Research Article

Ontologies in support of activities in geographical space

WERNER KUHN

Institute for Geoinformatics, University of Münster, Robert-Koch-Str. 26-28
D-48149 Münster, Germany; e-mail: kuhn@ifgi.uni-muenster.de

Abstract. A method is proposed to derive ontologies of geographical domains from natural language texts that describe human activities. Through its textual grounding, the method addresses the issue of where to take the contents of ontologies from. Through its focus on actions afforded by domain objects, it establishes a criterion for selecting the contents. The actions are organized into a hierarchical theory of human activities in the domain. Using an analysis of the German traffic code as a case study, the paper demonstrates the informal parts of the process to derive such ontologies.

1. Introduction

GIS should support human activities. Instead, they are often designed as passive models of the world, with too little concern for the task contexts in which they will be used. While it has been established that improved computer system usability requires a focus on human activities (Laurel 1993), it is still too often assumed that geographical information will become useful to everybody ‘by nature’ if it only represents reality as faithfully as possible. Heaps of difficult to use geographical data collections and geoprocessing software have been produced based on this assumption. Today, these information resources are confronted with numerous and rapidly growing user requests—but supply and demand do not easily match to support practical user workflows. What can be done to make geographical information more supportive of human activities?

Ontologies have been promoted as a means to improve access to and sharing of existing geographical information resources (Smith and Mark 1998, Fonseca et al. 2000). They have informally been around for decades in the form of feature-attribute catalogues and are now being recast and put online as registries for repositories of geographical data (Gallagher and Carnahan 2000). However, information system ontologies, as they are understood and designed today, contain a view of the world that has less to do with human activities than with existing data holdings. These, in turn, are usually based on map contents rather than on an analysis of actual user needs. While one could argue that maps serve user needs as well, it has become obvious that their digitized derivatives do not automatically. The various technical, institutional, and economic reasons for this problem cannot be addressed here. Common to their remedies, however, is a single and simple business idea: specify
and produce to user needs. How can ontologies for geographical information be designed with this requirement in mind?

This paper postulates that, in order to make geographical information more useful and usable, ontologies should be designed with a focus on human activities in geographical space. It presents a first cut at a method to derive domain ontologies from textual descriptions of activities that are to be supported by geographical information. Set in the larger context of improving the usability of geographical information systems and services, the method:

- gives priority to human activities
- exploits texts describing such activities, and
- applies the notion of affordances to connect activities to domain objects.

This novel combination represents an innovative contribution to the collection of techniques available for producing ontologies. It takes up ideas and developments that are independently being carried forward toward:

- text-analytical methods for requirements engineering (Rolland and Proix 1992);
- affordance-based information modelling methods (Kuhn 1996, Jordan et al. 1998, Mark, 2000);
- formal theories of human spatial behaviour (Timpf et al. 1992, Golledge and Stimson, 1997).

The current paper is limited to a case study introducing the method and exploring its potential and implications. The formalization of the activities as algebras is left for a separate treatment. After providing the necessary background on the notions of ontologies, activities, and affordances (§2) as well as on the algebraic approach to formalization (§3), the notion of textual grounding for ontologies is introduced (§4). Through a case study of an ontology for car driving, a step-wise procedure to derive an ontology from a text document is exemplified with the German traffic code (§5). The properties of the resulting ontology, in particular its hierarchical layering, are discussed. The conclusions contain a review of what has been achieved and an outlook on remaining and emerging research questions.

2. Background and related work

This section provides a brief review of the notions of ontologies, activities, and affordances, which are central to the ideas presented subsequently. It declares how we interpret and apply these notions rather than giving a full account of their treatments in the literature.

2.1. Ontologies for information systems

Gruber defines an ontology as ‘an explicit specification of a conceptualization’ (Gruber 2001). In the context of information system design, such a conceptualization is usually interpreted as an intensional description rather than as one of a particular state of affairs (Guarino and Giaretta 1995). Webster’s Third New International Dictionary defines, similarly, an ontology as ‘a theory concerning the kinds of entities ... that are to be admitted to a language system’. In the context of a GIS, such a language system can be a database or interaction language, carrying information about the parts of reality represented in the system.

