Baseline for Registering and Annotating Geodata in a Semantic Web Service Framework[1]

Eva Klien, Daniel Ingo Fitzner, Patrick Maué
Institute for Geoinformatics, University of Muenster

INTRODUCTION

Spatial Data Infrastructures (SDIs) provide access, reuse and integration of geographic information from multiple sources. However, this potential can only be fully exploited if a) geographic information registered in the SDI reaches a critical mass, and b) this information is 'findable'. Finding what you need is one of the crucial factors for open environments to become successful. However, current users of SDIs still face inefficient and frustrating search experiences. To improve the situation, we have to establish new ways for both, advertising and finding information resources in SDIs.

An SDI is characterized by the great variety both in its users and in its information sources. Searching for information is a communication process between the information requestor and the information provider, only that they are not interacting directly but through the query tool, which acts as broker for negotiating between them. Problems of different conceptualizations, terminology and missing information are easily resolved in a personal communication process, while a keyword-based search algorithm remains quite ignorant to these aspects. Search functionalities that account for semantic heterogeneities could improve the situation. A promising development in this direction is the combination of technologies around the Semantic Web with the standardization achievements in the geospatial community. Introducing formal descriptions of information sources written in logic opens possibilities for more sophisticated query processing, e.g. matchmaking based on logic reasoning. The usefulness of integrating such ontology-based discovery into information systems to overcome semantic heterogeneities has been shown (Lutz and Klien, 2006). However, the approach relies on the availability of formal descriptions of the information sources written in logic calculus.

In this paper, we present a strategy for registering geodata in a semantic web service (SWS) framework, which produces formal descriptions. The strategy involves the automatic transformation of service descriptions and data schemas into a formal representation language, which can be processed within the SWS framework. Based on this foundation, structures that are more sophisticated can be added to the service descriptions by generating semantic annotations (understood as mappings from elements of the data schema to elements in a domain ontology). This will enable querying not only on syntax, but also on meaning.

We first introduce the basic elements of the SWS framework and the formal language used in the presented approach to the extent needed for understanding the examples. The subsequent section provides an analysis of the infrastructure needed to enable automatic transformations from service descriptions and data schemas into the components of the SWS framework. We also illustrate the need for semantic annotation to ensure that we do not only use the correct formalisms but also capture the semantics of the registered data sets. With the help of a specific example, a walk-through showing how to register an OGC Web Feature Service (WFS) in the SWS framework illustrates the implementation. The benefits of having formal descriptions available are shown with a simple discovery scenario. The last section concludes and gives some thoughts on future work.

[1] This work is funded by the European Commission under the *SWING* project (FP6-26514)

THE SEMANTIC WEB SERVICE FRAMEWORK

Semantic web services are meant to enrich web services with machine-processable semantics. There are several developments in the area of SWS, which are documented in (Cabral et al., 2004). In the following, we introduce WSMO (Roman et al., 2005), which has been developed as a conceptual model for semantically describing all relevant aspects of web services in order to facilitate the automatisation of discovering, combining and invoking services over the Web. The WSMX execution environment is a reference implementation of WSMO. The Web Service Modeling Language (WSML) is the internal language of WSMX.

The Web Service Modeling Ontology

WSMO identifies ONTOLOGIES, WEBSERVICES, GOALS and MEDIATORS as the four top-level elements, for defining SWS (Roman et al., 2005). In our work on registration and semantic annotation of geodata, we deal with the WSMO elements ONTOLOGIES and WEBSERVICES. In addition, we need GOALS in the discovery scenario.

ONTOLOGIES provide the terminology used by other WSMO elements to describe the relevant aspects of the domains of discourse. Ontologies capture and formalize the meaning of the described components. Moreover, the formal definitions are machine-processable and thus allow sophisticated information processing based on logic reasoning. WEBSERVICES represent computational entities able to provide access to services that, in turn, provide some value in a domain; a WEBSERVICE comprises the capabilities, interfaces and internal working of a service. All these aspects are specified using the terminology defined in the ONTOLOGIES. Capabilities characterize the web service's state before and after an execution by specifying pre- and postconditions. GOALS describe aspects related to user desires with respect to the requested functionality; again, ONTOLOGIES can be used to define the used domain terminology, useful in describing the relevant aspects of GOALS.

