cost to pay for this enriched expressiveness and of the kind of reasoning inferences permitted by this model are strictly dependent on the domain and the task at hand. In some

cases, a simple subsumption inference would be required (for example to identify common ancestors in the hierarchy), whereas we can imagine that the automatic coalescence of terms might require more sophisticated inferences whose cost we cannot evaluate a priori. In some other cases, the simple matching between properties' characterisations might help in establishing or ruling out the possiblity of semantic relatedness. For example, two concepts are described by the same properties but with different characterisations; this might indicate that the concepts have been conceptualised differently.

7.1 Representation of attributes' behaviour over time

In the knowledge model the facets :VALUE-CHANGE-FREQUENCY and :VALUE-CHANGE-EVENT describe the behaviour of *fluents* over time, where the term *fluent* is borrowed from situation calculus to denote a property of the world that can change over time. Modelling the behaviour of fluents corresponds to modelling the changes in properties that are permitted in a concept's description without changing the essence of the concept. The behaviour over time is closely related to establishing the *identity* of concept descriptions (Welty & Guarino, 2001), in that some properties can change without affecting the identity of the changing individual. Describing the behaviour over time also involves distinguishing properties whose change is *reversible* from those whose change is *irreversible*.

Property changes over time are caused either by the natural passage of time or are triggered by specific event occurrences. We need, therefore, to use a suitable temporal framework that permits us to reason with time and events. The model chosen to accommodate the representation of the changes is the *Event Calculus* (Kowalski & Sergot, 1986). Event calculus deals with local event and time periods and provides the ability to reason about change in properties caused by a specific event and also the ability to reason with incomplete information.

Changes of properties can be modelled as *processes* (Sowa, 2000). Processes can be described in terms of their start and end points and the changes that happen in between. We can distinguish between *continuous* and *discrete changes*, the former describing incremental changes that take place continuously while the latter describe changes occurring in discrete steps called *events*. Analogously we can define *continuous properties* to be those changing regularly over time, such as the age of a person, versus *discrete properties* which are characterised by an event which causes the property to change. If the value associated with change frequency is `Regular` then the process is continuous, if it is `Volatile` the process is discrete and if it is `Once only` the process is considered discrete and the triggering event is set equal to *time-point* = *T*.

Any regular occurrence over time can be, however, expressed in form of an event, since most of the forms of reasoning for continuous properties require discrete approximations. Therefore in the knowledge model in Section 5, continuous properties are modelled as discrete properties where the event triggering the change in property is the passing of time from the instant t to the instant t'. Each allowed change of property is represented by a set of quadruples $\{((E_j, S_j, V_j), R_j) | j = 1, \ldots, m\}$ where E_j is an event, S_j is the state of the attribute-value pair associated with a property, V_j defines the event validity and R_j is a value in the set $\{Regular, Irregular\}$ and indicates whether the change in properties triggered by the event E_j is reversible or not. The model used to accommodate this representation of the changes adds reversibility to *Event Calculus*, where each triple (E_j, S_j, V_j) is interpreted either as the concept is in the state *Sj* before the event E_j happens or the concept is in the state S_j after the event E_j happens depending on the value (either "before" or "after") associated with V_j. The interpretation is obtained from the semantics of the event calculus, where the former expression is represented as $Holds(before(E_j, S_j))$ while the latter is represented as $Holds(after(E_j, S_j))$.

The idea of modelling the permitted changes for a property is strictly related to the philosophical notion of *identity.* In particular, the knowledge model addresses the problem of modelling identity when time is involved, namely *identity through change*, which is based on the common-sense notion that an individual may remain the same while showing different properties at different times (Kant, 1965). The knowledge model we propose explicitly distinguishes the properties that can change from those which cannot, and describes the changes in properties that an individual can be subjected to, while still being recognised as an instance of a certain concept.