Note that Gruber talks about an ‘explicit specification’ and Webster’s definition contains the qualification ‘entities that are to be admitted’. Thus, ontology designers
Ontologies in support of activities

have to make conscious and explicit choices of what to admit as referents in a particular system or language, rather than passively recording ‘what’s out there’. How to make these choices is a question that has been insufficiently addressed in the information system design literature. An answer pertinent to our approach can be found in Chandrasekaran’s work:

‘How task-dependent are ontologies? What kinds of things actually exist should not depend on what we want to do with that knowledge. In that sense, ontologies cannot be task-dependent. On the other hand, exactly what aspects of reality in some domain get identified and written down in a particular ontology depends on what tasks the ontology is being built for.’ (Chandrasekaran et al. 1998)

In the same spirit, though with a different goal than Chandrasekaran, we suggest that the tasks (or activities) to be supported by a GIS should determine the entities that are admitted to its languages. The point of this paper is to demonstrate how this can be achieved in practice. No claim is made that such a procedure is necessary or even useful when designing GIS models for their own sake rather than for the support of certain tasks.

Current ontological theories for information systems are largely static, emphasizing objects with attributes and relationships over operations. Conversely, existing knowledge models emphasize problem-solving methods over domain theories (Guarino 1995). The reasons for the current object-bias in ontologies and data models for geographical and other information systems are manifold. They include

- the roots of GIS in static, map-based models of the world;
- an emphasis on attributes and relationships rather than operations in object-oriented design methods;
- the weakness of logic-based formal languages in dealing with operations and their semantics;
- a presumed priority of objects in human (spatial) cognition;
- the lack of understanding how natural language represents actions (witness the noun-bias in WordNet and its theory (Fellbaum 1999)).

Several proposals have been made to include problem-solving knowledge in ontologies, both inside (Camara et al. 2000) and outside (Mizoguchi et al. 1996, Chandrasekaran et al. 1998, Grosso et al. 2000) geographical information science. At the same time, Guarino’s work aims at including more domain knowledge into systems that support problem solving (Guarino 1995). Our work brings more domain theory to knowledge engineering, in accordance with the research programs of Guarino, Smith, Mark and others, and more problem-solving knowledge to terminological ontologies, similar to (Mizoguchi et al. 1996, Chandrasekaran et al. 1998, Grosso et al. 2000) and others. We describe how the world works for certain agents in a domain, recognizing that activities occur in the world, independently of how we model human knowledge about that world. In our view, domain theories that do not relate to activities make no sense, while activity models without domain knowledge are not useful.

Similarly to (Camara et al. 2000), we consider geographical space to be a system of entities and actions. We take an action perspective on the selection and formalization of entities to be represented in a GIS in order to support activities. Our ontologies are intended to describe or partition (Smith 2000) a domain from the perspective of an agent engaged in an activity. Camara’s action-driven ontologies
constitute a complementary approach, allowing for the construction of ontologies that describe the contents of existing data collections from an action perspective.

Unlike Camara’s action-driven ontologies, our ontologies specify entities in the world, independently of whether and how these entities are represented in an information system. Unlike Chandrasekaran’s or Mizoguchi’s task ontologies, our ontologies capture knowledge about problem solving in the world, independently of whether and how such methods are modelled in a knowledge base. Our interest lies in knowledge about the world, not about problem-solving methods or reasoning. And unlike Mizoguchi, our task models are not intended to be domain-independent.

Summing up, we consider it inadequate to model domains independently of the activities in them, or activities independently of the domains. Supporting human beings in geographic space requires ontologies that are developed paying attention to both, objects and activities. Such an approach needs some theory of how activities are structured, before connecting them to objects through affordances.

2.3. Activities
While this research does not represent a thorough application of activity theory, it has been influenced by some basic tenets of that theory (Nardi 1996). In particular, our approach to ontologies rests on the commitments in activity theory (Kaptelinin et al. 1999) that:

- interaction with the world shapes the conceptualizations of it;
- activities are directed toward objects (things or living beings);
- activities are hierarchically structured;
- activities capture some context-dependence of the meaning of information;
- tools and other artefacts mediate and evolve human activities.

Theoretically, the common philosophy between our approach and activity theory is a view of the world from the perspective of an agent interacting with it. Practically, the most important borrowings from activity theory are the idea that:

- the semantics of activities and objects are inseparable
- activities and objects are both hierarchically structured.

Activity hierarchies range from motive-driven high-level activities through goal-driven actions down to operations that constitute the actions without being goal-directed themselves (Kaptelinin et al. 1999). Each of these three levels can itself consist of several layers of complexity. For example, there may be a sub-hierarchy of actions constituting an activity, before these are further broken down into operations.

