

Giving Meaning to GI Web Service Descriptions

(Extended Abstract¹)

Florian Probst and Michael Lutz
Institute for Geoinformatics (ifgi)
University of Münster, Germany
{f.probst|m.lutz}@uni-muenster.de

SUMMARY

Current GI service composition based on syntactic descriptions such as WSDL are error-prone because the meaning of the labels used in these descriptions is unclear. We identify three types of problems that can result from such semantically heterogeneous descriptions during service composition. These problems call for a Semantic Reference System for semantically annotating symbols, referencing symbols to concepts and semantically grounding these concepts. We present a three level architecture for such a Semantic Reference System and illustrate how it can be used for solving the problems identified using a real world example for service composition.

KEYWORDS: *GI web services, Semantic Reference System, GI service composition*

MOTIVATION / PROBLEM STATEMENT

Composability is often seen as one of the main strengths of web services. In the geospatial domain, after focusing mainly on services providing data and maps, currently the first (geo) processing services are emerging.

To enable meaningful composability of web services not only syntactic descriptions such as WSDL (Chinnici, Moreau *et al.* 2003) are needed. A further step has to be taken in providing the user with descriptions telling him what the labels of data types and operations used in a syntactic service description actually *mean*. This calls for a Semantic Reference System (Kuhn 2003) allowing for semantic annotation of symbols, referencing of symbols to concepts and semantic grounding of concepts with image schemata (figure 1)

In this paper we introduce the architecture for a three level Semantic Reference System consisting of application ontology, domain ontology and semantic grounding levels. We focus on data types that are used as operation input and output, rather than on the functionality of the operation. We illustrate the conceptual results with a “real world” example supporting the following hypothesis: By evaluating references of service descriptions to a Semantic Reference System meaningful interoperability between services can be ensured during service discovery.

¹ The full paper will be published in the Proceedings of the *Second International Workshop on Web Services: Modeling, Architecture and Infrastructure*, held in conjunction with ICEIS 2004.

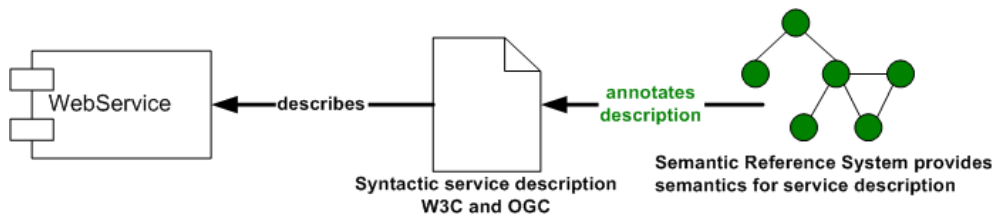


Figure 1: A Semantic Reference System provides (grounded) semantics for standard service descriptions

The following short scenario² is used to explain in which context we encountered and solved semantic problems: A service provider is about to build a composite service for the management of accidents involving toxic gas releases from a chemical plant. He builds the composite service starting with the most specialised service³ and subsequently adding further services until the composite service meets the user's requirements. The user already has the central, most specialized service of the composite service available. This is a service for calculating a toxic plume (afterwards called *CalculateGasDispersionService*). The first step is to check the input and output of the plume service. It requires information about the wind speed, the wind direction, the location of the gas emission and the emission rate as input. The output is a polygon indicating the dispersion of the gas. The user chooses as next step to find an additional service providing information on the wind direction and therefore searches for services providing weather information in a UDDI registry (Bellwood, Clément *et al.* 2003). He discovers two services: the *GlobalWeatherService* and the *AirportWeatherService* provided by CapeScience (<http://www.capescience.com>). The user now has to determine whether these services actually match (syntactically and semantically) the requirements of the *CalculateGasDispersionService*. The semantic problems he encounters are explained in the following chapter.

TYPES OF SEMANTIC PROBLEMS

Currently service discovery relies on the labels of input and output descriptions and on the labels of data types given in WSDL (or similar service metadata) descriptions. It is generally assumed that if the labels are the same, the transported information is, too. However, this is not always the case. Different types of semantic heterogeneity have been identified for GIS (Bishr 1998) and GI web services in general (Lutz, Riedemann *et al.* 2003). In the following, we present three types of heterogeneity problems that play a role during GI web service composition. Each problem type is illustrated by relating it to the scenario given above (figure 2).