The Web Service Modeling Language

WSML has been designed for writing down descriptions of WEBSERVICES, GOALS, ONTOLOGIES, and (to some extent) MEDIATORS (de Bruijn, 2005). WSML is a family of formal description languages used for the precise specification of the elements in the WSMO framework. The different variants of WSML (WSML-Core, WSML-Flight, WSML-Rule, WSML-DL, and WSML-Full) correspond to different logical language paradigms, namely Description Logic, Logic Programming and First-Order Logic. All of them are specified in terms of a general WSML-syntax, but each might impose different restrictions on certain syntactic elements of the language. The general WSML-syntax mainly consists of two parts: the conceptual syntax and the logical expression syntax. The conceptual syntax is a frame-like syntax with constructs like concepts, attributes, relations and instances. The logical expression syntax is used for further refinement of concept- or relation-definitions in the conceptual syntax.

We show examples in WSML syntax in the section "Walk-through for registering and annotating WFS in WSMO". We use WSML-Flight because this variant has proven to serve best our application's requirements regarding expressivity of language and reasoning capabilities.

APPROACH FOR REGISTERING AND ANNOTATING GEODATA

In OGC-compliant SDIs, geodata are served via WFS. In a standard Catalogue, users register WFS by providing metadata (e.g. ISO 19115) on the service and the data it serves. We suggest a new way for

registering WFS, which automates the registration process and generates formalized service descriptions usable for further information processing in a SWS framework.

The goal of the registration process is to generate a WSMO WEBSERVICE for a specific WFS that integrates information on the service functionality, on the geodata encoding, and on the semantics of the data that is served. The following two steps are required as part of this process: First, we execute an automatic transformation of the WFS descriptions into WSMO WEBSERVICE and WSMO ONTOLOGY constructs. The result of the automatic transformation cannot contain more information than the original documents - it is merely a translation into a different syntactic representation. So, how can we ensure that we do not only use the correct syntax, but also capture the semantics of the registered data sets? To satisfy this requirement, the second step of the registration process involves the semantic annotation of feature types served by the WFS by mapping elements of the feature type schema to concepts in domain ontologies.

First Step: Approach for Automatic Transformation

Figure 1 depicts the items involved in the process of automatically transforming WFS descriptions into WSMO WEBSERVICE and ONTOLOGY constructs. Subject to annotation are spatial information objects that model real world entities. These spatial information objects are represented as features in the Geographic Markup Language (GML) and served via WFS. Information on the service is accessible via its standardized operations GetCapabilities (returning a description of the service's capabilities) and DescribeFeatureType (returning the application schemas of the feature types served by the WFS).

Both service descriptions (WFS capabilities and feature type schema) are parsed with the help of a third party library[2]. During the transformation, rules are applied to reference the parsed information to the corresponding concepts in *OGC domain ontologies*. For this purpose, two OGC ontologies encoded in WSML are available in our environment: the WFS ONTOLOGY captures the service implementation rules for WFS as specified in the OGC WFS Implementation Specification (OGC, 2002) and the GML ONTOLOGY captures the encoding rules for features as specified in the OGC GML Encoding Specification (OGC, 2003). The WEBSERVICE and ONTOLOGY constructs that formally describe a specific WFS in WSMO are generated on-the-fly. This automatic translation into WSML consists of two parts:

a) **Translation of feature type schema:** For every feature type listed in the capabilities document, the algorithm creates the corresponding WSML representation. Those elements of the schemas that point to GML encodings (e.g. the geometry attribute) are referenced to corresponding concepts in the GML ONTOLOGY. The result of this translation step is a FEATURE TYPE ONTOLOGY (FTO) providing structural information of all feature types served by the specific WFS.

b) **Translation of service capabilities:** WFS are services that provide standardized operations for data access. The basic functionality (i.e. retrieving features) is the same for all WFS implementations. It is thus possible to use the generic WFS ONTOLOGY to define the WFS capabilities. The translation process further refines the description with specific information from the WFS to be registered. For example, with the help of the postcondition it is possible to constrain the output of the GetFeature operation to the features the service is actually serving. By referencing the features in the postcondition to the corresponding concepts in the FTO, it is possible to assess more detailed information on the feature type schema. The result of this translation step is a specific WFS WEBSERVICE.