The ontology literature talks about tasks in a rather broad sense and we use the term accordingly. Without precluding different interpretations depending on context, tasks are best related to activity hierarchies at the level of actions. In an information system context, a task is a goal-directed procedure of a user in the course of a broader activity. It usually consists of lower-level operations (Kirwan and Ainsworth 1992).

However, we consciously broaden our focus beyond interactions with information systems or artefacts when designing ontologies. The conceptualization of the world and the human activities within it does not need to distinguish technical artefacts from natural objects or living beings. The notion of affordances allows for an equal treatment of interactions with all these parts of the environment.
2.3. Affordances

The things that human beings distinguish in the world depend on the actions they afford. This implies that the contents of an ontology of roads, for instance, depend on whether it should support knowledge sharing about navigation or road maintenance (Rüther et al. 2000). It would therefore seem logical to start from an analysis of domain activities when designing ontologies. One reason that such an approach is rarely taken may be that current formalisms for ontology development do not encourage or even allow for it. This issue will be addressed in §3.

A more fundamental reason, however, is the implicit or explicit equating of concepts with objects in the majority of the ontology literature. Ontologists generally agree that they are concerned with objects, relations, states, events, and processes (Guarino and Giaretta 1995, Chandrasekaran et al. 1998). In practice, however, most domain theories consist of object classes with attributes representing state and with some relations, leaving events and processes for later or separate treatment. Apart from ignoring activities in the world in the theories about it, this current practice also does not take events and processes into account in the selection of objects and relations.

Rather than trying to retrofit dynamic concepts into static theories, or to give an independent account of them, we attempt a balanced treatment of objects and actions from the start. This approach has been inspired and is supported by Gibson’s idea of affordances:

‘The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill. The verb to afford is found in the dictionary, but the noun affordance is not. I have made it up. I mean by it something that refers to both the environment and the animal in a way that no existing term does. It implies the complementarity of the animal and the environment.’ (Gibson 1986, p. 127).

The tight link that affordances create between objects and actions in an environment turns actions into first-class citizens of our ontologies. It follows the commitments of activity theory stated above and it matches the algebraic approach to formalization explained below. Based on the assumption that activities are the key to context, it also provides a theoretical underpinning to modelling context, similar to Smith’s notion of an ecological niche (Smith 1999). Finally, affordance-based theories are likely to account for prototype effects in categories, though this hypothesis needs further analysis and cannot be explored in this paper.

The notion of affordances used here is rather broad, including potentially narrower interpretations, such as Norman’s use of the term as ‘perceived affordances’ (Norman 1988, 1998). It applies loosely to all levels of an activity hierarchy, including not only operations and actions, but also socially or culturally afforded activities. For example, we consider a road to afford the activity of driving to a human being in a car (standing, in this combination, for Gibson’s ‘animal’). A more restrictive use might see the road as affording support to a car and the car as affording driving actions and operations.

3. Formalization

Having concluded that the current practice of formalizing ontologies through various applications and extensions of first-order logic is not expressive enough for
our purposes, we give a brief introduction into our algebraic approach to formalization. We focus on the general requirements and refer to specialized literature showing how modern functional languages meet them.

3.1. Requirements

‘Formally, an ontology is the statement of a logical theory’ (Gruber 2001). A logical theory defines the terms of a language by axioms. Usually, first-order logical theories without types are considered sufficient for this purpose. Most commonly known languages or frameworks for ontology modelling are based on first-order logic: KIF (Genesereth and Fikes 1992), Ontolingua (Gruber 1992), and Common-KADS (Schreiber et al. 1994). However, the need for more expressiveness has been recognized:

‘further distinctions among monadic predicate types must be defined in order to better characterize domain categories such as objects, events, amounts of matter, spatial or temporal regions’ (Guarino et al. 1994, p. 566).

One can try to draw such distinctions in first-order logic, through ‘sortal’ and other predicates. However, using the same formal device (predicates) to represent the two different and partly incongruent ideas of classification and property-assignment seems questionable. The same objection applies to extensions of first-order frameworks by second-order predicates, as for instance in Protégé (Grosso et al. 2000). Using a typed variant of first-order logic, on the other hand, does not go far enough, because it only replaces type predicates by type labels and fails to provide semantics for the types.

In our view, making the necessary distinctions for ontological engineering requires at least the following additional capabilities of formalization tools:

Types
The basic purpose of ontologies is to draw distinctions between types of objects. Clearly, a typed language serves this purpose better than one without types.

Type Classes
In order to characterize types, one needs to state properties of them; for example, the fact that some types are ordered and others are not. This requires higher-order capabilities in the form of type variables and type classes.

