Semantic access to INSPIRE
How to publish and query advanced GML data

Sven Tschirner[1], Ansgar Scherp[2], and Steffen Staab[2]

[1] Federal institute of hydrology, Koblenz, Germany
tschirner@bafg.de
[2] Institute for Computer Science
University of Koblenz-Landau, Koblenz, Germany
<surname>@uni-koblenz.de

Abstract. The INSPIRE Directive establishes a pan-European ”Spatial Data Infrastructure” (SDI) to make available multiple thematic datasets from the EU member states through stable Geo Web-Services. Parallel to this ongoing procedure, the Semantic Web has technologically fostered the Linked Data initiative which builds up huge repositories of freely collected data for public access. Querying both data categories within distributed searches looks promising. To tackle the associated prerequisites, this paper presents firstly a general approach to translate sophisticated INSPIRE GML data models into Semantic Web OWL ontologies. This is done according to Linked Data principles while preserving selective INSPIRE structural information as annotations. Secondly, a feasible conversion of the Semantic Web query language SPARQL to its Geo Web counterpart ”OGC Filter Encoding” is proposed. The language mapping is required for a semantic wrapper over remote INSPIRE Download Services acting as a SPARQL-endpoint and bridging the gap between both worlds.

Keywords: INSPIRE, Linked Open Data, Semantic Web, SPARQL, Geo Web, GML, geospatial data, semantic enablement

1 Introduction

The INSPIRE Directive (2007/2/EC)[3] obliges national authorities of the EU-member states to contribute their spatial data according to over 30 harmonized themes (e.g. Hydrography, Protected Sites or Elevation), make them accessible and described via standardized Geo Web-Services. These datasets are considered to be up-to-date, quite reliably, EU-wide and mostly free available forming a very impressive data source for multi-thematic information retrieval.

Because of its free data usage and distributed service-architecture INSPIRE does have a lot in common with the Linked Open Data initiative (LOD). Combining both data worlds while starting federated searches over INSPIRE and LOD-datasets looks attractive as repositories differ in thematic coverage and

[3] http://inspire.jrc.ec.europa.eu/

data capture conditions. As a scenario let us consider a teacher planning a school trip which should be filled with leisure activities as well as some kind of nature exploration. He find accomodation and leisure facilities through Geo Linked Data[4] and spatially overlay them with INSPIRE themes Land Use or Protected Sites. Finally the teacher may prepare himself with reliable background information referenced by INSPIRE data and disseminated by public authorities enriching the world of free data. The same governmental controlled Protected Sites data could be combined with the LOD project GeoSpecies[5] to intersect INSPIRE protection goals with species populations from GeoSpecies at the same time.

To leverage such scenarios, the existing technological discrepancies between INSPIRE and LOD have to be overcome which are mainly due to different knowledge representation and web service interfaces. The goal is a feasible embedding of INSPIRE data in the Semantic Web. INSPIRE itself is based on the Geo Web technologies - which are ISO and OGC[6] standardizations for Geo Web-services and -transfer formats. So INSPIRE applies the ISO/OGC-approach of modeling physical things, so-called "features", in the Geography Markup Language (GML)[7]. GML is a XML-derivative and GML data models are written in XML-Schema Language (XSD). Conventional GML data models define simple and flat XML-documents having features with only few attributes and thus could be easily queried and transformed to Semantic Web RDF/OWL-triples. With regard to INSPIRE we are faced with advanced GML data models which disclose features with many dependant complex elements and a heavily nested, verbose XML-tree structure. That's why transforming GML to OWL is not straight-forward and target OWL-models in which to transform INSPIRE data must be well-thought-of for not generating triples without proper content.

Former work [3] [13] have already introduced Semantic Web-queries which are translated in order to request OGC data access services - specified as OGC WFS[8]. The conversion of GML-results is done on-the-fly which is a reasonable way to avoid duplicated storage of GML and RDF/OWL-instances and facilitates the access to up-to-date information. But these approaches are concerned either with flat GML-models and so encompass a query translation which is much simplier than INSPIRE requires where XPath-expressions for filter processes became inevitably. Or they don't take into account that the content of the GML→OWL transformation must be refined in order to fullfill Linked Data principles with resolvable URIs and cross-references.