[2] We are using the LGPL library *geotools* (www.geotools.org) for this task.

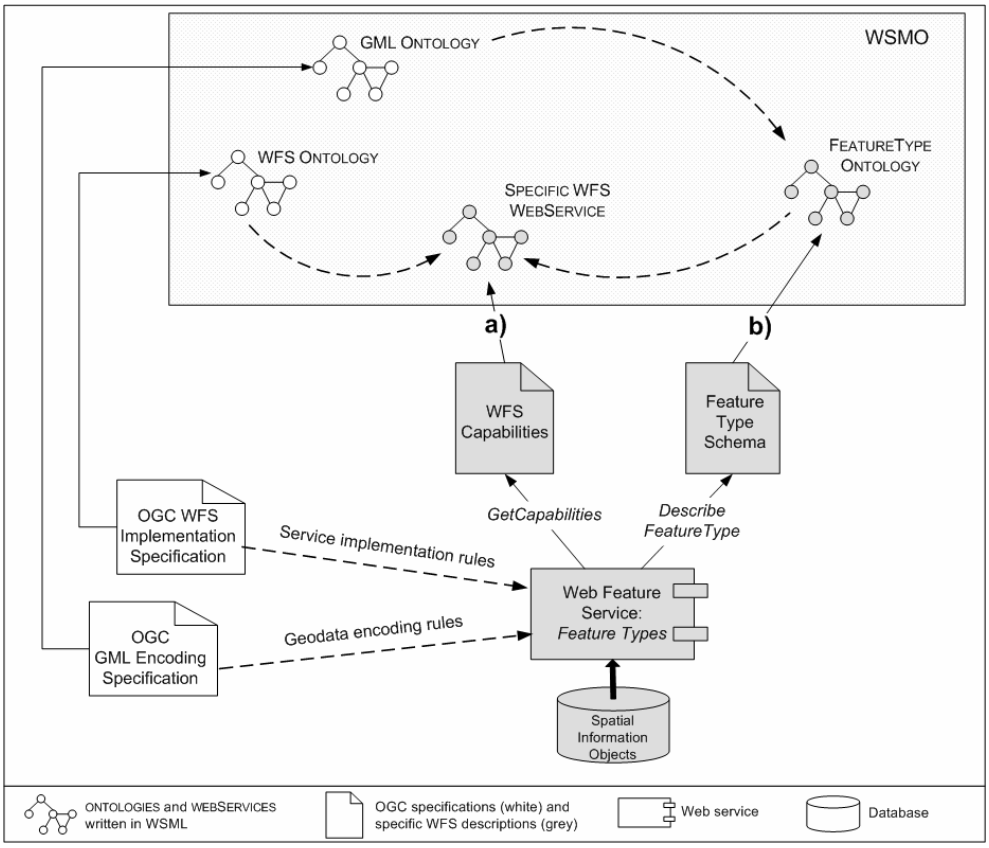

Figure 1: Items involved in registering a WFS in the WSMO framework.

At this stage, the automatic transformation has produced a specific WFS WEBSERVICE and FTO written in WSML, which do not contain more information than the original WFS descriptions. The crucial part is now to capture and explicate the meaning of the retrievable data.

Second Step: Approach for Semantic Annotation

Figure 2 depicts a schematic representation of a domain ontology (DO) on *Quarries* (sites, where mineral resources are produced or mined). This domain ontology has been developed in cooperation with the Bureau de recherches géologiques et minières (BRGM) as part of the work carried out in the SWING project[3]. DO's are developed to capture the conceptualization of a specific view on the world and formalize it in concept definitions. It is assumed that all members of the community will interpret the terminology used in their domain ontology in the same way and, at the same time, people from outside the community are able to explore the intended meaning with the help of the concept definitions.