Function variables
Properties of individuals can be described by functions, properties of types by axioms about these functions. Referring to collections of individuals based on their properties requires operations that take these functions as arguments, such as a filter operation.

Algebraic axioms
Stating that an object affords an action (say, that a bridge affords crossing a river) is not very useful without defining what it means to cross a bridge (i.e. that one ends up on the other side of the river). Such a statement about semantics is best expressed by algebraic axioms (i.e. equations that contain functional expressions).

Formalization techniques and tools that support all these features are readily available and there is no practical reason to limit formal theories to first-order logic if and where that is found inadequate.
3.2. Tool

For our research, we have chosen the functional language Haskell (Hudak 2000), because it provides typing as well as higher-order capabilities. Also, since Haskell is a full-fledged programming language, theories expressed in it can be tested. Haskell can be seen as an ontology development and testing environment with unique advantages: it is typed, algebraic, higher-order, executable, lean, and freely available. A discussion of these and other advantages of functional languages for formal software specifications can be found in (Frank and Kuhn 1999, Gerding et al. 2000).

A typical problem with tools for ontology design is the precise definition of what it means to be a sub-concept of another concept. Typically, this meaning is defined as some kind of an IS-A and PART-OF relationship. However, both these notions are vaguely defined and have multiple meanings (Woods 1975, Winston et al. 1987). We consider it essential for domain theories to be precise about these fundamental relationships. Functional languages of the Haskell family follow a type theory that offers clear and precise semantics for IS-A relationships through the notion of type classes. They also have the expressive power to give precise semantics to each PART-OF relation through constructor operations and their axioms.

4. Grounding

So far, we have only partially answered the question of how to choose the concepts to be included in a domain ontology. Selecting the concepts based on activities does not say how to find out what activities are relevant to the users in a particular domain. We refer to this issue as the grounding of ontologies: the claims of any domain theory need to be based on some observations in that domain. For our purposes, we are looking for observations of activities in the form of natural language descriptions.

4.1. Communicating domain semantics

Grounding an ontology in some tangible document or artefact outside the imagination of a knowledge engineer (or even outside that of a domain specialist or standardization committee) is clearly useful, though rarely achieved. A recent survey of ontology design methods (Fernandez Lopez 1999) has shown that these methods have one thing in common: they use brainstorming or undisclosed techniques to arrive at rather improvised collections of concepts in a domain.

In practice, ontology design is done by knowledge engineers who are typically not domain specialists. They have to use their own understanding of a domain as a basis for formalization, assisted by an informal and ad hoc process of interviews with domain experts, brainstorming, analyses of requirements documents or induction from existing databases. Thus, they face the well-known knowledge extraction problem of communicating domain facts and rules to knowledge engineers. Dealing with ontologies rather than epistemologies (i.e. with theories about domains rather than about the knowledge people have of them) does not make this problem any easier.

Furthermore, the resulting ontologies are often difficult for domain experts to verify. Whether they consist of expressions in first-order logic or diagrams in something like the Unified Modeling Language (UML) (Cranefield and Purvis 1999), typical domain specialists will find it hard to detect even basic misrepresentations or omissions. Teaching logic or software engineering tools to domain experts, on the other hand, is impractical and would not help, as these languages are notoriously
weak, even for knowledge engineers, in ascertaining the consistency and completeness of formalizations. Thus, one also faces the problem of validating ontology designs, both by the designers and the domain specialists.

For these reasons, we have decided to separate the two concerns of formalization and communication, concluding that they are practically incompatible today. The previous section of the paper discussed our approach to formalization. This section introduces an approach to the communication of domain requirements that is sometimes used in software engineering (Rolland and Proix 1992) and in the form of protocol analysis for human-computer interface design (Kirwan and Ainsworth 1992): analysing natural language descriptions of domain activities.

4.2. Natural language texts

The requirements for languages that can communicate domain requirements are tough to meet. Practical experiences in ontological design projects have taught us that:

- domain experts should already be familiar with such a language
- domain knowledge should already be expressed in the language
- translating from the language to a domain theory should be supported by tools.

If these requirements are all to be satisfied, the choice of languages reduces to, at best, one candidate: natural language is the communication device with which all domain experts are familiar, in which most domain knowledge is already expressed, and for which the broadest range of processing tools exists. Consequently, we ground our domain ontologies in natural language descriptions of domain activities (Kuhn 2000).