This paper presents a feasible way to perform SPARQL queries on INSPIRE which contains two main achievements. At first we propose a general approach for deriving INSPIRE ontologies from the INSPIRE UML/GML data models in order to define the target models for SPARQL querying. We also suggest common modeling aspects and refinements for Semantic Web-representations

[4] `http://linkedgeodata.org/About`

[5] `http://about.geospecies.org/`

[6] Open Geospatial Consortium, see `http://www.opengeospatial.org/`

[7] `http://www.opengeospatial.org/standards/gml`

[8] `http://www.opengeospatial.org/standards/wfs`

due to Linked Data principles. Secondly we outline a SPARQL-endpoint - which is configured with these INSPIRE ontologies and acting as a proxy over WFS-services which are the main realizations for INSPIRE Download Services. Therefore we specify a viable translation from SPARQL to the WFS-query language ”OGC Filter Encoding (FE)”[9] and tackle the prerequisite of references between the INSPIRE ontology concepts and the INSPIRE GML data structure.

2 Towards INSPIRE Linked Data - Basic Architecture

To fulfill the overall goal for federated queries on INSPIRE and Linked Data, the INSPIRE data must be transformed into a Semantic Web representation as Semantic Web-formats are appropriate for integration tasks and SPARQL used to query different repositories in particular. Two key tasks are identified: 1. we need to create common target ontologies for INSPIRE themes and 2. provide query capabilities using SPARQL. INSPIRE UML- and derived GML-data models are regarded as community harmonizations and thus have the same characteristics as Semantic Web domain ontologies. Hence each INSPIRE theme is syntactically transferred to one domain ontology (here called ”INSPIRE theme ontology”) while preserving INSPIRE element names. Common concepts from different themes, e.g. ISO metadata elements and spatial representations, are outsourced into core ontologies and imported into the theme ontologies if needed.

The reasons for explicit INSPIRE ontologies are:

- they serve as target models for a proper GML→OWL conversion set-up
- they store GML structural information as annotations needed for querying
- they facilitate references to INSPIRE background information
- they prepare the basis for alignment with Semantic Web upper ontologies

Querying capabilities help to perform requests and filter the desired instances from the requested repository. A smart way to support SPARQL queries over INSPIRE data can be accomplished with a proxy application, wrapping remote or local INSPIRE Download Services and acting as a SPARQL-endpoint; see Fig. 1. The architecture has multiple advantages. For instance, no data replication and additional storage facilities are needed, hence queries always access up-to-date datasets. Furthermore, INSPIRE data authorities are not bothered with extra configuration efforts required for semantic WFS-profiles.

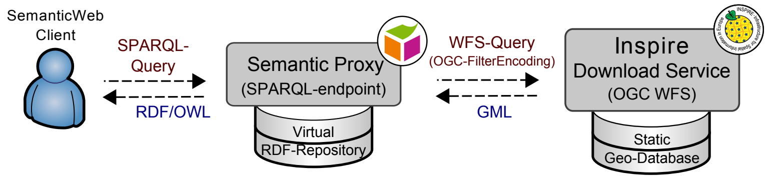

Fig. 1. Overall architecture - Proxy wrapping INSPIRE-Download Services

[9] http://www.opengeospatial.org/standards/filter

Opposed to alternative solutions which comprise one single GML→OWL-transfer into a static INSPIRE triple-store building query facilities on top, the proxy solution has to cope with a virtual repository. That means that all resulting information is temporary and has to be combined with all side effects. The biggest challenge is the query mapping from SPARQL as a powerful general query language to the less powerful language OGC Filter Encoding. The latter is supported by WFS-services and focuses on domain-optimized filters.

We first introduce our proposed query strategy and more details will follow in Sec. 4. Fig. 2 shows the overall query procedure. A SPARQL query is received (1) and converted to SPARQL algebraic expression[10] (2). The algebraic expression is assessed for INSPIRE concepts (3) for which XPath-expressions are concatenated to address the GML-element paths needed within Filter Encoding-query operators. Thereafter the translation SPARQL →Filter Encoding takes place (4) and the WFS GetFeature-request is send to all the WFS-services which serve the requested feature types (5). On return the GML-results are transformed to OWL-instances composing the virtual repository (6). This virtual repository is then queried with the former SPARQL-query from the second step (7) and finally SPARQL-results are returned to the user (8).