Problem Type I (Naming Heterogeneity)

*The output of a web service and the input of a second web service are represented with the same data type and refer to the same domain concept, but have **different** labels (names).*

The *AirportWeatherService* and the *GlobalWeatherService* both provide information about the wind direction. However, the labels of the data types containing the required information are different. The *GlobalWeatherService* refers to the information as `prevailing_direction` while the *Airport-*

² This scenario was developed in conjunction with the ACE-GIS e-Emergency composite service (<http://www.acegis.net/>).

³ To keep things simple, we assume that each web service has only one operation. We therefore use the terms web service and operation interchangeably.

WeatherService labels the information as `wind`. However, both elements represent the same (domain) concept of *wind direction*.

It can be solved in two steps. First, each of the elements used in the WSDL descriptions are referenced to the same domain concept. Second, the reference includes additional restrictions on the properties of the domain concept, constraining its meaning. If reference and restriction are identical, the meaning of the used symbol is the same.

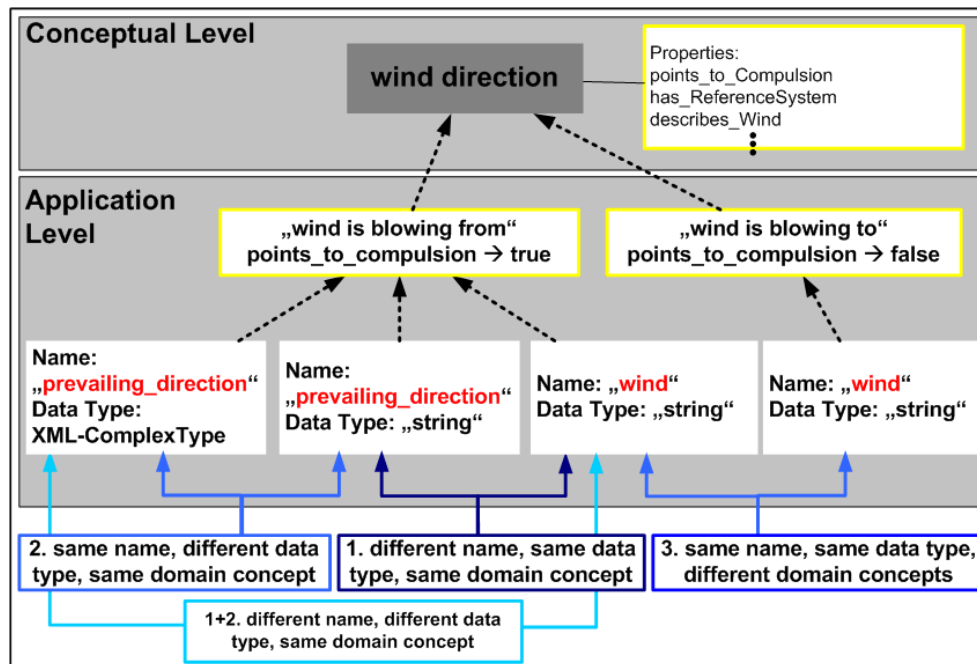


Figure 1: Types of semantic problems. Restrictions on a domain concept change its meaning.

Problem Type II (Data Type Heterogeneity)

The output of a web service and the input of a second web service have the same labels (names) and refer to the same domain concept, but are represented with **different** data types.

The example given above is further complicated by a second source of heterogeneity. The *GlobalWeatherService* provides the wind direction information represented as a complex type labelled `Direction`. The *AirportWeatherService* provides this information contained in a `String`. However, both elements represent the same domain concept⁴.

This problem is not simply to be considered syntactical heterogeneity since the meaning of the information contained in a complex type is not explicit to the user. When dealing with complex data

⁴ Note that in the example which is taken from “real” services, problem types I and II occur simultaneously. For clarification, the example problem is split into two problems types.

types, a semantic description of the data type can help in applying a suitable parser to transform the data types. In an intermediate step appropriate parsers can be offered to transform the information into the required data type of the preceding service.

Problem Type III (Conceptual Heterogeneity)

*The output of a web service and the input of a second web service have the same labels (names), are represented with the same data type, but refer to **different** domain concepts.*

Consider now the wind information represented as a `String` provided by the *AirportWeatherService* as described above. And consider further that the *CalculateGasDispersionService* also requires wind information represented as a `String`. However, the *CalculateGasDispersionService* interprets the provided `String` not as degrees characterising the direction the wind is blowing from. Instead it interprets the `String` as characterizing the direction the wind is blowing to. This misinterpretation introduces a 180-degree mismatch.