Using informal texts as a basis for formal ontologies raises concerns about ambiguity of natural language, as well as inconsistencies and incompleteness in textual descriptions. These concerns are non-trivial, but pertain to the general problem of moving at some point from the informal, imprecise statements in the real world to formal theories and models for information systems. Ambiguous descriptions for which domain experts feel competent seem preferable to (possibly!) unambiguous models, which can coerce them into accepting something that they might not fully understand or agree with. If there is ambiguity in the texts (and there usually is), it is likely to be revealed in deriving and executing a formal specification. If there is ambiguity in non-executable ontologies, it will be revealed much later (and at high costs) in system implementations. On the other hand, if today’s formal ontologies are unambiguous, they are likely to have lost crucial application semantics (for the reasons given above) and may thus lead to systems that are of limited use.

What happens if no natural language texts are available for a domain? Actually, for most GIS application domains, some kinds of work regulations or workflow descriptions are usually available. In cases where no texts can be found, the best procedure is to have such texts written first by domain experts or knowledge engineers, explaining the relevant activities in clear and simple terms, but exhaustively and in detail. Often, this phase is part of a requirements analysis anyway and leads to useful clarifications in the process of an information system design.

5. Case study

We demonstrate the current status of the method through a case study taken from the domain of car navigation: analysing the text of the German traffic code.
After explaining the reasons for this choice, a step-wise procedure for deriving formal ontologies from such a text is presented. Finally, a hierarchical model of the domain activities is developed. Formalization is left for a separate presentation.

5.1. Domain and text selection

While one might not expect the domain of car navigation to be easily amenable to a method starting from natural language descriptions of actions, there are actually several such descriptions available or easily obtainable:

- traffic codes describe all objects and actions relevant to legal driving
- driving instructions contain information for successful navigation
- travel narratives provide an account of observations and decisions during navigation.

We have chosen the German traffic code (StVO 2000) as the text to apply the method to and develop it. One reason was that such a code defines by definition all actions and constraints that are relevant to driving behaviour, at least from the legal point of view. Such codified descriptions are actually available for many domains where spatial information is being used, due to the legal or administrative regulations coming with many spatial activities.

In fact, any agency using spatial information is bound to have detailed descriptions about the operations implementing its mandate. In another (unpublished) case study, we have analysed workflow descriptions of the German Agency for Waterways (BAW). The goal was to gather information for the design of the agency’s feature-attribute catalogue. The agency has eventually decided to adopt the conventional bottom-up design procedure, identifying objects from existing data holdings. However, the study has revealed that such workflow descriptions constitute a surprisingly rich and precise inventory of what their operations are about.

Clearly, a legal code represents a special case of a natural language text with properties that other texts lack. Among these are completeness, consistency, and minimized ambiguity. We are currently collecting experience on the suitability of other, less-structured texts such as narratives, in the context of an ontology design project for location-based mobile services. It appears that even the possibility to obtain an incomplete ontology from unstructured texts represents a significant achievement in any domain modelling process.

Without loss of generality for the method, the rules in the traffic code have been simplified according to the following restrictions:

- Traffic consists of regular cars only that move forward and have no weight;
- there are no special road and traffic conditions (weather, slow traffic etc.);
- ill-defined conditions are omitted (those typically requiring court interpretation);
- there are no traffic jams and no obstacles on the road;
- there are only roads (with their parts, like lanes) and private or public properties;
- there are no highways, side lanes, paths, or narrow roads;
- there is no ‘zipping’ rule (merging of traffic from two roads);
- not endangering others is (conservatively) interpreted as not hindering them;
- not hindering is interpreted as keeping a (minimal) distance.

For the sake of understandability, the extracted terms have been translated into
English. For the actual analysis, however, it has proved important to stick to the language of the original text.

5.2. Ontologies from texts

After selecting (or producing) an appropriate text describing relevant domain activities, a sequence of text-analytical steps is used to derive a cross-tabulation of actions and objects in the domain. The basic idea is simply to extract verbs and nouns from the text, standing, respectively, for actions and objects. The actual procedure is more involved than this, because of the need to deal with various semantically equivalent grammatical or lexical forms (e.g. the expression of an action by nouns like gerunds). Of course, the details of this procedure are also language-dependent. The following description abstracts from the specifics of German and English to a common procedure that may need adaptation for other languages.