Events in this representation are always *point events*, and we consider *durational events* (events which have a duration) as being a collection of *point events* in which the attribute has the value specified by V_j as long as the event lasts. The duration is determined by the definition of an *event* in *Event Calculus*, where for each event is given an initial and a final time point. We realise that this representation oversimplifies the dynamics of process changes and we aim to investigate a more sophisticated change representation as future work.

The notion of changes through time is also important in order to establish whether a property is *rigid*. A *rigid property* is defined in Guarino *et al.* (1994) as a property that is essential to *all* its instances, whereas with *essential property* we mean a property holding for an individual in every possible circumstance in which the individual exists. The interpretation that is usually given to *rigidity* is that if x is an instance of a concept C then x has to be an instance of C in every possible world (Kripke, 1980). Here we specifically concentrate on one of these systems of possible worlds, *time*.

In Tamma and Bench-Capon (2001) we have related the notion of *rigidity* to those of *time* and *modality* and in Section 7.3 we show that, by using the information represented in the slot :VALUE-MODALITY and that concerning the behaviour over time, we can precisely identify rigidity in the subset of the set of possible worlds. By characterising the rigidity of a property in this subset of possible worlds we aim to provide knowledge engineers with the means to reach a better understanding of the *necessary* and *sufficient* conditions for the class membership. However, this does not mean that the rigidity of a property depends on any account of whether the property is used to determine class membership or not. That is, the final aim is to try to separate the properties constitutive of identity from those that permit reidentification.

7.2 The need for identity and rigidity: roles

Establishing whether rigidity holds for a property is not only central in order to distinguish necessary truth but also to recognise *roles* from concepts. The notion of *role* is as central to any modelling activity as those of *objects* and *relations*.

A definition of role that makes use of the formal meta-properties and includes also the definition given by Sowa (1984) is provided by Welty and Guarino. In Welty and Guarino (2001) they define a role as:

> the properties expressing the part played by one entity in an event, often exemplifying a particular relationship between two or more entities. All roles are *anti-rigid* and *dependent* . . . A property ϕ is said to be anti-rigid if it is not essential to *all* its instances, i.e. $\forall x \phi(x) \rightarrow \neg\Box\phi(x)$ . . . A property ϕ is (*externally*) *dependent* on a property φ if, for all its instances x, necessarily some instance of φ must exist, which is not a part nor a constituent of x, i.e. $\forall x \Box(\phi(x) \rightarrow \exists y \varphi(y) \wedge \neg P(y, x) \wedge \neg C(y, x))$

where $P(y, x)$ denotes that y is a *part* of x while $C(y, x)$ denotes that y is a *constituent* of x. In other words a concept is a role if its individuals stand in relation to other individuals, and they can enter or leave the extent of the concept without losing their identity. From this definition it emerges that the ability to recognise whether rigidity holds for some property ϕ is essential in order to distinguish whether ϕ is a role.

Steimann (2000) compares the different characteristics that have been associated in the literature with roles. From this comparison it emerges that the notion of role is inherently temporal; indeed roles are acquired and relinquished dependent on either time or a specific event. For example, the object *person* acquires the role *teenager* if the person is between 13 and 19 years old, whereas a person becomes *student* when they enrol for a degree course. Moreover, from the list of features in Steimann (2000) it emerges that many of the characteristics of roles are time − or event-related: an object may acquire and abandon roles dynamically, may play different roles simultaneously or may play the same role several times, simultaneously, and the sequence in which roles may be acquired and relinquished can be subjected to restrictions.

Roles may be "naturally" determined when social context is taken into account, and the social context determines the way in which a role is acquired and relinquished. For example, the role of `President of the country` is relinquished differently depending on the context provided by the country. So, for example, in Italy the role may be acquired and relinquished only once in the lifetime of an individual, whereas if the country is the United Sates, the role can be acquired and relinquished twice, because a president can be re-elected. Social conventions may also determine that once a role is acquired it cannot be relinquished at all. For example, the role `Priest` in a Catholic context is relinquished only with the death of the person playing the role.