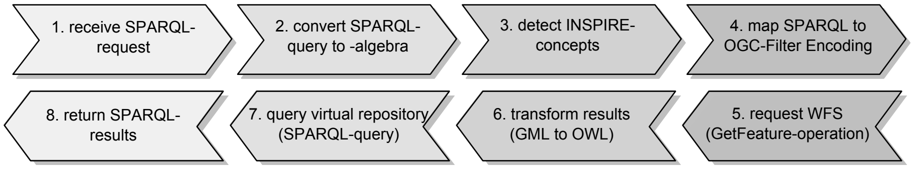
1. receive SPARQL-request
2. convert SPARQL-query to -algebra
3. detect INSPIRE-concepts
4. map SPARQL to OGC-Filter Encoding
8. return SPARQL-results
7. query virtual repository (SPARQL-query)
6. transform results (GML to OWL)
5. request WFS (GetFeature-operation)

Fig. 2. SPARQL-query handling in eight steps

The resulting INSPIRE ontologies are designed to allow custom queries, for instance to filter Protected Sites and GeoSpecies data. Therefore we outline significant query types which are regarded during ontology modeling (see Sec. 3.2) in order to keep ontological structures simple and adequate for querying:

- distinguish instancies by its classification attributes, e.g. ”which protected sites are classified with designation = 'UNESCO World Heritage'?”
- do spatial reasoning, e.g. ”which species habitats crosses protected site x?”
- filter temporally, e.g. ”which sites originates from era x, between dates y/z?”
- assess measures, e.g. ”which sites suit a hike, sites greater than x km^2?”

This paper continues in Sec. 3 with a description of engineering and modeling aspects of the proposed INSPIRE ontologies. Sec. 4 is about the concept of a semantic query layer and a prototype mentioned in Sec. 5. Then, Sec. 6 outlines related work and Sec. 7 closes with conclusions and possible advancements.

[10] http://www.w3.org/TR/rdf-sparql-query/; see Sec. 12.4 ”SPARQL Algebra”

3 Deriving INSPIRE ontologies

3.1 Conversion rules: INSPIRE GML to RDF/OWL

Before examining ontological details, we have to consider the basic conversion rules from the original GML INSPIRE data models (written in XSD) to the targeted INSPIRE OWL ontologies. For generic XSD→OWL transformations several approaches [1] [2] and operational applications exist. Accordingly conversion rules are defined to transform either XML- or XSD-documents to OWL, converting e.g. every `xsd:complexType` to `owl:Class`. These rules are helpful for deriving INSPIRE ontologies but don't regard GML or INSPIRE specifics.

GML and RDF/OWL - common characteristics Having a closer look at GML itself which is akin to RDF/OWL [11], as GML-Version 1 had once an explicit RDF-encoding. Since then GML carries the "Object-Property-pattern". This means that XML-elements at odd numbered levels of the DOM-hierarchy represent GML-objects and ones at even level represent GML-properties. Hence one may compare GML-objects to RDF-resources and GML-properties to RDF-predicates. Another similarity is the cross-referencing, which is often used in INSPIRE for cross-thematic linking, and the identification of GML-objects. These basic mappings between GML and RDF are 1. GML-attribute for cross-referencing `xlink:href` which equals RDF-attribute `rdf:resource` and 2. GML-object identifier attribute `gml:identifier` which equals RDF-attribute `rdf:about`. While the Object-Property-pattern is crucial for engineering the theme ontology, the common linking and identification is used during OWL-instance generation.

INSPIRE UML-notations - leading to main conversion rules The UML-models disclose the GML Object-Property-pattern and give further orientation for ontology engineering with UML-class stereotypes and UML-associations as cross-references. The prominent UML-stereotypes are `featureType` (= a WFS record type), `dataType` (= a complex type dependant of `featureType`), `codelist` and `enumeration` (both key-value list-types, open for new values or a closed list respectively) and `union` (semantically equivalent to the XSD union type).

Given these modeling hints, we can list our few main rules which imply that derived OWL-concepts are named after corresponding INSPIRE element types:

- every INSPIRE UML-class except stereotype `union` is converted to an `owl:Class`. Subtypes of stereotype `union` are modeled each as one `owl:Class`
- every value of `codelist` or `enumeration` is converted to an `owl:Individual` typed with the corresponding enumeration/codelist `owl:Class`
- every UML-attribute corresponding to a GML-property is converted to an `owl:ObjectProperty` or `owl:DatatypeProperty`. If multiple, equally-named GML-properties lead to both OWL property types we name the `owl:DatatypeProperty` with suffix `_dataValue` to be conform to OWL-DL
- every UML-association is converted to an `owl:ObjectProperty`

3.2 Modeling aspects of frequent elements

Besides these general rules there have to be ontological refinements for frequently used element types which may leverage usual query types listed in the Sec. 2. This section will examine these aspects in more detail. Furthermore we present our considerations about OWL-instance identification and infrastructural opportunities with INSPIRE reference data.