5.2.1. Looking for action

As a first step, the chosen text is scanned for all expressions denoting something being done in the domain. Thus, we are looking for all activities, actions and operations (in the sense of activity theory), without yet distinguishing these levels, as well as states, using the generic term action for all of them. The kinds of linguistic expressions sought are:

- verbs and verbal expressions, e.g. ‘drive’ or ‘keep distance’
- gerunds and other nouns, e.g. ‘driving’ or ‘speed reduction’

The collection of action terms is simplified by merging synonymous expressions, in particular:

- synonyms, e.g. ‘reduce speed’ and ‘break’
- different grammatical forms, e.g. ‘reduce speed’ and ‘speed reduction’

We choose canonical terms for these synonymous expressions and obtain as an intermediate result a list of actions in the domain. For the case of the traffic code, this list (translated to English and alphabetically ordered, but with the German original terms included) looks as follows:

Table 1. Alphabetical list of actions in (simplified) German traffic code.

<table>
<thead>
<tr>
<th>German Term</th>
<th>English Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>be passed</td>
<td>überholt werden</td>
</tr>
<tr>
<td>change lane</td>
<td>Fahrstreifen wechseln</td>
</tr>
<tr>
<td>control speed</td>
<td>Geschwindigkeit anpassen</td>
</tr>
<tr>
<td>drive</td>
<td>fahren</td>
</tr>
<tr>
<td>enter from curb or property</td>
<td>vom Fahrbahnrand anfahren, aus Grundstück einfahren</td>
</tr>
<tr>
<td>keep a distance</td>
<td>Abstand halten</td>
</tr>
<tr>
<td>stand on a property</td>
<td>auf Grundstück stehen</td>
</tr>
<tr>
<td>stop</td>
<td>halten</td>
</tr>
<tr>
<td>swerve-pass-merge</td>
<td>ausschernen-überholen-wiedereinordnen</td>
</tr>
<tr>
<td>turn</td>
<td>abbiegen</td>
</tr>
<tr>
<td>turn around</td>
<td>wenden</td>
</tr>
<tr>
<td>view a road section</td>
<td>Strecke übersehen</td>
</tr>
<tr>
<td>wait-continue</td>
<td>warten-weiterfahren</td>
</tr>
<tr>
<td>yield</td>
<td>Vorfahrt gewähren</td>
</tr>
</tbody>
</table>
A surprising observation about this collection is its brevity. After the indicated, relatively minor simplifications of the entire text (compared to the overall number of actions in the text), one ends up with a total of just 14 regulated action types.

5.2.2. Affording it

The next step identifies the objects that are affording the actions. In the traffic code example, after eliminating all other traffic participants, car drivers are the only subjects who are afforded any actions. Distinguishing affordances for additional actors is therefore not necessary here. It would extend the procedure but not change it.

The kinds of expressions sought in this step are:

- direct objects, e.g. ‘(change) the lane’
- indirect objects, e.g. ‘(yield to) oncoming car’

After these objects have been identified, they are again reduced to a set of canonical expressions, representing all object classes participating in driving actions. This list, again, turns out to be rather short:

<table>
<thead>
<tr>
<th>Table 2. List of affording objects in (simplified) German traffic code.</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncoming car</td>
</tr>
<tr>
<td>following car</td>
</tr>
<tr>
<td>preceding car</td>
</tr>
<tr>
<td>passed/passing car</td>
</tr>
<tr>
<td>other car</td>
</tr>
<tr>
<td>road</td>
</tr>
<tr>
<td>road section</td>
</tr>
<tr>
<td>roadway</td>
</tr>
<tr>
<td>lane</td>
</tr>
<tr>
<td>curb</td>
</tr>
<tr>
<td>property</td>
</tr>
<tr>
<td>intersection</td>
</tr>
<tr>
<td>junction</td>
</tr>
</tbody>
</table>

The distinction of several ‘kinds’ of cars (oncoming, following, etc.) would seem questionable. After all, they are all just cars. In the context of particular driving situations, however, they play certain roles that need to be distinguished.

As indicated in the previous section, we apply the notion of affordances loosely. Some of the listed objects are more literally affording an action than others are. For example, a property can certainly have the affordance for a car to stand on it, or another road the affordance to turn into it. The length of a road section is even defined, in one of the code’s rules, by how far it affords the driver to see ahead. On the other hand, stopping within that distance is not strictly afforded by the road section; the section ‘affords’ stopping within its extent only by virtue of the driver obeying that rule.

5.2.3. Constraining

Standard ontologies put great emphasis on the attributes or properties characterizing objects. Indeed, attributes are often implicitly or explicitly used as criteria for category membership. For instance, an object may count as a road if and only if it has a start- and end-point, a width, a number of lanes, and a pavement. Given the well-documented difficulties with such feature-based categorizations (Lakoff 1988),
we chose the algebraic style of categorizing objects according to the actions they afford. Having done that, however, both objects and actions can now be further described by their properties and the constraints on them.