[3] http://www.swing-project.org

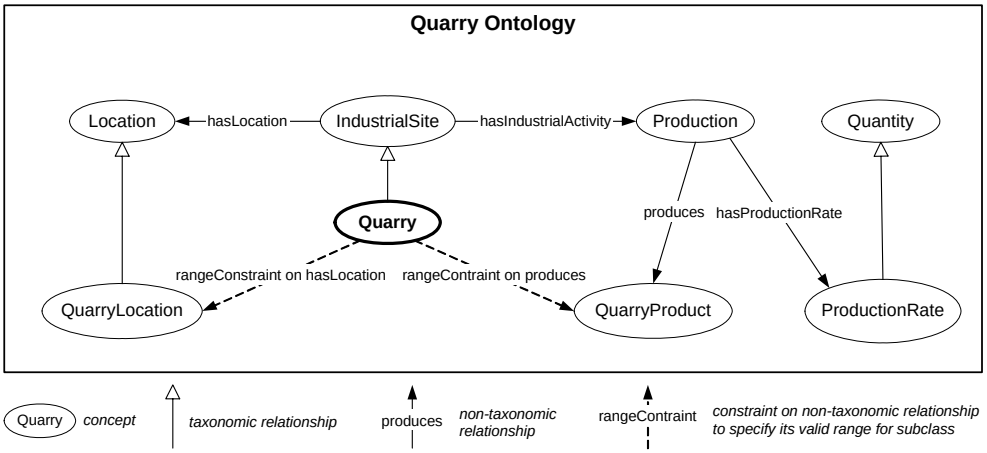

Figure 2: Schematic representation of an extract from the Quarry Ontology.

Quarry is the central concept of the QUARRY ONTOLOGY. It is defined as subconcept of *IndustrialSite*, which means that *Quarry* inherits all relationships that have already been defined for *IndustrialSite*. Some of the relationships require further restrictions. The range of *hasLocation* points to *QuarryLocation*, which is a subconcept of *Location*, and the *Production produces* not any Product, but *QuarryProduct*. Again, these concepts are further defined by adding or constraining their non-taxonomic relationships.

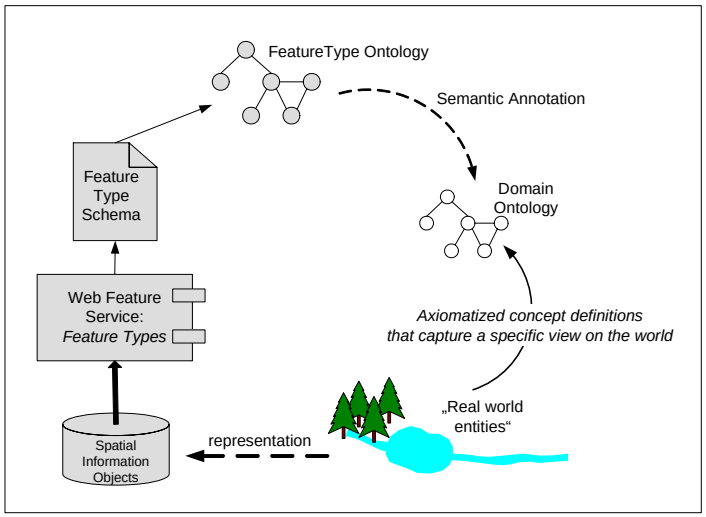

Figure 3: Setting for the semantic annotation of geodata (from (Klien, 2007)).

In order to generate semantic annotations for the concepts defined in the FTO, the data provider has to define mappings between concepts in the FTO and concepts in the DO. The tricky part in this endeavor is that FTO and DO do not capture the same kind of extensions – the FTO reflects the data schema for features, i.e. spatial information objects – whereas concepts in the DO capture the

meaning of real-world entities (as conceptualized by a specific user community). The mappings between FTO and DO cannot be defined via taxonomic *is-a* relationships because both ontologies represent different kind of entities. We have thus decided to introduce a non-taxonomic *annotate* relation into our environment, which explicitly expresses the fact that instances of one concept convey the same meaning, but are not necessarily of the same kind as the instances of the concept they are annotated with.