Classifications with codelist- and enumeration-types In INSPIRE data models every sixth element is of such a type which are best suited to differ and filter instances. We model every codelist-type as one `owl:Class` and every codelist-value as one `owl:Individual`. Enumeration-types are equally modeled. However there is a constraint `owl:oneOf` to their enumeration values.

Modeling this way allows us to investigate every codelist-value of a certain codelist, e.g. `ont_ps:ProtectionClassificationValue` with SPARQL triple: `?x rdf:type ont_ps:ProtectionClassificationValue` and opens the possibility to annotate codelist-values with standard-annotations, e.g. `rdfs:label`. Codelist-value individuals are simply named after their codelist:
e.g. the codelist `<http://inspire.ex.org/ProtectionClassificationValue>` has a value `<http://inspire.ex.org/ProtectionClassificationValue/archaeological>`. Hence no naming conflicts arise if two different codelists might have the same value names.

Geometries Concerning geometry handling, there is a current specification development called GeoSPARQL [10]. This state-of-the-art approach includes definitions for a unique geo-vocabulary for which SPARQL extended-functions for extensive spatial reasoning are defined, including topological and other geometrical functions. We extensively make use of this approach to serialize GML geometries resulting from WFS-queries. The proposed SPARQL proxy should provide corresponding GeoSPARQL-spatial filter functions.

Temporal values The conversion of INSPIRE temporal values to RDF/OWL is simple, since GML makes use of XSD-datatypes `xsd:date` or `xsd:dateTime`, too. Mostly resulting RDF typed literals are appropriate. Only in such cases where temporal values are coherent (e.g. start/end-times of a duration) complex RDF-resources have to be derived to retain the coherence and enwrap two or more typed literals for the actual temporal information (e.g. start/end-times).

Measure values Measure values, e.g. the GML-types `Area`, `Length` or `Velocity`, consist of numeric values and corresponding units of measurement (UOM). The encoding in RDF could be a `rdf:PlainLiteral`, e.g. `"200 km2"`, or a typed literal, such as `"200"^^http://www.ex.org/units/km2`. The first option merges the two pieces of information within one textual representation, so that both would have to be discerned during analysis. The drawback of the typed literal would be the definition of additional RDF-datatypes for maybe tens of UOM which should be avoided for interoperability reasons. Our design choice is therefore based on the ”Measure Units Ontology”[11] and shown in Fig. 3 (we use namespaces ’ont_ps’ for the theme ontology ”ProtectedSites” and ’base’ for INSPIRE basic types). The two pieces of information are bundled by a separate base:Measure instance. For numeric values we can simply adopt the XSD-types used in GML (xsd:integer, xsd:decimal etc.) and treat the UOM as an individual RDF-resource. Thus, we are able to filter measure values due to their UOM not only in SPARQL filters but even within SPARQL basic graph pattern.

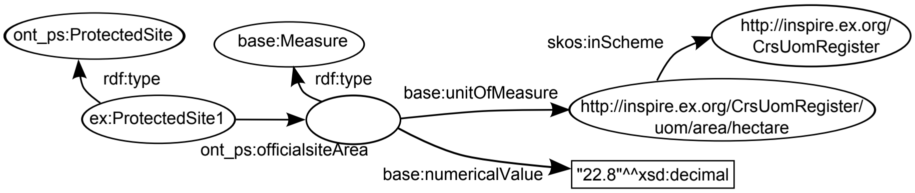

Fig. 3. Using measure values

Registries for reference data Like UOM, other kind of reference data exist, e.g. coordinate reference systems (CRS), languages-codes, country-codes or the INSPIRE codelists/ enumerations. Language-codes are trivial and equally treated in both worlds, INSPIRE and Semantic Web, by using either ISO 639-1/2 or RFC 3066. Other reference data should be explained and collected somewhere online, accessible via resolvable URIs. It makes sense to manage them centrally and in a unique fashion. The OGC has started to make SKOS-concepts[12] of specification elements and some CRS descriptions accessible online[13]. The main technical INSPIRE supporter, the JRC[14] provides SKOS-concepts with the terms of the European GEMET-thesauri and the INSPIRE feature concept dictionary.[15] Such infrastructural measurements will definitely facilitate a Semantic Web-enablement of INSPIRE but also could have positive side-effects for other domains, providing harmonized lists reused e.g. in folksonomies.