This can lead to the creation of composite services that produce results not intended by the user. This is due to the fact that currently most composite services are built manually using WSDL-based service descriptions. If inputs and outputs of operations have the same name and data type, the user cannot tell that these operations refer to different domain concepts. The underlying assumption causing this problem is that if something is described identically, it must have the same meaning.

By referencing the application level concepts to different domain concepts, or by using different restrictions on the same domain concepts, it can be prevented that services whose inputs and outputs do not match (on the conceptual level) are combined in a service chain.

SEMANTIC REFERENCE SYSTEM

This chapter gives a brief introduction of the three levels of the Semantic Reference System and how they are related to each other. The application and domain levels consist of ontologies, i.e. “explicit specifications of a conceptualization” (Gruber 1993). The necessity of introducing three levels is explained in the following section, which relates the user’s tasks during service composition to the semantic problems identified in the previous section.

Application Level

On this level the labels used in the WSDL-based description of the service are captured in an application ontology. This is done by classifying them in a subsumption hierarchy and by relating them to each other with non-taxonomic relations. The application ontology serves to capture the service’s “world view”. It presents only how the service models the information it is providing. The labels or symbols used in this application ontology e.g. *prevailing_wind* are referenced to a domain ontology concept e.g. *wind direction*. It is important to note that these references to the domain ontology do not necessarily represent inheritance relationships as could be inferred from (Guarino 1998 figure 4). Additionally, the reference can put restrictions on the properties of the domain concept thus constraining the meaning of the domain ontology concept. An application ontology concept has only one distinct meaning. This is illustrated by the fact that in a domain ontology a property range can be of data type Boolean. The same property on Application Ontology Level would have to be specified by choosing one of the Boolean values, not just specifying the data type.

Conceptual Level

Domain ontologies describe a certain part of the world (a domain) from a certain perspective. For example, a domain ontology for meteorology contains the relevant concepts for explaining meteorology. Such an ontology will never be complete since the meteorological knowledge evolves and the terms used change. But it will serve to represent the vocabulary humans use to communicate about meteorology. The concepts used in this ontology will be explained with respect to meteorology al-

though these concepts may have many more meanings in different contexts. The Domain Ontology Level is the level of human concepts. Here no data types or data models are described. Via the references and restriction posed from the application ontology, the domain ontology supplies meaning to the symbols used in the application ontology.

Semantic Grounding Level

The concepts described in the domain ontology need further grounding. A domain ontology relates concepts to each other, and thus restricts their interpretation. However, it cannot be assumed that every human user has the same understanding of a certain concept e.g. *wind direction*. To escape the vicious circle of defining concepts with other concepts on an ever higher level of abstraction a Semantic Grounding Level is required. We propose image schemata to semantically ground the concepts used on the Domain Ontology Level. Image schemas fall between abstract propositional structures (such as predicates) and concrete images (such as the spatial mental images in (Barkowsky 2001)). They are developed through bodily experiences and influence our reasoning through the recurrence of form and function. They stand in a long tradition of representing elements of knowledge into patterns or *schemas*. An image schema can be seen as a generic and abstract structure that helps people establish a connection between different experiences that have this same recurring structure. Therefore, meaning involves image-schematic structures (Johnson 1987; Gärdenfors 2000).

