Ontology-based Integration of Sensor Web Services in Disaster Management

Grigori Babitski[1], Simon Bergweiler[2], Jörg Hoffmann[1], Daniel Schön[3], Christoph Stasch[4], and Alexander C. Walkowski[4]

[1] SAP Research, Karlsruhe, Germany
{grigori.babitski|joe.hoffmann}@sap.com
[2] DFKI, Saarbrücken, Germany
Simon.Bergweiler@dfki.de
[3] Itelligence AG, Köln, Germany
daniel.schoen@itelligence.de
[4] Institute for Geoinformatics, Münster, Germany
{staschc|walkowski}@uni-muenster.de

Abstract. With the specifications defined through the Sensor Web Enablement initiative of the Open Geospatial Consortium, flexible integration of sensor data is becoming a reality. Challenges remain in the discovery of appropriate sensor information and in the real-time fusion of this information. This is important, in particular, in disaster management, where the flow of information is overwhelming and sensor data must be easily accessible for non-experts (fire brigade officers). We propose to support, in this context, sensor discovery and fusion by "semantically" annotating sensor services with terms from an ontology. In doing so, we employ several well-known techniques from the GIS and Semantic Web worlds, e.g., for semantic matchmaking and data presentation. The novel contribution of our work is a carefully arranged tool architecture, aimed at providing optimal integration support, while keeping the cost for creating the annotations at bay. We address technical details regarding the interaction and functionality of the components, and the design of the required ontology. Based on the architecture, after minimal off-line effort, on-line discovery and integration of sensor data is no more difficult than using standard GIS applications.

1 Introduction

Disasters may be caused by flooding, earthquakes, technical malfunctions, or terrorist attacks, to name a few. The efficient handling of such emergencies, i.e., the management of the measures taken to fight them, is a key aspect of public security. This is especially true in an increasingly tightly interlinked world, where problems in one area may quickly cause problems in connected areas. This phenomenon often causes disasters to exhibit an explosive growth, especially during their early stages. Defensive measures in such a stage are still premature, leading in combination with the explosive growth to what has been termed the "chaos-phase" [22]. Methods for shortening that phase are widely believed to be essential for limiting the damage caused by the disaster.

One of the characteristics of the chaos-phase is the overwhelming flow of information that must be managed by the defense organizations, such as fire brigades and the police. Depending on the scale of the disaster, each organization establishes a *crisis team*, i.e., a committee of officers deciding which actions to take, and monitoring their execution. To come up with informed decisions, members of the crisis team must process an enormous amount of heterogenous information, such as messages from the public, feedback from own forces in the field or from partner organizations, and – last not least – Geospatial information such as weather conditions and water levels. Our focus herein is on the latter. Since not only is the amount of information huge, but also it must be evaluated in a situation of extreme stress and pressure, it is of paramount importance that the information can be accessed quickly and with complete ease.

In the SoKNOS project[5], we develop a service-oriented system facilitating amongst other things the integration of Geospatial information. This integration is realized in a Geographic Information component (GI Plugin), which offers functionalities to query data from several geospatial web services, to visualize the data in a map component, and to analyze the data through integrated GIS functionalities. Additional analyzing capabilities (e.g. simulations) can be intergrated by adding external processing services. The difficulty of integrating new information into the map depends on the form the information comes in. Our most basic assumption is that the information is encapsulated into Web services conforming with the standard specifications of the Open Geospatial Consortium (OGC). The integration of basic maps is realized through adding data from Web Mapping Services (WMS). Vector data (e.g. risk objects) can be accessed through Web Feature Services (WFS) and hence require the creation of suitable queries which poses serious challenges; indeed, given the stress and pressure of the targeted scenario, pre-specified queries are necessary.

An interesting and important middle ground are sensors, accessible through e.g. the Sensor Observation Services (SOS) as specified by the Sensor Web Enablement (SWE) initiative of the OGC. As sensor data is time-dependent,

what the user needs to provide is, essentially, the desired Geographic area, the desired time interval, and the desired properties to be observed. The SOS specification lays the basis for doing so in an interoperable manner. Areas and time points are fully covered by standards. The main problems remaining are:

(I) For identifying observed properties, mediation is required between the terminology of the user and that of the Web service design.
(II) The user may not even know a technical term for the observed property she is looking for, necessitating an option to search by related terms.
(III) For fusing the information of several sensors, data transformation (e.g. units of measurement) is needed, and duplicate data needs to be detected and removed.
(IV) Sensors may become dysfunctional and in such case need to be replaced with suitable alternative sensors.

Characteristic properties of disaster management are that (II) and (IV) are likely to occur, that the number and types of required sensor informations are manifold, that the persons needing them act under high pressure, and that these persons have hardly any IT knowledge. Given this, (I)–(IV) constitute a serious difficulty.

[5] Service-oriented architectures supporting networks in the context of public security; http://www.soknos.de

In our work, we have developed and implemented a tool architecture that addresses (I)–(IV), up to a point where discovery and integration of sensor data is no more difficult than using standard GIS applications. The key technique is to make use of *semantic annotations* in a purpose-designed *ontology*. The technicalities will be summarized directly below, and detailed later on in the paper. First, we need to clarify that our approach encompasses a separate *service registration* activity, which contrasts with *service usage*. These correspond to the two fundamentally different phases in our domain, *off-line* (prior to the disaster) vs. *on-line* (during the disaster). On-line, pressured and hectic users need to comfortably discover and integrate sensor data. As the basis for that, our approach assumes that – off-line, in peace and with ample time – each service has previously been registered. Such registration means to acquire the service (finding it in the Web), to create a description including the semantic annotation, and to store that description within a local registry.[6] Apart from exploiting the off-line phase in a suitable preparatory way, the distinction between service registration and service usage also serves for decoupling these activities, allowing them to be performed by different people. The person performing the registration will also be associated with the fire brigade/police. But she may well have more IT knowledge than typical crisis team members. (That said, clearly, this person will not be a logics expert, so creating the semantic annotations needs to be reasonably easy; if it is not, then the effort for creating them is very likely to lead to non-acceptance anyhow.)

A commonly used definition is that *an ontology is a formal, explicit specification of a shared conceptualization* [7]. In our context, we define an ontology called *Geosensor Discovery Ontology* (GDO). The GDO defines a terminology suitable for describing sensor observations and related entities. Put in simple terms, the GDO contains:

(a) A taxonomy of phenomena, i.e., of properties that can be observed by sensors.
(b) A taxonomy of substances to which phenomena (a) may pertain.
(c) A taxonomy of Geographic objects to which phenomena (a) may pertain.
(d) The relations between (a), (b), and (c).