Identification As an important Linked Data principle, identifiers for RDF-resources should be kept unique and resolvable via HTTP-URIs. All INSPIRE features are identified with unique INSPIRE-IDs which are used here for individual URIs of OWL-instancies. An INSPIRE-ID consists of 1.) a namespace

[11] http://forge.morfeo-project.org/wiki_en/index.php/Units_of_measurement_ontology

[12] SKOS: Simple Knowledge Organization System; used for semantically modeling taxonomies; http://www.w3.org/TR/skos-primer/

[13] http://www.opengis.net/def/

[14] EU Joint Research Centre, http://www.jrc.ec.europa.eu/

[15] https://semanticlab.jrc.ec.europa.eu/

including details about the data-authority or -product, 2.) a `localId` which is an object identifier unique in the scope of the `namespace` and 3.) a `versionId` for an optional object versioning. These identifying attributes are proposed to be included in an HTTP-URI with additional format information for Linked Data content negotiation, distinguishing the return-format as e.g. `page` or `data`.
So the proposed URI-template looks like:
http://inspire.ex.org/{format}/{namespace}/{localId}[/{**versionId**}]
and an example would be:
http://inspire.ex.org/page/NL.KAD.AU.GEM/4507/V1.0

4 Semantic query layer for INSPIRE Download Services

In this section we show at first how to translate from SPARQL to the WFS-query language "OGC Filter Encoding" and then tackle the prerequisite of references between the INSPIRE ontology concepts and the INSPIRE GML data structure.

4.1 SPARQL conversion to OGC Filter Encoding

We map from a given SPARQL algebraic expression into the target language of the OGC Filter Encoding. More precisely the target language is actually a combination of a) WFS-Query addressing the feature-type to be returned and b) OGC-Filter Encoding (FE) providing spatial, comparison and logical filter operators. From now on, we only mention OGC-Filter Encoding in lieu of both.

SPARQL as well as the XML-encoded FE are declarative languages but they operate on different information units: SPARQL combines triples while FE filters complex GML-features as WFS-records. Given the proposed INSPIRE ontologies one GML-feature and its dependant GML-elements are usually transformed to more than one triple, so FE actually filters at a coarser level than SPARQL does. Besides, there are SPARQL-functions which have no real FE-correspondents, e.g. `isBlank(a)`, `langMatches(a,b)` or `regex(a,b)` (FE only supports a comparator with wildcards, not regular expressions). Furthermore SPARQL-solution modifiers could not be translated, either. With this in mind we conclude that SPARQL is more powerful (even though FE provides more domain-specific filters, e.g. spatial ones). A translation to FE may loose filter information and may be too restrictive at the same time. A viable solution is a two-step query process. In the first step, an overly coarse query is forwarded to INSPIRE Download Services returning a superset of intended query results. In the second step, the precise SPARQL query is re-executed on the returned result set in order to yield the intended SPARQL results.

Some SPARQL filter-functions and value-comparing graphpattern are mapped to FE comparison or spatial operators (e.g. SPARQL-function `sameTerm(A,B)` to FE-operator `<PropertyIsEqualTo>`). In cases where SPARQL applies path-comparing pattern (e.g. `?feature ont_ps:siteName ?siteName`), we propose the mapping to a combination of the two FE-operators `<Not><PropertyIsNull>` to assure that the resulting GML contains only features with these subelements (here the info `siteName`) for further SPARQL-analysis afterwards. The table 1 shows the mapping of the main SPARQL-operators to their FE-pendants.