In order to identify semantic annotations, we have to answer the following questions: "*does a spatial information object convey the same meaning as a concept of the domain ontology?*". This question can only be answered, if we can detect that the spatial information object represents a geographic entity, which in turn would be classified as instance of a domain concept (Klien, 2007). Metadata (e.g. ISO 19115) and feature type schema often do not reveal sufficient explicit information, so that existing automatic mapping techniques would not be supportive for this task (please refer to Klien (2007) for discussion on existing mapping techniques). For example, string-based matching of terms is not applicable (*fto:exploitationponctualproduction* will not match *domain:Quarry*, see example in the next section). In addition, the structure of the feature type and the structure of the domain concept are not comparable. Currently, only the data provider who is familiar with the spatial information objects can infer this kind of instance relationship and the mappings from FTO to DO have to be performed manually.

WALK-THROUGH FOR REGISTERING AND ANNOTATING WFS IN WSMO

BRGM advertises a WFS that provides data on quarries together with related information like beds or basins in France (see: http://swing.brgm.fr/). The QuarryWFS serves six different feature types, namely exploitationsboundaries, exploitationsponctuals, exploitationsponctualsproduction, beds, sites and basins. The term "exploitation" is a synonym to the term "quarries"; the service thus offers three feature types with information about quarries. To deal with a simple example, we will only consider the feature type "exploitationsponctualsproduction" and two of its attributes, namely "msGeometry" (the point geometry of the quarries) and "allowedProduction" (the maximum production of the quarry in tons per year).

First Step: Automatic Generation of QuarryWFS descriptions

a) **Translation of feature type schema**: Invoking the DescribeFeatureType operation on the QuarryWFS with a specific feature type id returns the feature type schema. Figure 4 shows how the feature type "exploitationsponctualsproduction" *(a)* can be translated into a FTO written in WSML (b), including references to concepts of the GML ONTOLOGY like *gml#Feature* and *gml#GeometryPropertyType*. Due to the standardization of both syntaxes, the translation is non-ambiguous and therefore straightforward.

```
<element name="exploitationsponctualsproduction" type="qua:exploitationsponctualsproductionType"
                                                    substitutionGroup="gml:_Feature" />
 <complexType name="exploitationsponctualsproductionType">
  <complexContent>
  <extension base="gml:AbstractFeatureType">
   <sequence>
    <element name="msGeometry" type="gml:GeometryPropertyType“ />
    <element name="ExploitationName" type="string" />
    <element name="Communities" type="string" />
    <element name="Substance" type="string" />
    <element name="Year" type="string" />
    <element name="AllowedProduction" type="string" />
    <element name="SiteName" type="string" />
    <element name="SiteType" type="string" />
   </sequence>
   </extension>
  </complexContent>
 </complexType>                                                              a)
```

```
concept exploitationsponctualsproduction subConceptOf gml#Feature
 msgeometry impliesType (1 1) gml#GeometryPropertyType
 exploitationname impliesType (1 1) _string
 communities impliesType (1 1) _string
 substance impliesType (1 1) _string
 year impliesType (1 1) _string
 allowedproduction impliesType (1 1) _string
 sitename impliesType (1 1) _string
 sitetype impliesType (1 1) _string                                          b)
```

Figure 4: Transforming the feature type schema (a) into the FTO written in WSML (b).

b) Translation of service capabilities: Invoking the operation GetCapabilities on the QuarryWFS returns the capabilities document, which – among other metadata – provides a specification of the service's operations. The automatic translation generates a specific WSMO WEBSERVICE for the QuarryWFS. In this example, we keep the specification simple and concentrate on the postcondition of the GetFeature operation that is restricted to features of type "exploitationsponctualproduction". This restriction is specified by reference to the concept *fto#exploitationsponctualproduction* as defined in the FTO (Figure 5).