To ensure sustainable modeling, the GDO design follows the guiding principles of the DOLCE foundational ontology [16, 5]. Simply put, DOLCE corresponds to a kind of widely accepted "best practice" for ontological modelling, serving to avoid common modelling flaws and shortcomings.

The semantic annotations associate, for a SOS service, each of the service's observed properties with a concept from (a). Clearly, these annotations are easy to create. Our architecture provides a simple user interface for doing so via drag-and-drop. In the obvious manner, the annotations solve problem (I). Since phenomena (a) are organized in a taxonomy (enabling us to find more general/more specialized sensors), the GDO also provides sophisticated support for problem (IV). Substances and Geographic objects are likely candidates a fire brigade officer will use as related terms, hence (b), (c), and (d) together serve to solve problem (II). Problem (III), finally, is solved by standard transformations and straightforward usage of the SOS output information.

[6] Hence the term "discovery" in this paper refers to finding a suitable sensor, on-line, in a (potentially huge) local registry, *not* in the Web.

It is also required to make the entire functionality easily accessible to the user. Our Graphical User Interface does so via standard paradigms, and intutive extensions thereof. For service discovery, the area of interest is marked by mouse movements as a rectangle on a map; the desired time points are given by manipulating the boundaries of a time interval; search in the GDO – which from the user's perspective corresponds to selecting the desired observations – is realized by text search combined with taxonomy browsing and following links (given by the relations between pairs of concepts in the ontology). Once services are discovered, fusing and displaying their data amounts to a single drag-and-drop action for the user. The architecture was successfully demonstrated to an evaluation team of German fire brigade and police officers, obtaining a very positive rating; we give some more details on this in Section 6.

The paper is organized as follows. Section 2 provides a brief background on the OGC Sensor Observation Service and the Semantic Web. Section 3 introduces concrete use cases that we will use for illustration. Section 4 covers our architecture, detailing after an overview the design of the GDO, the semantic annotations, as well as sensor discovery and fusion. Section 5 discusses related work, and Section 6 concludes with summary remarks and a discussion of open issues.

2 Background

We briefly give the most relevant background on the SOS service specification, and the Semantic Web domain.

2.1 Sensor Observation Service

The goal of the OGC Sensor Web Enablement initiative is to enable the creation of web-accessible sensor assets through common interfaces and encodings [2]. Therefore, the SWE initiative defines standards for the encoding of sensor data as well as standards for web service interfaces to access sensor data, task sensors or send and receive alerts. The Sensor Observation Service (SOS) is part of the SWE framework and offers a pull-based access to observations and sensor descriptions [18]. The SOS operations are grouped into three different profiles: the *core* profile for retrieving the service descriptions, sensor descriptions and observations; the *transactional* profile for registering new sensors and inserting new observations; the *enhanced* profile for offering additional service functionalities.

In this work, we focus on the basic operations of the SOS defined in the core profile. The core profile comprises the GetCapabilities, DescribeSensor and GetObservation operation. The GetCapabilities operation returns a service description of the service containing information about the supported operations and parameters as well as the observations which are provided, e.g. spatial and temporal extent of the observations, producing sensors and observed properties. Sensor metadata like sensor position, calibration information or sensor administrator can be retrieved using the DescribeSensor operation. The sensor descriptions are usually encoded in the Sensor Model Language (SensorML), a data model and XML encoding for sensor metadata [1]. The core operation of the SOS depicts the GetObservation operation. It offers the possibility to query observations filtered by spatial and temporal extent, producing sensors, certain observed properties, and/or value filters.

The Observations and Measurements (O&M) specification [3] is utilized by the SOS to encode the data gathered by sensors. It defines a model describing sensor observations as an act of observing a certain phenomenon. The basic observation model contains five components: The *procedure* provides a link to the sensor which generates the value for the observation.The *observedProperty* references the phenomenon which was observed. The *Feature Of Interest (FOI)* refers to the real world entity (e.g., a river) which was target of the observation. The time, when the observation was made, is indicated by the *samplingTime* attribute.The *result* element contains the observation value. The observation acts as a property value provider for a feature: It provides a value (e.g. 27 Celsius) for an observed property (e.g. temperature) of the FOI (e.g. weather station) at a certain timestamp. The location to which the observation belongs is indirectly referenced by the geometry of the FOI.

2.2 Semantic Web

In a nutshell (and as far as relevant for this paper), the Semantic Web community is concerned with the investigation of how *annotations* within a *formal language* can help with performing many tasks in a more flexible and effective way. Specifically, we are herein concerned with a form of *semantic service discovery*. The idea is that each Web service of interest is annotated with (an abstract representation of) its meaning – what does it do? – and services are discovered by matching this annotation against a discovery query – what kind of service is wanted? – given in the same logic. Since the annotations and queries, formulated relative to a formal domain model encoding complex dependencies, can be far more precise than free text descriptions, this approach has the potential to dramatically improve precision and recall.

Semantic discovery is, by the standards of the field, a long-standing topic in the Semantic Web. Earlier approaches were often based on annotating with, and reasoning about, complex logic languages such as 1st-order logic or rich subsets thereof. See e.g. [13] for a classical Desciption Logics formalization. Arguably, most of these approaches suffer from the prohibitive complexity of creating semantic annotations and discovery queries (and from the prohibitive computational complexity of the required reasoning). A more recent trend in the Semantic Web community is to use more “lightweight” approaches putting less of a burden on these activities, at the cost of reduced generality and power – the slogan being “a little semantics goes a long way” [8]. Our approach falls into this class, with carefully designed technology targeted at providing added value, while keeping the complexity at a level that will lead to actual acceptance by end users (fire brigades etc) in the relevant domain.

3 Example Scenario

In our example scenario, the floodwater level of the Rhine river in Germany rises immensely during a long lasting thunderstorm. Cologne and the industry park of Dormagen are affected by the flood. People have to be evacuated and organizations from other

German federal states are called to support the disaster management. After a dike has broken and a chemical plant is flooded nearby the Rhine river, explosions occur which release pollutants into the air and the water. The emergency staff as well as residential areas around the chemical plant are threatened by the released air and water pollutants. We consider the following use cases for the proposed architecture:

(A) **Discovery and fusion of heterogenous water level measurements.** To get a more precise overview, all water gauges along the Rhine upstream of Cologne shall be integrated into the SoKNOS System. The sensor data is provided by different SOS services, using different identifiers for the observed phenomenon (e.g. water level, water gauge, gauge height), using different units of measurement, and partially overlapping each other. The challenges addressed by our architecture are to mediate between the identifiers and the terminology of the non-expert user, to make the sensors easy to find among a huge set of available sensors, to merge multiple data points, and to recognize redundant data.