For the aforementioned reasons, ways of representing roles must be supported by some kind of explicit representation of time and events. The knowledge model we have presented provides sufficient semantics to model the dynamic features of roles. Indeed the model provides a way to explicitly represent time intervals which can be used to used to model roles as fluents; moreover, the facet :VALUE-CHANGE-EVENT gives knowledge engineers the ability to model events, which describe the events that constrain the acquisition or the relinquishing of a role.

The ability to distinguish roles gives also a deeper understanding of the possible contexts in which a concept can be used. Recognising a role can be equivalent to defining a context, and the notion of context is the basis on which prototypes and exceptions are defined.

7.3 Modality: weighing the validity of attributes' properties

The term modality is used to express the way in which a statement is true or false, which is related to establishing whether a statement constitutes a *necessary truth* and to distinguishing necessity from possibility (Kripke, 1980). The term can be extended to qualitatively measure the way in which a statement is true by trying to estimate the number of possible worlds in which such a truth holds. This is the view we take in this work, by denoting the degree of confidence that we can associate with finding a certain world with the facet :VALUE-MODALITY of slot S of frame F. This notion is quite similar to that of *rankings* as defined by Goldszmidt and Pearl (1996, 60): "Each world is ranked by a non-negative integer κ representing the degree of surprise associated with finding such a world".

Here we use the term modality to denote the degree of surprise in finding a world where the property P holding for a concept C does not hold for one of its subconcepts C'. The additional semantics encompassed in this facet is important for reasoning with statements that have different degrees of credibility. Indeed there is a difference in asserting facts such as "mammals give birth to live young" and "birds fly", the former is generally more believable than the latter, for which many more counterexamples can be found. The ability to distinguish facts whose truth holds with different degrees of strength is important in order to find which facts are true in every possible world and therefore constitute *necessary truth*. The concept of necessary truth brings us back to the discussion about *rigidity*, in fact it can be assumed that the value associated with the :VALUE-MODALITY facet together with the temporal information on the changes permitted for the property can determine whether the property described by the slot is a rigid property. In particular, we can exactly determine rigidity in a subset of all possible worlds. Indeed, since an ontology defines a vocabulary, we can restrict ourselves to the set of possible worlds which is defined as the set of maximal descriptions obtainable using the vocabulary defined by the ontology (Plantiga, 1989). Then, under the assumption of restricting the discourse to this set of possible worlds, *rigid properties* are those whose :VALUE-MODALITY facet is equal to `All` and cannot change in time, that is, whose :VALUE-CHANGE-FREQUENCY facet is set to `Never`.

The ability to evaluate the degree of confidence in a property describing a concept is also related to the problem of reasoning with ontologies obtained by integration. In such a case, inconsistencies can arise if a concept inherits conflicting properties. In order to be able to reason with these conflicts some assumptions have to be made, concerning how likely it is that a certain property holds; the facet :VALUE-MODALITY models this information by modelling a qualitative evaluation of how subclasses inherit the property. This estimate represents the common sense knowledge expressed by linguistic quantifiers such as `All`, `Almost all`, `Few` and so on.

In case of conflict the property's degree of credibility can be used to rank the possible alternatives following an approach similar to the non-monotonic reasoning approach presented in Goldszmidt & Pearl (1996): in case of more conflicting properties holding for a concept description, properties are ordered according to the degree of credibility, that is, a property holding for all the subclasses is considered to have a higher rank than one holding for few of the concept subclasses. This ordering of the conflicting properties needs to be validated by the knowledge engineer, although it reflects the common-sense assumption that, when no specific information is known, people assume that the most likely property holds for a concept.