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
namespace { _"http://swing-project.org/webservice/brgmWFS#",
     fto _"http://swing-project.org/ontologies/brgmFTO#" }

webService brgmWFS

capability brgmWFS_capability

postcondition getFeature_postcondition
   nonFunctionalProperties
     dc#description hasValue "The Service returns features of the types
     that are further specified in the BRGM FeatureType Ontology"
   endNonFunctionalProperties

   definedBy
   ?features memberOf fto#exploitationsponctualsproduction.
```

Figure 5: WSMO WEBSERVICE for QuarryWFS.

Second Step: Semantic Annotation of Feature Types

The feature type "exploitationsponctualproduction" denotes point objects that model quarries (real-world geographic entities). In turn, real world quarries are described in the concept definition of *domain#Quarry*, which means that theoretically they would be classified as instances of the domain concept *domain:Quarry*. The feature type's attributes refer either to the information object (e.g. "msgeometry") or to the geographic entity (e.g. "allowedProduction"). Those attributes that describe the information object have been referenced to the GML ONTOLOGY in the previous automatic translation step. Attributes referring to the geographic entity have to be semantically annotated with concepts from the QUARRY ONTOLOGY. In our example, "allowedProduction" is mapped to the domain concept *domain:ProductionRate* (Figure 6).

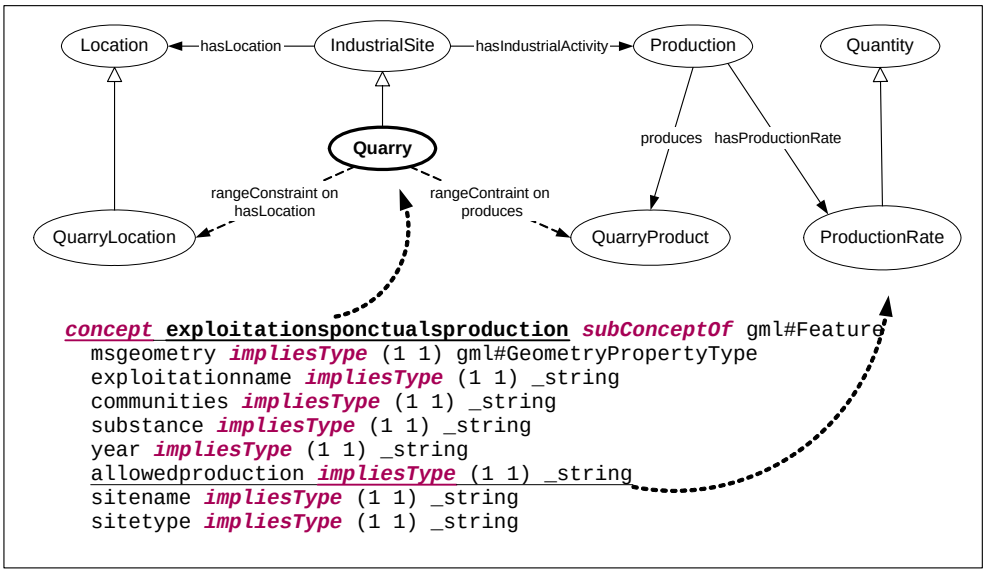

Figure 6: Two elements of the feature type schema mapped to domain concepts.

Once the data provider has defined the mappings, they are formalized in WSML axioms and added to the FTO. Figure 7 depicts the formalized semantic annotations for the feature type "exploitationsponctualproduction" and its attribute "allowedProduction". Remember, FTO and DO are two different "sides" in the framework that describe different kinds of entities and it would be awkward to define taxonomic relationships between their instances. Thus, we realize the semantic annotation bridge by a binary predicate ("annotate") connecting concept names. The axiom *defineAnnotation* lists all *annotate* relations that might exist between predicates in the FTO and concepts in the DO.