(B) **Replacement of a water level measurement sensor.** The data displayed to the crisis team of course must be up-to-date. Since access to SOS services is pull-based, the map component sends new queries periodically. One of the sensors may have become damaged, and hence may now be out of order. The challenge addressed by our architecture is to recognize this, and to discover and integrate a suitable replacement sensor automatically.

(C) **Discovery and fusion of heterogenous air pollutant concentration measurements.** With conventional methods, the monitoring of air pollutant concentration is a time consuming and complicated task. There are only few vehicles with appropriate sensors. Hence the spatial resolution of the measured values is rather coarse grained. It takes considerable time for the vehicles to arrive at the area of interest, and the measurements are transferred through verbal communication, prone to delays and misunderstandings. This can be improved considerably through leveraging on resources – SOS services – that happen to be available in the particular scenario: the monitoring systems of chemical plants near the flooding. These SOS services could of course also be integrated off-line into conventional systems. But our approach allows to discover and use them with ease, based on minimal integration effort. Indeed, since registering a service requires hardly more effort than knowing where the service is and which phenomena it observes (see Section 4.3 below), it is conceivable that the integration is performed on-line, e.g. by a system administrator, upon demand by the crisis team members.

4 Semantic Sensor Integration

We now explain in detail our architecture, its individual components, and their design and functionality. We begin in Section 4.1 with an overview, giving a rough picture of the components and their interaction. We then delve into the details, describing in Section 4.2 the design of our ontology, explaining in Section 4.3 our semantic annotations and how they are created, describing in Section 4.4 our methods for sensor discovery, and describing in in Section 4.5 our methods for sensor data extraction and fusion. All user interactions are illustrated with screen shots, and all methods are exemplified with the use cases introduced in the previous section.

4.1 Architecture

Figure 1 shows an overview of our architecture. There are six components. Two of these are graphical user interfaces (GUIs, shown in the top left part of the figure), two are backend components (shown in the bottom left part), and two are data stores (shown on the right).

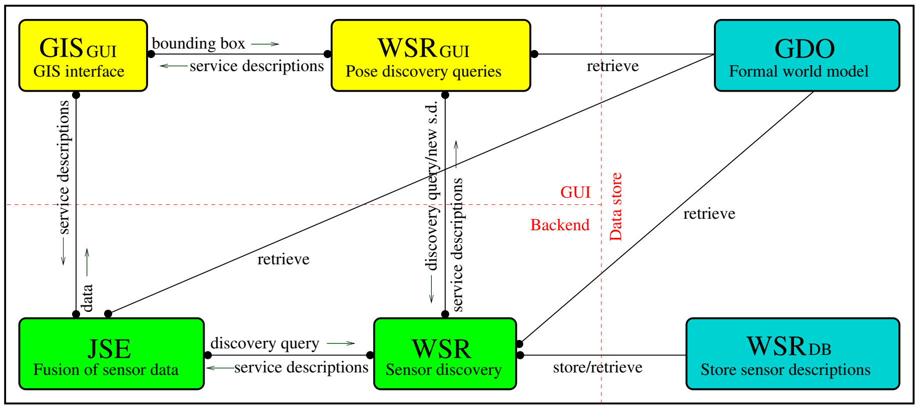


Fig. 1. An overview of our architecture.

The Geographic Information System (GIS) GUI is basically a standard GIS map component, extended to cater for the required interactions with the Web Service Registry (WSR) GUI and the Joined Sensor Engine (JSE). The Web Service Registry GUI is the user interface of the Web Service Registry, which serves for registering and discovering Web service descriptions – i.e., descriptions of SOS services, including their semantic annotations, in our case. The Joint Sensor Engine extracts the data from a set of discovered services. It makes the required data transformations and it detects duplicate data. Most importantly, it monitors the performance of the services, and replaces them – via posing a suitable discovery query to the WSR – fully automatically in case of failure. The Geosensor Discovery Ontology (GDO) is a formalization of the domain, i.e., of the relevant terminology relating to sensor data, as outlined in the introduction. The Web Service Registry database (DB), finally, is the storage container for service descriptions. A brief summary of the interactions is as follows:

– **GIS GUI with Web Service Registry GUI.** The user specifies a bounding box via marking a rectangle on the map within the GIS GUI; the bounding box is sent to the Web Service Registry GUI, to form part of the discovery query. The discovery query is completed in the Web Service Registry GUI, and the discovered services are sent back to the GIS GUI. From that point on, the GIS GUI is responsible for displaying the data of these services.

– **Web Service Registry GUI with Web Service Registry.** Discovery queries are created in the Web Service Registry GUI, comprising the desired area (the bounding box), the desired time interval, as well as the desired kind of phenomenon to be observed. The queries are sent to the Web Service Registry, which performs the discovery and sends the discovered service descriptions back to the Web Service Registry GUI. Additionally, the user may enter a new service description (possibly including a semantic annotation) in the Web Service Registry GUI, which is then sent to the Web Service Registry for storage.
– **GIS GUI with Joined Sensor Engine.** Whenever the GIS GUI needs to extract up-to-date data from the discovered sensors, it sends their descriptions to the Joined Sensor Engine. Based on the descriptions, the Joint Sensor Engine connects to the services, and extracts and fuses their data, which is then sent back (as a single data set) to the GIS GUI.
– **Joined Sensor Engine with Web Service Registry.** Whenever service monitoring inside the Joint Sensor Engine finds that a sensor has failed, it queries the Web Service Registry for replacement services, delivering equivalent data.
– **Web Service Registry with Web Service Registry DB.** The Web Service Registry connects to the database for storage and retrieval of service descriptions.
– **Web Service Registry GUI with Geosensor Discovery Ontology.** For specifying a discovery query, the user needs to find the desired concepts in the Geosensor Discovery Ontology, i.e., suitable phenomena or related entities. For that, the Web Service Registry GUI uses the structure of the Geosensor Discovery Ontology, which is read from the storage.
– **Web Service Registry with Geosensor Discovery Ontology.** Discovery is made not only directly on the concepts in the query, but also indirectly through the connections within the Geosensor Discovery Ontology, read from the storage.
– **Joined Sensor Engine with Geosensor Discovery Ontology.** For the purpose of data transformation, the Joined Sensor Engine needs information from the Geosensor Discovery Ontology in order to detect equivalent observed properties.

These functionalities and interactions will now be explained in detail. We start by detailing the structure of the GDO, which lies at the heart of our approach.

4.2 Ontology Design

The GDO is formalized in F-Logic [12], a logic based programming language which we chose mainly for practical reasons: F-Logic provides sufficient modelling power for our purposes, while at the same time being computationally efficient in the reasoning tasks we require.[7] In what follows, we do not delve into details of the formalization. Instead, we describe the design of the GDO at an intuitive level.