7.4 Prototypes, exceptions and concepts

In order to get a full understanding of a concept it is not sufficient to list the set of properties generally recognised as describing a typical instance of the concept but we do need to consider the known exceptions. In this way, we partially take the cognitive view of prototypes and graded structures, which is also reflected by the information modelled in the facet :VALUE-MODALITY. In this view all cognitive categories show gradients of membership which describe how well a particular subclass fits people's idea or image of the category to which the subclass belong (Rosch, 1975). Prototypes are the subconcepts which best represent a category, while exceptions are those which are considered exceptional although still belonging to the category. In other words all the sufficient conditions for class membership hold for prototypes. For example, let us consider the biological category *mammal*: a *monotreme* (a mammal who does not give birth to live young) is an example of an exception with respect to the property of giving birth to live young. Prototypes depend on the context; there is no universal prototype but there are several prototypes depending on the context, therefore a prototype for the category *mammal* could be *cat* if the context taken is that of *animals that can play the role of pets* but it is *lion* if the assumed context is *animals that can play the role of circus animals*. In the knowledge model presented above we explicitly describe the context in natural language in the *Documentation* facet, however, the context can also be described by the roles that the concept which is being described is able to play.

Ontologies typically presuppose context and this feature is a major source of difficulty when integrating them.

Prototypes are also quite important in that they provide a frame of reference for linguistic quantifiers such as *tall*, *short*, *old* and so on. These quantifiers are usually defined or at least related to the prototypical instance of the class whose members they are describing, and indeed their definition changes if we change the point of reference. For example, if we are defining the concept *tall* using as our frame of reference the class :PERSON then *tall* means over 2 metres, whereas if we define *tall* with respect to the class :BUILDING it means over 300 metres. And again, depending of the level of granularity chosen for the description the linguistic quantifiers can have more specific meanings. For example, if we subdivide the class :BUILDING into two subclasses, :COTTAGE and :SKYSCRAPER, then an adjective such as *tall* related to the prototypical instances of the two classes takes the meaning of over 10 metres in the former case and over 300 metres in the latter case.

Therefore including the notions of prototypes and exceptions permits us to provide a frame of reference for defining these qualifiers with respect to *a specific class*. For the purpose of building ontologies for multi-agent systems, distinguishing the prototypical properties from those describing exceptions increases the expressive power of the description. Such distinctions do not aim at establishing default values but rather at guaranteeing the ability to reason with incomplete or conflicting concept descriptions.

The ability to distinguish between prototypes and exceptions helps to determine which properties are necessary and sufficient conditions for concept membership. In fact a property which is prototypical and that is also inherited by all the subconcepts (that is, it has the facet :VALUE-MODALITY set to `All`) becomes a natural candidate for a necessary condition. Prototypes, therefore, permit the identification of the subconcepts that best fit the cognitive category represented by the concept in the

specific context given by the ontology. On the other hand, by describing which properties are exceptional, we provide a better description of the class membership criteria in that it permits us to determine what are the properties that, although rarely holding for that concept, are still possible properties describing the cognitive category.

Further, the information on prototypes and exceptions can prove useful in dealing with inconsistencies arising from ontology integration. When no specific information is made available about a concept and it inherits conflicting properties, then we can assume that the prototypical properties hold for it.

The inclusion of prototypes in the knowledge model provides the grounds for the semi-automatic maintenance and evolution of ontologies by applying techniques developed in other fields such as machine learning.

8 A modelling example

We are now ready to complete the example by modelling the concept *blood pressure* with the enriched knowledge model presented above. In modelling the concept of blood pressure we take into account that both the systolic and diastolic pressure can range between a minimum and a maximum value but that some values are more likely to be registered than others. Within the likely values we then distinguish the *prototypical* values, which are those registered for a healthy individual whose age is over 18, and the *exceptional* ones, which are those registered for people with pathologies such as hypertension or hypotension. The prototypical values are those considered normal, but they can change and we describe also the permitted changes and what events can trigger such changes. Prototypical pressure values usually change with age, but they can be altered depending on some specific events such as shock and haemorrhage (causing hypotension) or thrombosis and embolism (causing hypertension). Also conditions such as pregnancy can alter the normal readings.