```
axiom defineAnnotation
definedBy
annotate(exploitationsponctualsproduction, domain#Quarry).
annotate(allowedproduction, domain#ProductionRate).
```

Figure 7: Formalized semantic annotations.

.

BENEFITS

To illustrate the benefits of having formal service and data descriptions available in WSMO, we introduce a simple discovery scenario. A user is looking for information on the production rate of quarries in a specific area of France. The search interface of the SWING application allows the user to formulate this question by utilizing the vocabulary specified in the domain ontologies (QUARRY ONTOLOGY, GML ONTOLOGY, WFS ONTOLOGY). A simple example is the following question: "Find me all services that provide Quarry features that contain information on production rates". This query is translated into WSML[4] as depicted in Figure 8. The postcondition of the goal's capabilities states that the user is searching for services that provide features, which are semantically annotated with the domain concept *domain:Quarry*. And those features should also have an attribute, which is semantically annotated with the domain concept *domain:ProductionRate*.

The user request is represented in WSML as a conjunctive query. The answer to a query is – as in databases – the set of all tuples of instances that satisfy the query. In a conjunctive query, all constraints are connected by conjunction (logical AND). Since we do not deal with concrete instances (data items) in the discovery scenario, the user request is represented as a so called meta-query, which is – in principle – the same as a query in databases or logic programs, but which does not query for data but for schema information (in our case concept definitions). In this simple discovery scenario, a matchmaking component uses the goal specification (Figure 8) to query the concept definition (*fto#exploitationsponctualproduction*) offered by the service brgmWFS (see Figure 5). Since the advertised concept delivers the requested functionality, i.e. it is a *gml#Feature*, it is semantically annotated with *domain#Quarry* and it has an attribute that is annotated with *domain#ProductionRate*, the user query matches the service.

```
namespace { _"http://swing-project.org/ontologies/userGoal#",
  gml _"http://swing-project.org/ontologies/gml#",
  domain _ "http://swing-project.org/ontologies/quarries#"}

goal userGoal

capability goalCapability

postcondition
  nonFunctionalProperties
    dc#description hasValue "returns features that are semantically annotated
    with the domain concept Quarry and that have an attribute that is
    semantically annotated with the domain concept ProductionRate"
  endNonFunctionalProperties
  definedBy
    ?x memberOf ?C and
    ?C subConceptOf gml#Feature and
    annotate(?C,?y) and
    ?y subConceptOf domain#Quarry and
    (?C[?r impliesType ?T]) and
    annotate(?r, ?RC) and ?RC subConceptOf domain#ProductionRate.
```

Figure 8: User GOAL written in WSML.

CONCLUSION / FUTURE WORK

We have presented a strategy for registering and annotating geodata in a SWS framework. The availability of formal service and data descriptions enables sophisticated information processing

[4] The automatic translation of a user query into WSML has not been implemented yet

within the SWS framework. In this paper, we have shown the benefits for discovery, as logic-based reasoning allows query processing not only on syntax, but also on meaning. So far, it is only possible to define these semantic annotations manually. Automating the process with existing ontology mapping techniques like text and schema comparison is not reliable. Future work will concentrate on methods that allow to automatically infer a feature type's meaning, by exploring implicit information hidden in text documents or instance data. One proposal in this direction is a method for detecting class membership by analyzing geometry and topology in geospatial datasets (Klien, 2007). In the case of WFS the focus is on the content level, i.e. what data can be retrieved. But combining SWS and OGC services with the goal of realizing composition of geospatial services needs further work on formalizing processing semantics (Lutz, 2006). The question of how to annotate geoprocessing services will be addressed at a later stage of the SWING project.

REFERENCES

Cabral L., Domingue J., Motta E., Payne T.R., and Hakimpour F. 2004 Approaches to Semantic Web Services: An Overview and Comparisons. In *Proceedings of the 1st European Semantic Web Symposium (ESWS2004)*. Heraklion, Crete, Greece

de Bruijn J. (Ed.) 2005 The Web Service Modeling Language WSML. http://www.wsmo.org/TR/d16/d16.1/v0.2/

Klien E. 2007 A rule-based strategy for the semantic annotation of geodata. *Transactions in GIS, Special Issue on the Geospatial Semantic Web*, forthcoming

Lutz M. 2006 Ontology-Based Descriptions for Semantic Discovery and Composition of Geoprocessing Services. *GeoInformatica,* forthcoming

Lutz M., and Klien E. 2006 Ontology-Based Retrieval of Geographic Information. *International Journal of Geographical Information Science (IJGIS) 20*(3): 233-260

OGC 2002 *Web Feature Service Implementation Specification, Version 1.0.0*. Open Geospatial Consortium

OGC 2003 *Geography Markup Language (GML) Implementation Specification, Version 3.0*. Open Geospatial Consortium

Roman D., Keller U., Lausen H., de Bruijn J., Lara R., Stollberg M., et al. 2005 Web Service Modeling Ontology. *Applied Ontology 1*(1): 77-106