The GDO is designed to support discovery of SOS services, so, naturally, it builds on the relevant specifications [3, 18]. SOS service descriptions contain keywords (called “observed properties” in (O&M) [3]) indicating the properties measured by the sensor.

[7] There is also a version of the GDO formulated in the standard description logic based language OWL [17]. In our work, this version mainly serves as a reference model. For the sake of simplicity, we do not discuss the OWL version and its relation to the F-Logic version.

These properties are not standardized, but the CF Metadata[8] contains a (incomplete) collection. The GDO models those properties relevant for our application, as well as some supplementary entities, in the form of taxonomies of categories. Our technology connects those to real sensors via F-Logic rules.

An important aspect of the GDO is that it follows well-established ontological design principles. We align the GDO with the well-known DOLCE foundational ontology. DOLCE essentially is a kind of widely accepted "best practice" for ontological modelling. This serves to avoid common modelling flaws and shortcomings. For details regardng DOLCE, we refer the reader to the literature [16, 5, 6]. In what follows, a rough understanding of the following four concepts will suffice. **Endurants** and **perdurants** are distinct regarding their behavior in time. Endurants are wholly present at any time they exist, whereas perdurants extend in time by accumulating different temporal parts. Perdurants embrace entities generally classified as events, processes, and activities. An endurant "lives" in time by participating in some perdurant(s). For example, a building (endurant) participates in its lifespan (perdurant). In the GDO, we use two sub-categories of endurant: "non-agentive physical object" and "amount of matter". **Qualities** are the basic entities we can perceive or measure, for example the volume of a lake, the color of a rose, or the length of a street. DOLCE distinguishes physical and temporal qualities, which pertain to endurants and perdurants, respectively. **Roles** are played by endurants. For example, a physical object may play the role "observed object", but it may also play the role, e.g., of an "operation site" or of a "target".

To exemplify the importance of such ontological precision: in (O&M), some vital concepts are under-specified or ambigiously defined. For example, "observed property" and "phenomenon" are defined vaguely and used more or less like synonyms. According to DOLCE, they would be a mixture of endurant, perdurant, and quality (see a detailed discussion in [19]). Similarly, "feature of interest" is not perceived as a role (which is done according to DOLCE), but instead as an endurant – although, quite clearly, being observed is not a characteristic property of an object. The Rhine is a river; will it become a different object because it is being observed? Such terminological inclarity is unproblematic when used amongst members of a closed community who know what is meant, but may cause problems when crossing community boundaries – e.g. during a disaster. That said, the GDO is not dogmatic in its alignment to DOLCE; we follow the DOLCE guidelines where sensible, and opt for pragmatic solutions in cases where a full solution would unnecessarily complicate matters.

The GDO is based on the design pattern depicted in Figure 2. That is, the ontology is built as a specialization of that pattern, extending the pattern's high-level categories with whole taxonomies, i.e., with hierarchies of more concrete categories, and instantiating the high-level relations with relations between such concrete categories. In what follows, we briefly explain the main aspects of the design.

At first glance, one sees that the pattern does not only cover sensor observations – **observable qualities** – but also **weather phenomenon**, **substance**, **geosphere region**, and **boundary of geosphere regions**. This enables *search by related terms*: rather than laborously searching through a huge set of observable qualities, the user may select a

[8] NetCDF Climate and Forecast (CF) Metadata Convention (http://cf-pcmdi.llnl.gov)

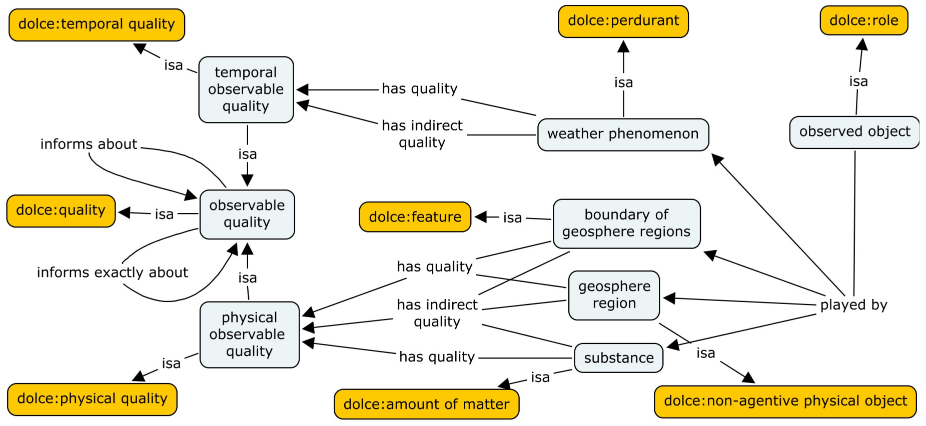


Fig. 2. The design pattern underlying the GDO (Geosensor Discovery Ontology), slightly simplified for presentation. Concepts inherited from DOLCE are marked by inscription and color.

related concept which pertains to the desired quality.[9] The advantage is that the taxonomies of related concepts tend to be much smaller than that of possible sensor observations. For example, for a non-expert user “wind direction” (or “water level”) are probably much easier to find via “wind” (or “river”) than via browsing the taxonomy of observable qualities. That said, browsing is of course also an option in our system.

In the GDO, **weather phenomenon** captures things such as rain shower, wind, fog; **substance** is orientated at chemical terminology, distinguishing between pure substances and blended subtances, covering things such as oxygen and nitratemonoxide (pure substances), and salt water (a mixture of substances); **geosphere region** covers things such as athmosphere, ground, body of water; **boundary of geosphere regions** covers things such as earth surface, water surface. If needed, these 4 top-level categories can easily be augmented by additional ones. One simply adds the new categories, classifies them according to DOLCE, and gives them the **played by** relation to **observed object** – which is defined as a role, c.f. the above discussion.

In accordance with DOLCE, observable qualities are distinguished into temporal ones (e.g. speed, flow rate) and physical ones (e.g. temperature, distance). Another aspect worth noting is that observable qualities may be related – one quality **informs about** another – or even equivalent – one quality **informs exactly about** another. An example of the former is fog density, which informs about range of sight. An example of the latter are the two ways of observing wind direction: *from where* vs. *whereto*.

4.3 Semantic Annotation

As stated, our semantic annotations are simple, in order to ensure practicality for organizations such as fire brigades. The precise form of the annotations is as follows:

[9] The relation may be direct or indirect; hence the **has quality** and **has indirect quality** relations in Figure 2. To exemplify the difference: water (directly) has a temperature; in contrast, pressure is not a property of the athmosphere, but is often (indirectly) associated with it.