The concept is described as a frame, and the attributes describing it are modelled as *template slots* (that is, slots which describe properties of the subclasses and the instances of the class) and template facets. We assume here that the events causing slots to change their values are described somewhere else in the ontology as concepts themselves. So if we assume that the value associated with the slot `systolic-blood-pressure` may change depending on the event *pregnancy*, then the concept `Pregnancy` should also be modelled in the ontology.

The model is presented below. We have abbreviated the terms `Regular` and `Irregular` respectively with R and I when filling the :VALUE-CHANGE-EVENT facet.

Frame `Blood-Pressure`
Template-Slot `systolic-blood-pressure`
Template-Facet :VALUE-TYPE Integer
Template-Facet :CARDINALITY 1
Template-Facet :NUMERIC-MINIMUM 0
Template-Facet :NUMERIC-MAXIMUM 300
Template-Facet :VALUE-LABEL value
Template-Facet :VALUE-PROTOTYPES [90, 130]
Template-Facet :VALUE-EXCEPTIONS [0, 89] ∪ [131,300]
Template-Facet :VALUE-MODALITY 3
Template-Facet :VALUE-CHANGE-FREQUENCY volatile
Template-Facet :VALUE-CHANGE-EVENTS ((age = 60,[0, 89] ∪ [131, 300], after), I)
Template-Facet :VALUE-CHANGE-EVENTS ((haemorrhage,[0, 89],after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((shock,[0, 89], after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((thrombosis,[131, 300],after),R)
Template-Facet :VALUE-CHANGE-EVENTS ((embolism,[131, 300],after),R)
Template-Facet :VALUE-CHANGE-EVENTS ((pregnancy,[0, 89] ∪ [131, 300],after),R

Template-Slot `diastolic-blood-pressure`
Template-Facet :VALUE-TYPE Integer
Template-Facet :CARDINALITY 1
Template-Facet :NUMERIC-MINIMUM 0
Template-Facet :NUMERIC-MAXIMUM 200
Template-Facet :VALUE-LABEL value
Template-Facet :VALUE-PROTOTYPES [60, 85]
Template-Facet :VALUE-EXCEPTIONS [0, 59] ∪ [86, 200]
Template-Facet :VALUE-MODALITY 3
Template-Facet :VALUE-CHANGE-FREQUENCY volatile
Template-Facet :VALUE-CHANGE-EVENTS ((age = 60,[0, 59] ∪ [86, 200], after), I)
Template-Facet :VALUE-CHANGE-EVENTS ((haemorrhage,[0, 59], after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((shock,[0, 59], after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((thrombosis,[86, 200], after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((embolism,[86, 200],after), R)
Template-Facet :VALUE-CHANGE-EVENTS ((pregnancy,[0, 59] ∪ [86, 200], after), R)

9 Conclusions

This article has presented an ontology model that supports knowledge-sharing in multi-agent systems where the agents can be heterogeneous. The proposed model extends the usual ontology models and has been implemented in a frame-based knowledge model inspired by OKBC. The extension concerns the explicit representation of additional information about the properties of the attributes used to describe a concept. The ontology model extension encompasses semantic information designed to characterise the behaviour of properties in agents' concept descriptions.

The novelty of this extended knowledge model is that it explicitly represents the behaviour of attributes over time by describing the changes in a property that are permitted for members of the concept. It also explicitly represents the class membership mechanism by associating with each slot a qualitative quantifier representing how properties are inherited by subconcepts. Finally, the model does not only describe the prototypical properties holding for a concept but also the exceptional properties.

We have also related the extended knowledge model to the formal ontological analysis by Welty and Guarino (2001) which permits us to build ontologies that have a cleaner taxonomic structure and so gives better prospects for maintenance and integration. Such a formal ontological analysis is usually difficult to perform and we believe our knowledge model can help knowledge engineers to determine the meta-properties holding for the concept by forcing them to make the ontological commitments explicit.