Table 1. Mapping SPARQL- to FE-operators

BGP(triple-pattern)	translate every path/value comparisons into appropriate FE-operators and combine them with the logical operator <AND>
Join(P1,P2)	like BGP-handling using both graph pattern P1 and P2
LeftJoin(P1,P2,F)	like BGP-handling, but only for mandatory branch P1, optional branch P2 and Filter F are ignored
Filter(F,P)	like BGP-handling using both filter F and graph pattern P
Union(P1,P2)	translate every comparisons under pattern P1 into appropriate FE-operators, enwrap them with logical <AND>, do the same for P2 and enwrap both <AND>-operators with logical <OR>

This simple conversion example transforms SPARQL-path comparisons and one temporal filter:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ont_ps: <http://inspire.ex.org/ProtectedSites/>

Select ?feature ?name ?beginLifespan
Where{
  ?feature a ont_ps:ProtectedSite .
  ?feature ont_ps:siteName ?name .
  ?feature ont_ps:beginLifespanVersion ?beginLifespan .
  FILTER( ?beginLifespan > "2009-01-01T12:00:00"^^xsd:dateTime )
}
```

to its corresponding WFS-GetFeature request (abbreviated):

```
...
<wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:
ProtectedSitesFull:3.0" typeName="ps-f:ProtectedSite">
  <ogc:Filter>
    <ogc:And>
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>ps-f:beginLifespanVersion</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
      <ogc:PropertyIsGreaterThan>
        <ogc:PropertyName>ps-f:beginLifespanVersion</ogc:PropertyName>
        <ogc:Literal>2009-01-01T12:00:00</ogc:Literal>
      </ogc:PropertyIsGreaterThan>
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>ps:siteName/gn:GeographicalName/gn:spelling/
                 gn:SpellingOfName/gn:text</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
    </ogc:And>
  </ogc:Filter>
</wfs:Query>
...
```

4.2 Annotations for GML-element references

In order to provide a SPARQL-endpoint which acts as a wrapper around WFS-services the SPARQL-endpoint must map INSPIRE ontologies to GML data models. When the SPARQL-endpoint receives a query, INSPIRE OWL-concepts have to be detected by their URIs (see Fig. 2, third step) and resolved to corresponding GML-element paths in order to compose a WFS-requests with OGC Filter Encoding filters (fourth step). For mapping OWL-concepts to GML-elements we annotate INSPIRE OWL-concepts with relative XPath-expressions for indicating their corresponding GML-element paths. We restrict the usage of XPath-version 1.0 to unique element references which means we avoid wild-cards for node-tests ('*') or predicate-filtering ('@*').

To this end we introduce instances of `owl:AnnotationProperty`:

- Annotation property `cf:xmlnamespace` is used at ontology-level to indicate which GML XML-namespaces are used in the particular INSPIRE ontology, e.g. `<http://inspire.ex.org/ProtectedSites> cf:xmlnamespace "xmlns:ps=\"urn:x-inspire:specification:gmlas: ProtectedSites:3.0\""`. Given these annotations, abbreviations of GML XML-namespaces with XML-prefixes are unique in the scope of the ontology
- Annotation property `cf:xmlname` is used at concept-level stating which OWL-class corresponds to which GML-object element, declared as a qualified XML-name, e.g. `ont_ps:ProtectedSite cf:xmlname "ps:ProtectedSite"` or `ont_ps:ResponsibleAgency cf:xmlname "ps-f:ResponsibleAgency"`
- Annotation property `cf:xpath` is used at concept-level, stating which OWL-property corresponds to which GML-property, declared as a relative XPath-expression (only XPath forward axes), e.g. `ont_ps:isManagedBy cf:xpath "ps-f:ProtectedSite/ps-f:isManagedBy/ps-f:ResponsibleAgency"`

With the `cf:xpath-annotation`, we adopt the triple-relationsship "subject-predicate-object" to the GML-model where in the given example above the GML-object element `ps-f:ProtectedSite` stands for the subject, GML-property `ps-f:isManagedBy` for the predicate, `ps-f:ResponsibleAgency` for the object. Figure 4 shows an example with annotations explained in detail subsequently:

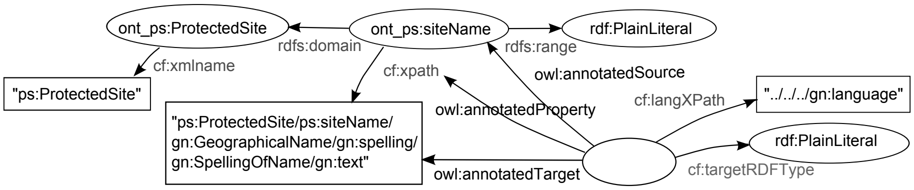

Fig. 4. XPath-expressions annotated to ontology concepts

In Fig. 4, the predicate `ont_ps:siteName` has a complex `cf:xpath`-annotation. This XPath-expression let us extract XML-information nested deeply within the

XML-tree so that we can ignore verbose GML element hierarchies (e.g. ISO-metadata elements) which are not needed in triple form. The other annotations `cf:langXPath` and `cf:targetRDFType` work as transformation hints and are realized as reification annotations so that they are unique for every `cf:xpath`. The reason is that there might be the need to assign multiple `cf:xpath` to one OWL-predicate if different INSPIRE GML-properties are equally named (distinguished only by their superordinated GML-object type) and so are derived both to one OWL-predicate. Or if INSPIRE GML data models allow alternative element storage, so multiple GML element paths each as one `cf:xpath` must be applied. The additional annotation `cf:langXPath` indicates where to find language-information about a text element directing there with a relative XPath-expression related to `cf:xpath`. The annotation `cf:targetRDFType` declares which RDF-datatype should be used for GML→OWL-transformation. Given the configuration example in Fig. 4, a transformation of this GML-snippet:

```
<ps:ProtectedSite>
  ...
  <ps:siteName>
    <gn:GeographicalName>
      <gn:language>deu</gn:language>
      ...
      <gn:spelling>
        <gn:SpellingOfName>
          <gn:text>Kleinkinzig− und Rötenbachtal</gn:text>
          <gn:script xsi:nil=”true”/>
        </gn:SpellingOfName>
      </gn:spelling>
    </gn:GeographicalName>
  </ps:siteName>
```

leads to the simple statement:
`ex:ProtectedSite1 ps:siteName "Kleinkinzig- und Rötenbachtal"@de`

Other GML-information which is likewise related to another GML-element as the language-code could be handled in the same way. A good example is the UOM-information related to its measure-value, so there could be an annotation `cf:uomXPath` and so on. In order to finally prepare WFS-querying, we need at least one more annotation (i.e. `cf:entityType`) for OWL-classes indicating which corresponding GML-object element is expected to serve as a WFS-feature type. In this case, the OWL-class has the annotatation `cf:entityType "featureType"`.

5 Prototype implementation

The INSPIRE test platform consists of the INSPIRE-enabled WFS ”Deegree3 inspireNode”[16] and representative test data. We have created two ontologies for the INSPIRE themes ”Protected Sites” and ”Administrative Units”[17]. These are tested with national Slovak protected sites (419 features, GML-size: 11 MB) and administrative units from the Dutch Kataster (443 features, GML-size: 23 MB).

[16] WFS-version 1.1.0; `http://www.deegree.org/`

[17] soon available under: `http://inspire.west.uni-koblenz.de:8080/ontologies`

In the INSPIRE implementation process Download Services are scheduled to be operational not until the end of 2012, so for now there is only test data available like the data we have used so far [18].

The Proxy-application is based on the Sesame framework[19] acting as an inference-layer using the ”Sesame” Sail-API. Sesame was chosen due to its well-structured API and fast conversion of SPARQL-query to -algebra. Internally, experimental GeoSPARQL-filters as well as programmetically generated WFS-GetFeature requests are realized with the Deegree3-API. The WFS-results are transformed with the fast non-extracting, XPath-capable XML-parser ”VTD”[20]. For example the parsing of all protected sites elements is finished within 6 seconds, which also includes all further transforming into RDF-triples. Testing federated queries is done with the Sesame-based APIs ”NetworkedGraphs” and ”DistributedSAIL” from the University of Koblenz-Landau [12].

6 Related Work

At least since the OGC Geospatial Semantic Web Interoperability Experiment in 2006 [7], the Geo Web community is concerned with the integration of Semantic Web-technologies within OGC Geo Web-services (OWS) and vice versa. Some attempts are constrained to a semantic reasoning support of OWS-communication [5], not publishing the Geo Web-data in the Semantic Web. Other approaches, like the active OGC SensorWeb domain, provide measure and sensor data through LOD-endpoints [9] [8]. They even interlink OGC sensor data with other LOD-repositories [6]. Otherwise, without a SPARQL-endpoint, query capabilities are neglected which we suggest to be more important than RDF-browsing and necessary to foster many Semantic Web use-cases. Some projects tackle SPARQL- or DL-queries translated to WFS-queries [3] [13]. We take a further step and query WFSs which operate on sophisticated, heavily nested INSPIRE GML models. The resulting INSPIRE Linked Data is created regarding INSPIRE-similarities to LOD concerning syntax, identification and referencing [11] and based on ontologies (schema-level) which is not quite usual for LOD-data [4].