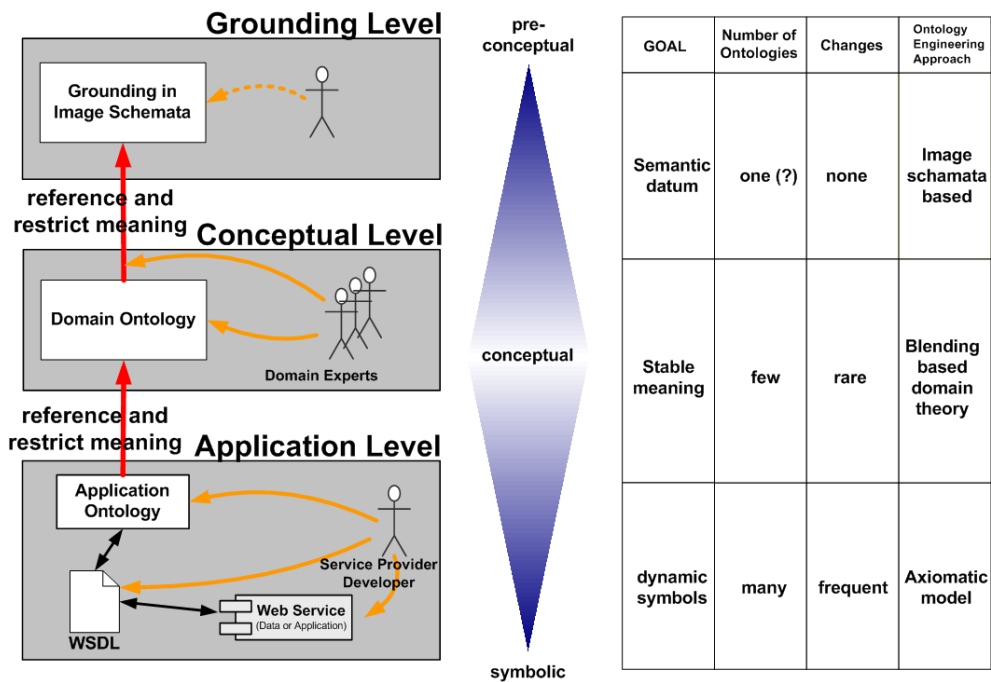


Figure 2: Three level Semantic Reference System; left: the levels meet different requirements

SEMANTIC PROBLEM TYPES RELATED TO USER TASKS DURING SERVICE COMPOSITION

The types of semantic problems identified in the previous section will now be related to the user's tasks during service composition in order to explain the need of a three level semantic reference system. Each of the levels and the references between them will be illustrated by providing formal definitions of concepts for the example scenario. For enhanced readability these concepts are stored in *one* ontology (rather than three) and marked with a prefix according to their level (AO for the application, DO for the domain and SG for semantic grounding level). The ontology language employed is OWL (McGuinness and Van Harmelen 2003). To build the ontologies we employed the ontology editor Protégé 2.0 beta (Noy, Sintek *et al.* 2001) in combination with an plug-in supporting OWL.

1. *Defining the concept of "wind direction"*. As starting point of service discovery, the user needs the possibility to specify his concept of wind direction. This is performed with querying a domain ontology (for meteorology). In the example, he will find the concept wind direction with all properties relevant in the domain of meteorology. These are, for example, that wind direction is measured in degrees or in compass directions and that the wind direction points to the source of a compulsion or away from it⁵. Compulsion is an image (Johnson 1987) schema and therefore the domain ontology references this property of wind direction to the grounding level. The possibility of referencing domain ontology concepts to image schemata decreases the likelihood of ambiguously interpreted domain ontology concepts. Figure 3 shows an example for such a definition of the domain ontology concept *wind direction* that uses other domain as well as grounding level concepts (SG_ToDirection and SG_FromDirection).

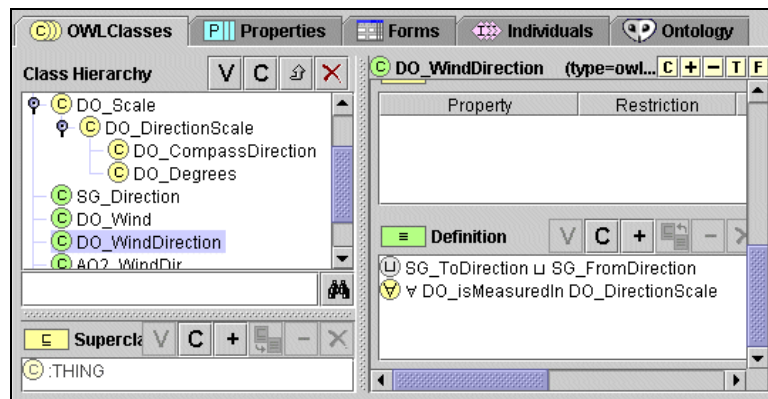


Figure 3: Definition of the domain ontology concept *wind direction* (DO_WindDirection) in the ontology editor Protégé using the OWL plugin.

The set of image schemata, their internal structure and how they can serve as semantic grounding level need further investigation. However, the possibility to break the vicious circle of defining concepts with other (undefined) concepts is appealing.

2. *Finding services dealing with the identified concept.* With the chosen concept "wind direction" the user discovers all three services in a application ontology registry since their labels *wind*,

⁵ For simplicity reasons, this difference is currently modelled using two concepts on the semantic grounding level, namely SG_ToDirection and SG_FromDirection. These will be replaced by image-schematic concepts in a future version.

wind and *prevailing_direction* refer to application ontology concepts (AO1_WindDir and AO2_WindDir in Figure 4) that are defined using this domain ontology concept. However, the restrictions the application ontologies put on their references reveal that the two concepts are different because they represent the direction *in which* respectively *from which* the wind is blowing.