A possible drawback of this approach is the large number of additional facets that need to filled when writing the ontology to be used by the agent. We realise that this can make building an ontology from scratch even more time-consuming but we believe that the outcomes in terms of better understanding of the concept and the role it plays in a context together with the guidance in determining the meta-properties at least balances the increased complexity of the task.

References

Chalupsky, H, 2000 "Ontomorph: a translation system for symbolic knowledge" *Principles of Knowledge Representation and Reasoning. Proceedings of the Seventh International Conference* (KR'2000) 471–482; citeseer.nj.nec.com/chalupsky00ontomorph.html.

Chaudhri, V, Farquhar, A, Fikes, R, Karp, Pand Rice, J, 1998 "OKBC: a programmatic foundation for knowledge base interoperability" *Proceedings of the Fifteenth American Conference on Artificial Intelligence* (AAAI-98) 600–607.

Cocchiarella, N, 1991 "Formal ontology" in H Burkhardt and B Smith (eds) *Handbook of Metaphysics and Ontology* Philosophia Verlag.

Falasconi, S, Lanzola, G and Stefanelli, M, 1996 "Using ontologies in multi-agent systems" *Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop* (KAW) pp-pp; http://ksi.cpsc.ucalgary.ca/KAW/.

Fridman Noy, N and Musen, M, 1999, "SMART: automated support for ontology merging and alignment "*Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management* (KAW) pp-pp; http://ksi.cpsc.ucalgary.ca/KAW/.

Goldszmidt, M and Pearl, J, 1996, "Qualitative probabilisties for default reasoning, belief revision, and causal modelling" *Artificial Intelligence* **84**(1–2) 57–112.

Gómez-Pérez, A, 1998, "Knowledge sharing and reuse" in J Liebowitz (ed.) *The Handbook of Applied Expert Systems* CRC Press LLC.

Gruber, TR, 1993, "A translation approach to portable ontology specifications" *Knowledge Acquisition* **5**(2) 199–220.

Guarino, N, 1998, "Formal ontologies and information systems" *Proceedings of FOIS'98* pp-pp.

Guarino, N, Carrara, M and Giaretta, P, 1994, "An ontology of meta-level-categories" *Principles of Knowledge Reasoning: Proceedings of the Fourth International Conference* (KR94) 270–280.

Hirsch, E, 1982, *The concept of Identity* Oxford University Press.

Jennings, N, 1995, "Agent software" *Proceedings of UNICOM Seminar on Agent Software* 12–27; http://www.ecs.soton.ac.uk/ nrj/pubs.html.

Kant, I, 1965, *Critique of Pure Reason* St Martin's Press, translation by N Kemp Smith from *Kritik der reinen Vernunft*, 1787.

Karp, P, 1993, "The design space of frame knowledge representation systems" Technical Report 520.

Klein, M, 2001, "Combining and relating ontologies: an analysis of problems and solutions" *Proceedings of the IJCAI'01 Workshop on Ontologies and Information Sharing* 53–62.

Kowalski, R and Sergot, M, 1986, "A logic-based calculus of events" *New Generation Computing* **4** 67–95.

Kripke, S, 1980, *Naming and Necessity* Harvard University Press.

Lassila, O and McGuinness, D, 2001 (forthcoming), "The role of frame-based representation on the Semantic Web" *Electronic Transactions on Artificial Intelligence (ETAI) Journal: area The Semantic Web.*

Lewis, D, 1993, *Counterfactuals* Blackwell.

Lowe, E, 1989, *Kinds of Being. A Study of Individuation, Identity and the Logic of Sortal Terms* Basil Blackwell.

McGuinness, D, 2000, "Conceptual modelling for distributed ontology environments" *Proceedings of the Eighth International Conference on Conceptual Structures Logical, Linguistic, and Computational Issues* (ICCS 2000) pp-pp.