Definition 1. *Assume that s is a SOS service. A* service description *of s is any set D that contains the URL of s as well as a* semantic annotation *α of s, defined as follows. Assume that $OP(s) = \{op_1, \dots, op_k\}$ is the set of observed properties supported by s, across offerings, and assume that OQ is the set of concepts in the GDO that are sub-concepts of* ***observable quality****. Then a semantic annotation of s is a partial function $\alpha : OP(s) \mapsto OQ$.*

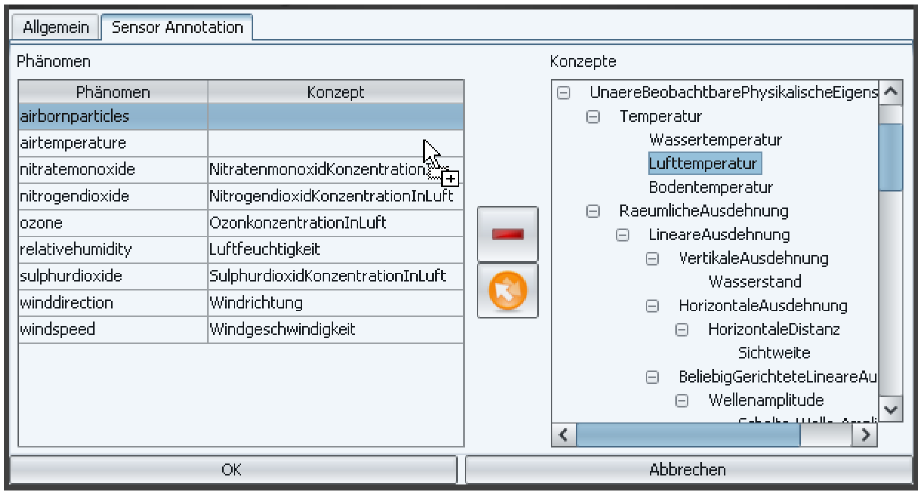
Sub-concept here refers to the taxonomic structure of the GDO: concept c_1 is a sub-concept of concept c_2 iff c_1 lies below c_2 (directly or indirectly) in the tree of concepts. In practice, and in our prototype, of course the form of the service descriptions (i.e., the precise set of attributes stored for each service) is fixed. What that form is – other than that it complies with Definition 1 – is not important to this work. Note that α is a partial function, hence allowing the annotation to be incomplete. This allows to register a service without giving it a full semantic annotation. In order to use a particular output (a particular observed property) of a service with our architecture, that output must be annotated, i.e., be in the domain of the annotation function α.

Each observed property is characterized by a single concept of the GDO. This is appropriate because it complies well with the intended meaning of the SOS specification: each sensor output corresponds to one atomic category of possible observations. It is important to note that such a simple correspondence would *not* be valid for more complex OGC services. For example, it would make no sense to restrict the annotation of a WFS service to a single concept in an ontology: since WFS services are databases that may contain a whole variety of data, a description of their data content would definitely need to be some sort of combination of concepts (see also [15]). From a Semantic Web perspective, ours is a classical example of a light-weight approach, c.f. Section 2.2.

In our architecture, the simple semantic annotations as per Definition 1 suffice to conveniently discover and, where needed, replace SOS services (details follow in the next sub-sections). Creating the annotations can, obviously, be supported in a straightforward manner using classical GUI paradigms. Figure 3 shows a screenshot of our implemented tool, in a situation corresponding to use case (C) of Section 3, i.e., annotation of air pollutant concentration measurements with concepts from the ontology.

As can be seen in Figure 3, the WSR GUI contains a tab for annotating sensor services. The WSR displays the service's observed properties, as well as any α assignments that have already been made. In a separate part of the window ("Konzepte"), the ontology is displayed. One can search concepts in the ontology via several options that will be detailed in the next section, when we describe how to create discovery queries. Once the desired concept is found, one simply drags it onto the corresponding observed property – in Figure 3, the concept "Lufttemperatur" is dragged onto the output property "airtemperature". The new assignment is stored in the service's annotation α. If the output was already assigned previously, then that assignment is over-written.

Clearly, this annotation process requires no more expertise than a basic familiarity with computers, as well as some familiarity with SOS service observations and with the GDO. It is realistic to assume that such expertise will be available, or easy to create, within the relevant organizations and their partners.



Phänomen	Konzept
airbornparticles	
airtemperature	
nitratemonoxide	NitratenmonoxidKonzentration
nitrogendioxide	NitrogendioxidKonzentrationInLuft
ozone	OzonkonzentrationInLuft
relativehumidity	Luftfeuchtigkeit
sulphurdioxide	SulphurdioxidKonzentrationInLuft
winddirection	Windrichtung
windspeed	Windgeschwindigkeit

- UnaereBeobachtbarePhysikalischeEigens
  - Temperatur
    - Wassertemperatur
    - Lufttemperatur
    - Bodentemperatur
  - RaeumlicheAusdehnung
    - LineareAusdehnung
      - VertikaleAusdehnung
        - Wasserstand
      - HorizontaleAusdehnung
        - HorizontaleDistanz
          - Sichtweite
      - BeliebigGerichteteLineareAu
        - Wellenamplitude

Fig. 3. A screen shot of our GUI for creating semantic annotations. Since our tool is built in cooperation with (and for the use of) German disaster defence organizations, the inscriptions are in German; explanations are in the text.

4.4 Sensor Discovery

As is common in semantic service discovery, c.f. Section 2.2, the discovery is formulated as a process of matching the available services against a discovery query. In our approach, the semantic annotations serve for terminology mediation, and for allowing *indirect* matches. The latter enables the user to find the desired services via intuitively related terms, rather than having to laborously search for the actual technical term.

Service descriptions and the semantic annotations they contain were defined already in Definition 1. Discovery queries and matches are defined as follows:

Definition 2. *Assume that $\mathcal{CO}$ is the set of all concepts in the GDO. A* semantic discovery query *sQ is a subset $sQ \subseteq \mathcal{CO}$. Assume that D is the description of a service s, that $OP(s) = \{op_1, \dots, op_k\}$ is the set of observed properties supported by s, and that $\alpha \in D$ is the semantic annotation of s. Then sQ and s* match *in op_i iff op_i is in the domain of α, $\alpha(op_i) = c_0$, and there exists $q_0 \in sQ$ such that q_0* is connected to *c_0. The latter notion is defined inductively as follows:*

(1) Every $c \in \mathcal{CO}$ is connected to itself.
(2) If the GDO contains a relation with domain $c_1 \in \mathcal{CO}$ and range $c_2 \in \mathcal{CO}$, then c_1 is connected to c_2.
(3) If $c_1 \in \mathcal{CO}$ is a super-concept of $c_2 \in \mathcal{CO}$, then c_1 is connected to c_2.
(4) If $c_1 \in \mathcal{CO}$ is connected to $c_2 \in \mathcal{CO}$, and c_2 is connected to $c_3 \in \mathcal{CO}$, then c_1 is connected to c_3.