In a regulative document like the one we are analysing, attributes are typically the subject of constraints. For example, the speed of driving or the right of way are being regulated. Thus, we obtain constraints on properties of objects or actions. For example, lanes and driving both have a direction attribute, which must have matching values for a particular instantiation of driving. To keep the method presentation short, these constraints are not further discussed in this paper.

5.2.4. Cross-tabulating

The result of the process so far can be represented as a simple cross-tabulation of actions and objects (table 3). The entire text-analytical process so far could in principle be automated, but our understanding of it is still far from allowing for that. It has also proven difficult to make effective use of text-analytical tools. Obvious tool candidates are so-called taggers, i.e. programs that label words in texts with their grammatical roles, the freely available Morphy (Lezius 2000) being a good example for German. Experience so far suggests that a manual extraction and classification of all expressions referring to actions and objects, though quite tedious, is the most effective procedure.

5.3. Action hierarchies

The next analytical step defines an ordering. The relation (table) that has so far been established between actions and objects in the domain is reminiscent of the relation between attributes and objects called a context in formal concept analysis (Wille 1992). The difference is that actions take the place of attributes in our case. Whether they determine objects with the same rigor as attributes do is a deep question going far beyond the scope of this work. Nevertheless, the intent of a concept, in our case, is defined by actions rather than attributes. The notions from formal concept analysis (formal contexts and concept lattices) are being applied to ontologies for geographical information by (Kokla and Kavouras 2000).

In formal concept analysis, an ordering is achieved through the subconcept-superconcept relation (Wille 1992). For example, the driving, passing, and breakdown lanes would be subconcepts of the superconcept lane. However, concept lattices are, in most practical applications, equating concepts with objects. We are interested in finding an ordering of actions instead. Actions have inherent semantics of their own, which allow an ordering that goes beyond the subconcept-superconcept relation for objects.

In search of such an action ordering, we are applying the notion of lexical entailment from recent theories of verbs in natural language (Fellbaum 1998). A verb v1 entails a verb v2, if the statement ‘someone v2’ follows logically from ‘someone v1’. For example, snoring entails sleeping, because if someone snores, he sleeps. Linguists have defined four kinds of entailments (and applied them, for instance to the WordNet database model (Fellbaum 1999)):

- Troponymy: v1 is v2 in some manner (e.g. speeding entails driving)—this corresponds to the subconcept–superconcept relation for objects;
Table 3. Cross-tabulation of actions and objects in (simplified) German traffic code.

<table>
<thead>
<tr>
<th>Actions</th>
<th>Oncoming car</th>
<th>Following car</th>
<th>Preceding car</th>
<th>Passing car</th>
<th>Other car</th>
<th>Road</th>
<th>Road section</th>
<th>Roadway</th>
<th>Lane</th>
<th>Curb</th>
<th>Property</th>
<th>Intersection/junction</th>
</tr>
</thead>
<tbody>
<tr>
<td>be passed</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>change lane</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>control speed</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drive</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enter</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>keep distance</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stand</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stop</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>swerve-pass-merge</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>turn</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>turn around</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>view</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>wait-continue</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>yield</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Ontologies in support of activities
Proper inclusion: v1 and v2 occur together, but not for the entire duration (e.g. passing entails swerving)—this corresponds to the part-whole relationship for objects, but the inclusion can occur in either direction;

(Backward) presupposition: v2 precedes v1 (e.g. driving on a road entails entering it);

Causation: v1 causes v2 (e.g. being passed causes one to control the speed).

These notions of entailment provide the required ordering on actions: We say that action a2 is based on action a1, if the corresponding verb v2 lexically entails v1. For example, the action of speeding is based on the action of driving. With this ordering, we are able to build an action hierarchy, which represents a layered model of increasingly complex behaviour.

A useful constraint and guideline for deriving layers of actions was to keep the ontology incrementally consistent: the specified action at each level has to obey all the rules that are applicable up to this level. For example, if cars can only stand still, they are not permitted on lanes.

The cross-tabulation of all actions in our case study (table 4) shows their entailments, with T standing for troponymy, I for (proper) inclusion, P for (backward) presupposition, and C for causation. A lattice-like structure corresponding to this matrix has been drawn up in figure 1.