McGuinness, D, Fikes, R, Rice, J and Wilder, S, 2000, "An environment for merging and testing large ontologies" *Principles of Knowledge Representation and Reasoning. Proceedings of the seventh International Conference* (KR'2000) 483–493.

Mena, E, Illarramendi, A, Kashyap, V and Sheth, A, 2000, "OBSERVER: an approach for query processing in global information systems based on interoperation across pre-existing ontologies" *Distributed and Parallel Databases. An International Journal* **8**(2) 223–271.

Neches, R, Fikes, R, Finin, T, Gruber, T, Patil, R, Senator, T and Swartout, W, 1991, "Enabling technology for knowledge sharing" *AI Magazine* **12**(3) 36–56.

Parsons, S, Sierra, C and Jennings, N, 1998, "Agents that reason and negotiate by arguing" *Journal of Logic and Computation* **8**(3) 261–292.

Pinto, H, Gómez-Pérez, A and Martins, J, 1999, "Some issues on ontology integration" *Proceedings of the IJCAI'99 Workshop on Ontology and Problem-Solving Methods: Lessons learned and Future Trends*, Volume 18, 7.1–7.11; http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/.

Plantiga, A, 1989, *The Nature of Necessity* Clarendon Press.

Preece, A, Hui, K, Gray, A, Marti, P, Bench-Capon, T, Cui, Z and Jones, D, 2001, "KRAFT: an agent architecture for knowledge fusion" *International Journal of Cooperative Information Systems* **10**(1–2) 171–196.

Rosch, E, 1975, "Cognitive representations of semantic categories" *Journal of Experimental Psychology: General* **104** 192–233.

Rosch, E, 1999, "Reclaiming concepts" *Journal of Consciousness Studies* **6**(11–12) 61–77.

Searle, J, 1983, *Intentionality* Cambridge University Press.

Sowa, J, 1984, *Conceptual Structures: Information Processing in Mind and Machine* Addison-Wesley.

Sowa, J, 2000, *Knowledge Representation: Logical, Philosophical, and Computational Foundations* Brooks Cole Publishing Co.

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

Studer, R, Benjamins, V and Fensel, D, 1998, "Knowledge engineering, principles and methods" *Data and Knowledge Engineering* **25**(1–2) 161–197.

Sycara, K, Klusch, M, Widoff, S and Lu, J, 1999, "Dynamic service matchmaking among agents in open information environments" *SIGMOD Record* **28**(1) 47–53.

Tamma, V and Bench-Capon, T, 2001, "An enriched knowledge model for formal ontological analysis" *Proceedings of the International Conference on Formal Ontology and Information Systems* (FOIS'01) pp-pp.

Touretzky, D, 1986, *The Mathematics of Inheritance Systems* Morgan Kaufmann.

Uschold, M, Clark, P, Healy, M, Williamson, K and Woods, S, 1998, "An experiment in ontology reuse" *Proceedings of the Eleventh Workshop on Knowledge Acquisition, Modeling and Management* (KAW'98) pp-pp.

Uschold, M and Gruninger, M, 1996, "Ontologies: principles, methods, and applications" *Knowledge Engineering Review* **11**(2) 93–155.

Visser, P, Jones, D, Bench-Capon, T and Shave, M, 1998, "Assessing heterogeneity by classifying ontology mismatches" *Formal Ontology in Information Systems. Proceedings FOIS'98* 148–182.

Welty, C and Guarino, N, 2001, "Supporting ontological analysis of taxonomical relationships" *Data and Knowledge Engineering* **39**(1) 51–74.

Wiggins, D, 1967, *Identity and Spatio-Temporal Continuity* Basil Blackwell.

Wooldridge, M and Jennings, N, 1995, "Intelligent agents: theory and practice" *Knowledge Engineering Review* **10**(2) 115–152.