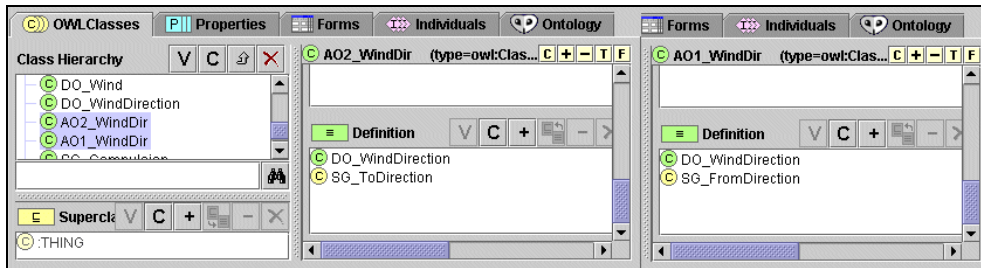


Figure 4: Definition of two application ontology concepts representing *wind direction* (AO1_WindDir and AO2_WindDir).

Discovering whether two concept definitions are equivalent or similar and whether concepts are related in a subsumption hierarchy is easy in the simple example we use for illustration. However, when using more complex definitions, it becomes much more difficult. In such cases a reasoning engine such as RACER (Haarslev and Möller 2003) can be used to compute a subsumption hierarchy and to identify equivalent concepts. The computed subsumption hierarchy for our example is shown in Figure 5.

With the currently existing WSDL descriptions the user has to judge whether the application concepts differ in such a way that chaining the services will produce wrong results. In this example, chaining the *AirportWeatherService* or the *GlobalWeatherService* to the *CalculateGasDispersionService* both would result in a 180-degree mismatch.

The domain ontology concept plus the restrictions on its interpretation provide *meaning* to the labels used in the WSDL service descriptions. Here it becomes obvious why domain ontology and application ontology need to form separate levels. The domain ontology provides a stable source of broadly defined (human) concepts. The application ontology provides the service developer with a flexible means to restrict or constrain the meaning of the domain ontology concepts. Finding services using the same restrictions on domain concepts solves naming problems (problem type I). The possibility for the user to be aware of different restrictions solves conceptual problems (problem type III).

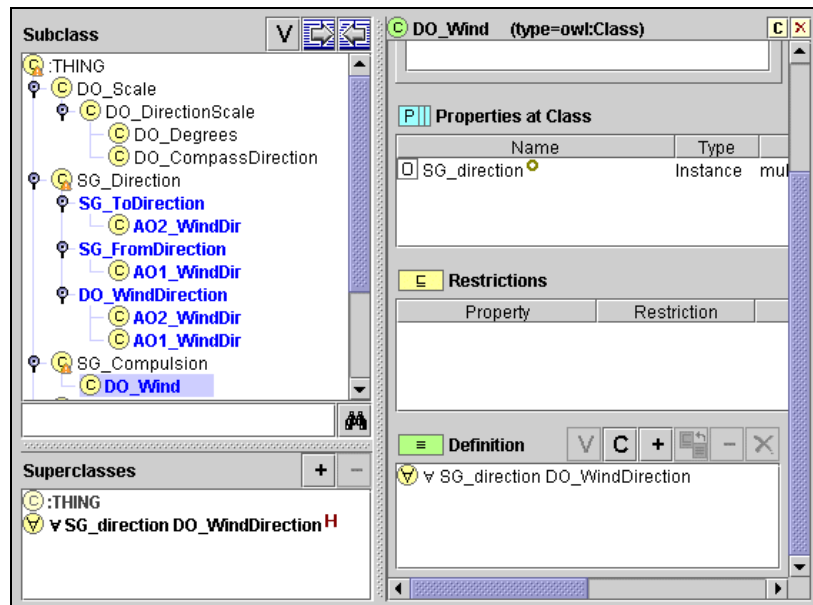


Figure 5: Subsumption hierarchy computed by RACER.

- Learn how the service represents the needed information. Consider the user searches for services dealing with wind direction that refers to the `SG_ToDirection` concept. This results in finding the *AirportWeatherService* and the *GlobalWeatherService*. Now the user needs to know which data types are used to represent the information required. For this purpose the application ontologies of the services are queried. The resulting human-readable description of the data types employed in the service can be used to deal with problems of type II.