In words, a discovery query is just some collection of terms from the ontology. What the discovery does is to look for services s whose annotation contains a term c_0 which one of the query terms (namely q_0 in the definition) is "connected" to. All these services s – along with the relevant observation op_i and ontology term c_0 – are returned, provided the spatial and temporal aspects match as well (see below).

Connected in Definition 2 refers to a combination of relations in, and taxonomic structure of, the GDO. It is best understood as defining a set of possible paths through the ontology. Item (1) in Definition 2 says that empty paths are allowed: a query concept q is, of course, relevant to itself. Item (2) says that a path may follow a relation between two concepts c_1 and c_2 – if c_1 is relevant to the query, then c_2 is as well because c_1 relates to c_2. For example, c_1 may be the concept **river**, the relation may be **has quality**, and c_2 may be **water level**; c.f. use case (A) of Section 3. Item (3) in Definition 2 says that a path may go downwards in the taxonomy, i.e., go from c_1 to c_2 if c_1 lies above c_2 in the taxonomy. This is so because, if c_1 is relevant to the query and c_2 is a special case of c_1, then clearly c_2 is relevant to the query as well. For example, the query concept may be **body of water**, which is a super-concept of **river**, from which by item (2) we may get to **water level**. Item (4) states transitivity, a technical vehicle for expressing concisely whether or not there exists a path between two concepts.

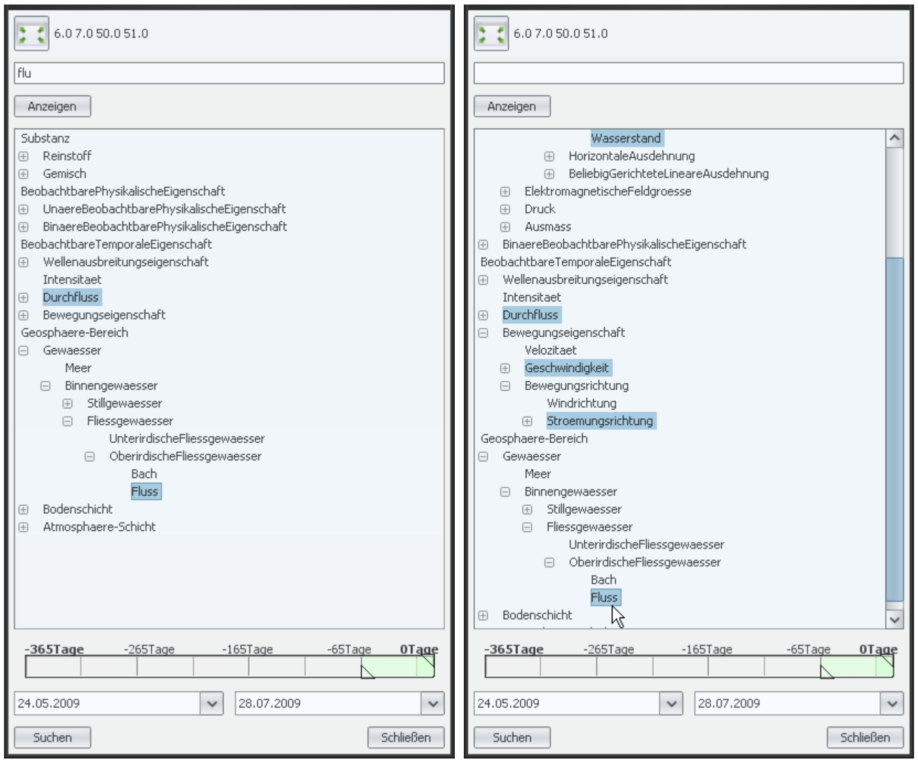
Items (1)–(4) in Definition 2 are implemented in a straightforward way using F-Logic *rules*. Such a rule takes the form $\textit{rule-head} \Leftarrow \textit{rule-body}$, meaning that truth of the rule body (right hand side) implies truth of the rule head (left hand side). Rule head and body are composed of F-Logic atoms. Item (4), e.g., is implemented by the rule `∀X,Y,Z connected(X,Z) <- connected(X,Y) AND connected(Y,Z).` While one could of course implement items (1)–(4) "by hand", the F-Logic implementation is efficient, and has the advantage of full flexibility: our approach and implementation can be trivially adapted to extended or modified matching methods, as long as the matching is expressible within the realm of F-Logic.

The above clarifies the *semantic* part of the discovery. On top of that, we need to specify the desired geographical region and time points. Consequently, a discovery query Q consists of a semantic discovery query sQ in combination with a bounding box *bb* and a time interval ti, both defined in the usual way. An observed property op_i of a service matches a query Q iff it matches sQ according to Definition 2, *and* the bounding box of the corresponding offering has a non-empty intersection with *bb*, *and* the time interval of the offering has a non-empty intersection with ti.[10]

Having clarified the inner workings of discovery, the important question remains how that functionality interfaces with the user. *How do non-experts such as fire brigade officers, acting under great stress, create discovery queries?* Given that our queries are combinations of standard constructs and very light-weight semantics, such query creation is quite feasible. Figure 4 shows the relevant screen shots for illustration.

We do not show a screenshot for specifying the bounding box and time interval because these interactions are obvious. The bounding box is specified within the GIS Plugin via marking a rectangle on the map. The time interval is specified via a time line with lower and upper bounds, shown at the bottom of the windows in Figure 4 (in the windows, the right-hand part of the interval has been selected). The core part of query creation consists of finding the desired set sQ of terms from the ontology. The WSR

[10] One can rank the services depending on the match quality. In our implementation, the ranking is a combination of the distance (path length) between the relevant query and annotation terms (q_0 respectively c_0), as well as the size of the intersections with *bb* respectively ti.


6.0 7.0 50.0 51.0

flu

24.05.2009 28.07.2009

6.0 7.0 50.0 51.0

24.05.2009 28.07.2009

Fig. 4. Screen shots of our GUI for creating discovery queries.

GUI offers three options: *text search*, *browsing*, and *following relations*. The first two facilities are illustrated in Figure 4 left-hand side, the third one is shown in Figure 4 right hand side. On the left, the user has entered the text “Flu”, which string-matches with “Fluss” (**river**); the GDO taxonomy tree is opened, highlighting that concept. Alternatively, the user could choose to browse for **river**, which would be done via clicking downwards in the taxonomy tree shown below “Geosphaere-Bereich” (**geosphere region**). On the right, the user wishes to give the precise phenomenon for the query, and chooses to look at the terms related to “Fluss” (**river**). This is done by a double-click on that concept. All related concepts, among them the desired “Wasserstand” (**water level**) are shown and highlighted.