7 Conclusions and Outlook

This paper presents an approach to enable federated queries over INSPIRE and LOD. We identify two key tasks, INSPIRE ontologies and support for querying, and propose how to solve them. Our solution has been tested with a prototype and turned out to be viable and efficient while establishing a semantic query layer for arbitrary WFS-services not constrained to INSPIRE data publishing.

Open issues are investigations in linking this INSPIRE- with other LOD-data, e.g. the data about European directives from the ”Reporting Obligations

[18] http://inspire.jrc.ec.europa.eu/index.cfm/pageid/44

[19] http://www.openrdf.org/

[20] http://vtd-xml.sourceforge.net/

Database"[21], or the full integration of spatial reasoning as soon as the specification GeoSPARQL will become stable. One big challenge will be a coordinated Semantic Web-infrastructure for INSPIRE reference data and management facilities of INSPIRE ontologies.

The Semantic Web is about handling information resources directly so we focus on the WFS-service and GML-encoded geodata itself, not the metadata layer consisting of OGC Catalogue Services and ISO-metadata. But INSPIRE also builds up a vivid metadata layer so that a combination of metadata and geodata forming one harmonized distributed Semantic Web-graph will certainly be a matter for future work.

References

1. Bedini, I., Gardarin, G. & Nguyen, B. [2008]. Deriving ontologies from XML schema. in Proc. of EDA 2008. Vol. B-4. pp. 3-17.
2. Bohring, H. & Auer, S. [2005]. Mapping XML to OWL ontologies. in Leipziger Informatik-Tage. pp. 147-156.
3. Gomes Jr, L. C. & Medeiros, C. B. [2007]. Ecologically-aware queries for biodiversity research. in IX Brazilian Symp. on GeoInformatics. pp. 73-84.
4. Jain, P., Hitzler, P., Yeh, P., Verma, K. & Sheth, A. [2010]. Linked data is merely more data. in H. H. D. Brickley, V. K. Chaudhri & D. McGuinness, eds, Linked Data Meets Artificial Intelligence. AAAI Press. Menlo Park, CA. pp. 82-86.
5. Janowicz, K., Schade, S., Bröring, A., Keßler, C., Maué, P. & Stasch, C. [2010]. Semantic enablement for spatial data infrastructures. in Trans. in GIS. Vol. 14(2).
6. Keßler, C. and Janowicz, K. [2010]. Linking sensor data - Why, to What, and How?, in Proc. of the 3rd Int'l workshop on SSN10, CEUR-WS, vol. 668.
7. Lieberman, J. [2006]. Geospatial Semantic Web Interoperability Experiment Report. OGC Inc. OGC 06-002r1.
8. Page, K., Roure, D. D., Martinez, K., Sadler, J. & Kit, O. [2009]. Linked sensor data: RESTfully serving RDF and GML. in Second Int'l Workshop on SSN2009.
9. Patni, H., Henson, C. & Sheth, A. [2010]. Linked sensor data. in IEEE, ed., 2010 Int'l Symp. on Collaborative Technologies and Systems. pp. 362-370.
10. Perry, M. & Herring, J. [2011]. OGC GeoSPARQL - A geographic query language for RDF data. OGC Inc. OGC 11-052r3. Url: http://portal.opengeospatial.org/files/?artifact_id=44722
11. Schade, S., Cox, S., Panho, M., Santos, M. & Pundt, H. [2010]. Linked data in SDI or how GML is not about trees. in Proc. of the 13th AGILE Int'l Conf. on Geographic Information Science - Geospatial Thinking.
12. Schenk, S., Staab, S. [2008]. Networked Graphs: a Declarative Mechanism for SPARQL Rules, SPARQL Views and RDF Data Integration on the Web. In: Proc. Int'l. WWW Conf., New York, NY, USA, ACM, pp. 585-594.
13. Zhao, T., Zhang, C., Wei, M. & Peng, Z. [2008]. Ontology-Based geospatial data query and integration. in T. Cova, H. Miller, K. Beard, A. Frank & M. Goodchild, eds, Geographic Information Science. Vol. 5266 of LNCS. Springer Berlin. Heidelberg.

[21] http://rod.eionet.europa.eu/void.rdf