Thus, we have achieved a layering of the actions in the domain of car driving according to the German traffic code. Its formalization and validation through algebraic types and type classes in a functional language is left for another paper. The implementation proceeds top-down in the hierarchy, which is bottom-up in terms of complexity. The diagram of figure 1 shows indeed the structure of the inheritance hierarchy for functions among the necessary data types. As such, it serves as a basis to derive a hierarchy of objects affording actions.

6. Conclusions

Ontologies have become the subject of engineering processes (Guarino 1998). They are commonly defined as specifications of concepts that occur in a domain. Like any other specification, they require methods to produce and validate them. We presented such a method that obtains a textual grounding for ontological engineering and offers a way to design ontologies in support of activities. It identifies actions from textual domain descriptions and subsequently identifies the object classes that afford these actions.

Summarized and somewhat simplified, the steps of the method are to:

1. Select or produce a natural language text describing activities in a domain.
2. Extract actions from the verbs found in the text.
3. Identify object classes affording these actions from the nouns.
4. Order actions according to entailment relations among the verbs.
5. Produce a hierarchical theory of the domain in the form of an action hierarchy.

The method is still under development, as part of research efforts toward broader practical and theoretical goals:

- replacing traditional feature-attribute catalogues for geographical information by formal, machine-readable ontologies;
- using the notion of affordances to capture how geographical objects support human actions;
Table 4. Cross-tabulation of actions showing entailment relations.

<table>
<thead>
<tr>
<th>Actions</th>
<th>Be passed</th>
<th>Change lane</th>
<th>Control speed</th>
<th>Drive</th>
<th>Enter</th>
<th>Keep distance</th>
<th>Stand</th>
<th>Stop</th>
<th>Swerve-pass-merge</th>
<th>Turn</th>
<th>Turn around</th>
<th>View section</th>
<th>Wait-continue</th>
<th>Yield</th>
</tr>
</thead>
<tbody>
<tr>
<td>be passed</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>change lane</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>control speed</td>
<td></td>
<td></td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>drive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>keep distance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stop</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>swerve-pass-merge</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>turn</td>
<td></td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>turn around</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>view section</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wait-continue</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>yield</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
modelling the contextual component in geographical information through activities;
• complementing first-order logic as a formalization device by universal algebra;
• prototyping ontologies in object-oriented languages that treat functions as first-class citizens.

The proposed approach is meant to complement rather than replace existing ones. The characteristics by which the method differs from others are:

• its grounding in texts
• its activity-oriented approach
• the layering of the resulting ontologies.

The specification of concepts at multiple layers of increasing complexity is a key requirement of ontological design. It allows for a link to artificial intelligence approaches for modelling complexity (Axelrod 1997). Our domain theories are based on the assumption that the increasing complexity of human conceptualizations of the environment results from (i.e. responds to) increasingly complex activities, rather than the other way round. While a justification of this assumption lies outside the scope of this work, we believe that such a paradigm makes ontological theories not only more supportive of human activities, but also more concise.

Eventually, our ontologies will consist of complete executable specifications in a functional language of activities and object classes. The key advantage of such a rich and executable model is that an ontology can demonstrate itself to its designer as well as to a domain expert. Most inconsistencies and omissions can then either be avoided by construction during the design of the ontology or they lead to observable errors during execution (Frank and Kuhn 1999).
Combining the layering of activities with executability will lead beyond a typical formal ontology (in the sense of a collection of theorems and axioms). A goal of our case study is indeed to produce an agent-based simulation of legal driving behaviour in Germany. One could argue that a simulation of domain processes is something else than an ontology establishing the theory—and indeed it is. However, if a domain theory is being developed without being checked through such a simulation, it needs alternative means of validation.

Acknowledgments

The ideas for this work took shape during a study for the German Agency for Waterways (BAW) and a project with MobileGIS Ltd. Many colleagues have provided valuable comments and suggestions, among them Andrew Frank, Karen Kemp, David Mark, Christoph Rüther, Barry Smith, Steve Smyth, and Sabine Timpf and the reviewers. Gwendolyn Raubal helped with English translations of the German traffic code vocabulary. Support from the European Science Foundation and from the Hessian Surveying and Mapping Agency is gratefully acknowledged.

References


KUHN, W., 1996. Formalizing spatial affordances, formal models of commonsense worlds (Specialist Meeting, NCGIA Research Initiative 21). National Center for Geographic Information and Analysis (NCGIA), San Marcos, TX. http://www.geog.buffalo.edu/nccgia/i21/papers/kuhn.html