Note how this form of discovery addresses problems (I) and (II) described in the introduction. Problem (I) – mediation is required between the terminology of the user and that of the Web service design – does not occur in Figure 4 because the required mediation has already been done at the point these interactions happen. The translation of terms is stored in the semantic annotations, and from the point of view of the end-user (who is likely to be different from the person doing the annotations) there *is* only one terminology. As for problem (II) – the user may not even know a technical term for the observed property she is looking for – this is addressed by the option to follow relations

(Figure 4 (b)), and by the option to not even search for the actual phenomenon by hand but instead leave it up to indirect discovery (c.f. Definition 2) to make the connections.

Once the completed discovery query has been sent to the WSR, all matching services are returned. The user may simply select all these services, or, in case the query was for more general ontology terms, he/she may select a subset. To help with the latter, the WSR GUI offers the option to display, for each service, the actual observations (the ontology terms annotated at the service) that match the query.[11]

4.5 Sensor Fusion and Replacement

Our architecture also serves to fuse data from different sensor services (c.f. problem (III) from the introduction), and to replace damaged sensors through appropriate other sensors (c.f. problem (IV) from the introduction). This is realized by the Joint Sensor Engine (JSE). After the user has selected sensor services in the WSR GUI and dropped them into the GIS GUI, the GIS GUI sends a request to the JSE. The JSE retrieves the data from the SOSs, and transforms these as necessary. Afterwards, new observation layers are added to the map displaying the features of interest (FOI) as well as the actual sensor values. We now describe these functionalities in more detail. We ignore the case where the user selects only a single service in the WSR GUI. Obviously, this is simpler to handle than the more general case where several services are selected.

After the user has dropped the services onto the GIS GUI, a sensor request is created and sent to the JSE. This request includes the endpoints of the sensor services, a layer id, the observed properties, the sensor IDs, and a temporal and spatial extent. The JSE translates the sensor request into service-specific SOS requests, and calls the services accordingly. The SOS responses are then merged as follows.

First, depending how data is distributed over several SOS instances, there may be redundant data provided by more than one instance. For example, in our use case (A) from Section 3, two sensors for data "upstream of Cologne" and "downstream of Cologne" might duplicate the data for Cologne itself. The JSE checks whether such duplicates occur, by comparing the relevant concepts of the GDO. If the observed properties (i.e., the annotated concepts) are the same, and that is also the case for the FOIs and the time-stamps of the data, then only one of the duplicate values is considered.[12]

Second, data transformation may be necessary. Trivially, this is the case for units of measurement, which need to be normalized to the style of presentation used in the GIS GUI. This is done via standard techniques. The more interesting case is that of sensors which measure equivalent observable qualities, such as wind direction *from where* vs. *whereto*. Note that this is an important issue for crisis team work because, to correctly interpret such data, without IT support one needs to be aware of rather subtle context information – e.g. wind direction is interpreted differently in Germany and the Netherlands, so one would need to take the respective location of the service into consideration. The GDO resolves this issue via the aforementioned **informs exactly**

[11] More advanced support may be possible relating to, e.g., quality-of-service parameters of the services. This is a direction for future work.

[12] Note here that the GDO is required for being able to do so: duplicate detection via sensor IDs is not possible because those IDs are not maintained globally, i.e., across SOS services.

about relation, c.f. Section 4.2. By virtue of the semantic annotations, the JSE knows that the observations are different; by virtue of the **informs exactly about** relation, the JSE knows that they are equivalent. That said, our solution is preliminary in that the GDO does not state how to actually transform measurements of these observations into one another. To state this in the GDO, one would need to include arithmetic terms in the ontology. This is not possible in either of OWL or F-Logic. Our current implementation simply hard-codes this arithmetic into the JSE. A more flexible solution, e.g. via stating the arithmetics within ontology comments, is a topic for future work.

The JSE monitors service invocations, and automatically replaces a service if the monitoring concludes that the service is not functional anymore. We explain below exactly when that conclusion is made. First, we define what sensor replacements are:

Definition 3. *Assume that s is a service, that $OP(s) = \{op_1, \dots, op_k\}$ is the set of observed properties supported by s, that D is the description of s, and that $\alpha \in D$ is the semantic annotation of s. Assume similar notations for another service s'. Then* s' can replace s in op_i by op'_j *iff: op_i is in the domain of α, $\alpha(op_i) = c_0$; op'_j is in the domain of α', $\alpha'(op'_j) = c'_0$; and c'_0* can replace c_0. *The latter notion is defined inductively as follows:*

(1) Every $c \in \mathcal{CO}$ can replace itself.
(2) If $c_1 \in \mathcal{CO}$ is a sub-concept of $c_2 \in \mathcal{CO}$, then c_1 can replace c_2.
(3) If c_1 ***informs exactly about*** *c_2 according to the GDO, then c_1 can replace c_2.*

Note that, as before, this definition covers only the semantic part of the replacement. In addition to the conditions stated, we require that the respective FOIs are identical, and that the respective time stamp of s' is at least as recent as the last valid measurement provided by s. Definition 3 should be largely self-explanatory. Item (1) is obvious, item (2) says that we can replace a sensor with a more specialized sensor, and item (3) states that we can replace a sensor with a sensor providing an equivalent observation. These items may be combined in an arbitrary fashion. For illustration of item (3), re-consider the wind direction example mentioned above. An example where item (2) is relevant is that were both s and s' measure the speed of a river, and op_i is annotated with **velocity** while op'_j is annotated with **stream velocity**. To illustrate item (1), consider use case (B) from Section 3, where water level sensors may require replacement.

To perform a replacement, the JSE contacts the WSR with a discovery query that contains the URL of s, as well as the desired observation op_i (if more than one op_i are needed, several queries are posed). The WSR returns the suitable replacements s' and op'_j as per the above. The replacement is triggered iff monitoring detects one of the following situations: the service does not respond; an error occurs; the answering time exceeds a given time interval; the observation values provided by a specific sensor are empty or outside a given interval.

5 Related Work

There are several projects in which OGC SWE services have been applied to risk monitoring and disaster management, e.g. [11, 20]. In difference to our work, these projects focus on service architectures and SWE protocols for data exchange and fusion without any formalized knowledge.

There is some previous work on creating ontologies in the context of SOS services, most importantly the SWEET project[13] which has developed ontologies [21] that cover a broad spectrum of GIS terminology. The GDO models the SoKNOS-relevant subset of observable qualities defined in the CF Metadata and in SWEET. The ontology structure of SWEET was examined closely, and some relevant approaches were adapted to suit DOLCE. Overall, the GDO is more specialized than SWEET, and more suitable for our application; it is distinguished through its conformity with DOLCE.