CONCLUSION AND FUTURE WORK

We have identified three types of semantic problems that may occur during web service composition: Naming heterogeneity, conceptual heterogeneity and data type heterogeneity. We have illustrated these types by relating them to a real world example based on services employed in a composite service for the management of accidents involving toxic gas releases from a chemical plant. Meaningful composability of web services needs semantic interoperability between web services. We claim that semantic interoperability needs a sound theory of semantic grounding.

Therefore we have introduced a three level Semantic Reference System for tackling the identified problem types. It allows the user to

- discover appropriate services, even when they are named differently than expected,
- identify data type heterogeneity and take further syntactical integration steps, and
- discover conceptual heterogeneity problems and thus avoid the construction of composite services producing unintended results.

We have shown in an example ontology how to give meaning to the symbols (labels of data types) used in WSDL descriptions. This is done by referencing the symbols to an application ontology, which in turn derives meaning for its concepts from a domain ontology, which in turn grounds its concepts on an image schemata-based grounding level.

We argue for a three level Semantic Reference System to allow for the different degrees of flexibility, unambiguousness and stability such a system has to provide (Figure 2). Apart from the grounding of meaning with image schemata on the grounding level, the innovation in this approach lies in the flexibility of the application ontology level. On this level, a service developer can model virtually anything (including totally fictive worlds) using restrictions on domain ontology concepts. Likewise, a user can find a certain concept based on a domain ontology vocabulary and learn how its meaning is restricted on the application ontology level. While this avoids that statements that are valid in a certain application ontology need to be valid in another one, they can still be traced back to their common domain ontology concept.

Future work includes the exploration of the reliability and usability of image schemata for grounding the semantics of domain ontology concepts. Also, we aim to extend the shown example into three distinct ontologies establishing a working prototype of the described Semantic Reference System. This involves the implementation of an environment for testing the semantics of inputs and outputs of composite services.

ACKNOWLEDGEMENTS

The work presented in this paper has been supported by the European Commission through the ACE-GIS project (grant number IST-2002-37724) and the German Federal Ministry for Education and Research as part of the GEOTECHNOLOGIEN program (grant number 03F0369A). It can be referenced as publication no. GEOTECH-51.

REFERENCES

- Barkowsky, T. (2001). Mental Processing of Geographic Knowledge. *Spatial Information Theory - Foundations of Geographic Information Science, Proceedings of COSIT 2001, Morro Bay, CA, USA, September 2001*. D. Montello. Berlin, Heidelberg, New York, Springer. 2205: 371-386.
- Bellwood, T., L. Clément, et al. (2003): UDDI Version 3.0.1: <http://uddi.org/pubs/uddi-v3.0.1-20031014.htm>, last access: 2003-11-28.
- Bishr, Y. (1998). "Overcoming the semantic and other barriers to GIS interoperability." *International Journal of Geographical Information Science* 12 (4): 299-314.
- Chinnici, R., J.-J. Moreau, et al. (2003): Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language: <http://www.w3.org/TR/2003/WD-wsdl20-20031110/>, last access: 2003-11-28.
- Gärdenfors, P. (2000). *Conceptual Spaces - The Geometry of Thought*. Cambridge, MA, Bradford Books, MIT Press.
- Gruber, T. (1993). "A Translation Approach to Portable Ontology Specifications." *Knowledge Acquisition* 5 (2): 199-220.
- Guarino, N. (1998). *Formal Ontology and Information Systems*. Formal Ontology in Information Systems, Trento, Italy, IOS Press.
- Haarslev, V. and R. Möller (2003): RACER User's Guide and Reference Manual Version 1.7.7: <http://www.sts.tu-harburg.de/~r.f.moeller/racer/racer-manual-1-7-7.pdf>, last access: 2003-11-15.
- Johnson, M. (1987). *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason*. Chicago, The University of Chicago Press.
- Kuhn, W. (2003). "Semantic Reference Systems." *International Journal of Geographical Information Science* (accepted for publication).

- Lutz, M., C. Riedemann, et al. (2003). A Classification Framework for Approaches to Achieving Semantic Interoperability. *COSIT 2003 (Conference on Spatial Information Theory)*. W. Kuhn, M. Worboys and S. Timpf. Ittingen, Switzerland, Springer. 2825: 200-217.
- McGuinness, D. L. and F. Van Harmelen (2003): OWL Web Ontology Language, W3C Candidate Recommendation: <http://www.w3.org/TR/2003/CR-owl-features-20030818/>, last access: 2003-10-15.
- Noy, N. F., M. Sintek, et al. (2001). "Creating Semantic Web Contents with Protege-2000." *IEEE Intelligent Systems* 16 (2): 60-70.