Semantic discovery of OGC services has previously been investigated in the following three works. [15] design a Desciption Logics based approach to discovery of WFS services, and [14] design a 1st-order approach to WPS service discovery. Recent work has developed a more light-weight logic programming based approach to WPS discovery [4]. Although our approach uses similar machinery (F-Logic), there is no technical or conceptual relation between the two works.[14] Recent work [9] is similar to ours in that it also addresses semantic annotation of SOS services. However, the intentions, and consequently the employed methods, are very different. Whereas we aim at quick discovery and fusion of sensors in situations of great stress, [9] aim at a deep analysis of sensor *data*, automatically identifying phenomena such as blizzards from the sensor output. Thus, in stark difference to our light-weight annotation of SOS descriptions, [9] use more heavy-weight annotation and reasoning about data *content*. Finally, in [10] a method is proposed for linking Geosensor network data and ontologies. In difference to our work, the focus of [10] is mostly on the generation of annotations, the main contribution being an implementation of such methods within the Protégé ontology editor; also, the application domain is different, namely transportation.

6 Conclusion

We have presented an architecture for flexible discovery and integration of SOS services, based on light-weight semantic annotations. The annotations are sufficiently easy to create for end-user acceptance, while at the same time they provide significant added value through the ease of finding suitable sensors, and the ease of fusing their data and dealing with service failure. Hence ours appears to be a good compromise between the power of semantic annotations and the difficulty of creating and maintaining them.

An evaluation by real fire brigade men has largely confirmed this view. Three groups of men ranked the discovery functionalities – text search, browsing, linked concepts, indirect discovery – with school grades. All grades were among the best 2 grades available, and top grades were given 5 times. The men expressed the view that such a tool would be useful for crisis team work. They were especially enthusiastic about indirect discovery (discovery via related terms) because, under the stress of a crisis, it will often happen that crisis team members don't immediately recall the correct technical terms.

[13] Semantic Web for Earth and Environmental Terminology, http://sweet.jpl.nasa.gov/index.html

[14] [4] focus exclusively on formulating the dependencies between inputs and outputs of a service – an issue which does not even arise for SOS services. While we match through a notion of paths through the ontology, [4] use query containment.

Of course, our architecture is far from perfect and several issues have been left unaddressed as yet. Some important ones regard the selection of services, once a set has been discovered. Our current ranking methods are fairly primitive. A tool for quickly comparing services, i.e., showing at one glance their most relevant strong/weak aspects, would be desirable. Also, depending on the level of user acceptance of creating more complex annotations, techniques such as presented in [9] (c.f. Section 5) may be quite useful for automatically issueing warnings regarding potentially dangerous events.

Acknowledgments

Part of this work was supported by the German Federal Ministry of Education and Research (BMBF) under grant number 01—S07009 and the SAP AG within the context of the SoKNOS project.

References

1. Botts, M.: OpenGIS Sensor Model Language (SensorML) implementation specification, version 1.0.0. Tech. Rep. 07-00, Open Geospatial Consortium (2007)
2. Botts, M., Percivall, G., Reed, C., Davidson, J.: OGC white paper - OGC Sensor Web Enablement: Overview and high level architecture. Tech. Rep. 07-165, Open Geospatial Consortium (2007)
3. Cox, S.: Observations and Measurements - part 1- observation schema version 1.0. Tech. Rep. 07-022r1, Open Geospatial Consortium (2007)
4. Fitzner, D., Hoffmann, J., Klien, E.: Functional description of geoprocessing services as conjunctive datalog queries. Geoinformatica (2009), currently under review
5. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A.: Sweetening WordNet with DOLCE. AI Magazine 24 (3), 13–24. (2003)
6. Gangemi, A., Mika, P.: Understanding the semantic web through descriptions and situations. In: DOA/CoopIS/ODBASE 2003 Confederated International Conferences DOA, CoopIS and ODBASE, Proceedings. LNCS, Springer (2003)
7. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition 5(2), 199–220 (1993)
8. Hendler, J.: On beyond ontology: What's next for the semantic web? Keynote Talk at the 2nd International Semantic Web Conference (ISWC'03) (2003)
9. Henson, C.A., Pschorr, J.K., Sheth, A.P., Thirunarayan, K.: International Symposium on Collaborative Technologies and Systems (CTS 2009), chap. SemSOS: Semantic Sensor Observation Service (2009)
10. Hornsby, K., King, K.: Linking geosensor network data and ontologies to support transportation modeling. In: Nittel, S., Labrinidis, A., Stefanidis, A. (eds.) GeoSensor Networks: Second International Conference, GSN 2006. pp. 191–209. Springer (2008)
11. Jirka, S., Bröring, A., Stasch, C.: Applying OGC Sensor Web Enablement to Risk Monitoring and Disaster Management. In: GSDI 11 World Conference, Rotterdam, Netherlands (June 2009)
12. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. J. ACM 42(4), 741–843 (1995)
13. Li, L., Horrocks, I.: A software fframework for matchmaking based on semantic web technology. In: 12th International Conference on the World Wide Web (WWW'03) (2003)

14. Lutz, M.: Ontology-based descriptions for semantic discovery and composition of geoprocessing services. Geoinformatica (2006)
15. Lutz, M., Klien, E.: Ontology-based retrieval of geographic information. International Journal of Geographic Information Science 20(3), 233–260 (2005)
16. Masolo, C., Guarino, N., Oltramari, A., Shneider, L.: The WonderWeb library of foundational ontologies. Tech. rep. (2003)
17. McGuinness, D., van Harmelen, F.: Web Ontology Language (OWL) Overview. http://www.w3.org/TR/owl-features/ (Feb 2004), w3C Recommendation
18. Na, Arthur Priest, M.: Sensor Observation Service - implementation specification version 1.0. Tech. Rep. 06-009r6, Open Geospatial Consortium (2007)
19. Probst, F.: Semantic reference systems for observations and measurements. PhD Dissertation (2007)
20. Raape, U., Teßmann, S., Wytzisk, A., Steinmetz, T., Wnuk, M., Hunold, M., Strobl, C., Stasch, C., Walkowski, A.C., Meyer, O., Jirka, S.: Decision support for tsunami early warning in indonesia: The role of standards. In: Cartography and Geoinformatics for Early Warning and Emergency Management (2009)
21. Raskin, R., Pan, M.: Knowledge representation in the semantic web for earth and environmental terminology (SWEET). Computers and Geosciences 31(9), 1119–1125 (2005)
22. Vemmer, T.: The Management of Mass Casualty Incidends in Germany - From Ramstein to Eschede. BoD (2004)