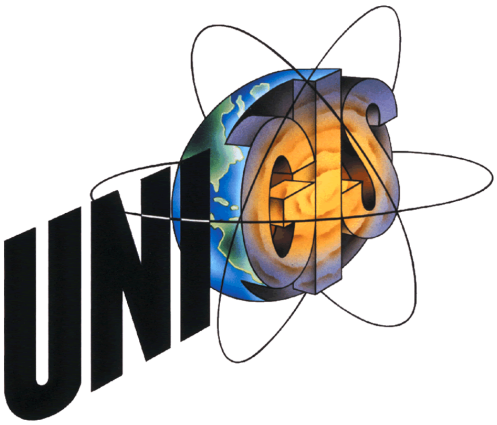


Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

Semantischer Zugriff auf INSPIRE

Verteilte Suche und Publikation von GML-Daten im Semantic Web am Beispiel INSPIRE

vorgelegt von

Dipl.-Ing. Sven Tschirner
U1430, UNIGIS MSc Jahrgang 2009

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) - MSc (GIS)“

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

Koblenz, 31.01.2012

Zusammenfassung

Die INSPIRE-Richtlinie bringt in absehbarer Zukunft europaweit harmonisierte Datenbestände überwiegend entgeltfreier Geofachdaten hervor. Der interoperable Datenaustausch wird dabei über standardisierte Geo Web-Dienstschnittstellen und Transferformate sichergestellt und die Geofachdaten in kompatiblen Fachportalen dargestellt und zur Recherche angeboten. Die starke Fixierung auf die Technikdomäne des Geo Web und die damit vorgegebenen Auswertemittel und Anwendungsfälle könnten sich jedoch hinderlich auf die Nutzung der eigentlichen Rohdaten auswirken. Denn sie erschweren eine semantische Datenanalyse, die gleichzeitige Auswertung mehrerer INSPIRE-Datentöpfe oder eine Verschneidung mit den heute bereits existierenden frei zugänglichen Internet-Datenbeständen behördlicher oder Nutzer-generierter Inhalte. Durch das Semantic Web und seiner Linked Open Data-Initiative zur Verknüpfung gemeinnütziger Daten bietet sich alternativ dazu eine technische Plattform, die aufgrund semantischer Auswertemethoden prädestiniert ist für derartige Integrationsaufgaben und entscheidend zur breiteren Nutzung von INSPIRE-Daten beitragen kann.

Im Sinne einer Machbarkeitsstudie präsentiert die vorliegende Arbeit einen umfassenden Ansatz zur Erschließung und verteilten Suche von INSPIRE-Daten im Semantic Web. Dazu werden in einer anfänglichen Untersuchung zunächst die Gemeinsamkeiten des Semantic Web und des Geo Web herausgestellt und die jeweiligen Standardisierungen und etablierten Methoden beleuchtet. Besonderes Augenmerk gilt der Geodatenverarbeitung, die als Bindeglied beider Disziplinen vom OGC erkannt und durch eine aktuelle Standardisierungsbemühung namens GeoSPARQL gefördert wird. Der GeoSPARQL-Spezifikationsentwurf, der die Einführung professioneller GIS-Auswertestrategien im Semantic Web beabsichtigt, wird in der Arbeit intensiv diskutiert. Die daraus gewonnenen Erkenntnisse wurden im Rahmen des offiziellen Review-Verfahrens als Korrekturvorschläge eingebracht.

Der technische Lösungsansatz der Arbeit ist zweigeteilt und beinhaltet erstens ein Modellierungskonzept, das Regeln zur Umformung von INSPIRE UML-Datenmodellen in Semantic Web OWL-Ontologien unter Berücksichtigung von Linked Open Data-Prinzipien definiert. Zweitens wird eine Systemarchitektur basierend auf einer Proxy-Anwendung vorgeschlagen, die das Schemawissen der OWL-Ontologien nutzt, um Semantic Web-Filteranfragen der verbreiteten Sprache SPARQL in WFS GetFeature-Aufrufe entfernter INSPIRE-Downloaddienste umzuwandeln. In einer Ergebnistransformation werden die ausgegebenen INSPIRE GML-Datensätze in Semantic Web-Formate konform zu den OWL-Ontologien überführt. Die Konzepte der Sprachabbildung und Ergebnistransformation sind in einem Prototypen realisiert und ihre Durchführbarkeit in verschiedenen Testszenarien unter Beweis gestellt. Mit der vorgeschlagenen Systemlösung wird ein transparenter und selektiver Semantic Web-Zugriff auf INSPIRE-Dateninhalte erzielt und die Möglichkeit eröffnet, ihr Nutzungspotential durch Mittel der verteilten Suche und semantischen Auswertung zu steigern.

Abstract

In the near future the INSPIRE-directive will make accessible pan-European harmonised geodatasets mostly free of charge. The interoperable data exchange will be established by standardised Geo Web-service interfaces and transfer formats while visualizing and querying geodatasets within compatible geoportals. The restriction to the Geo Web technical domain and the thereby predetermined analysis methods and use cases might imply obstacles for the usage of INSPIRE raw data. Because they generally hamper a semantic analysis, the simultaneous evaluation of multiple INSPIRE data pools or the integration with the already existing freely accessible Internet-databases of governmental or user-generated contents. The Semantic Web and its Linked Open Data initiative for linking noncommercial data offers an alternative technical platform which is predestined for such integration tasks due to its semantic evaluation means and therefore could make a promising contribution to a broader usage of INSPIRE data.

Serving as a feasibility study this thesis presents a comprehensive approach for accessing and doing distributed queries to INSPIRE data by means of the Semantic Web. For this purpose an examination is done to point out similiarities of both the Semantic Web and the Geo Web and to have a look at the particular standardisations and well-established methods of each discipline. Special attention is given to the field of geodata processing, which the OGC considers to be the linking key of both disciplines and therefore brought forward a new standardisation approach called GeoSPARQL. The GeoSPARQL draft which has the goal to introduce professional GIS analysis methods into the Semantic Web is explicitly discussed within this thesis and the results are submitted as comments during the official GeoSPARQL review.

The thesis' technical approach consists of two concepts. First, a modelling concept with conversion rules defining the conversion of INSPIRE UML data models into Semantic Web OWL ontologies, taking into account Linked Open Data principles. Second, a proposal for a system architecture based on a proxy application, which uses the schema knowledge of the OWL ontologies in order to convert Semantic Web queries of the common language SPARQL into WFS GetFeature requests and forward them to remote INSPIRE Download Services. Finally the resulting INSPIRE GML datasets are transformed into Semantic Web formats conform to the OWL ontologies. Both concepts, the language conversion as well as the result transformation are realised in a prototype implementation and their feasibility has been proved in several test scenarios. Hence, the proposed solution offers a transparent and selective Semantic Web access to INSPIRE data providing the opportunity to apply distributed queries and semantic analysis for an overall increase in value.

Erklärung

Ich versichere, diese Master Thesis ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäß übernommen wurden, sind entsprechend gekennzeichnet.

Ort, Datum	Unterschrift

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Hintergrund, Einordnung

Die europäische **INSPIRE-Richtlinie** (2007/2/EG, INSPIRE: *Infrastructure for Spatial Information in Europe*[1]) [Europäisches Parlament und Europäischer Rat 2007] ist Auslöser einer Entwicklung, in deren Folge die öffentlichen Verwaltungen der EU-Mitgliedsländer angehalten sind, ihre öffentlichen Informationen in eine gemeinsame IT-Infrastruktur, eine sogenannte *Geodateninfrastruktur* (GDI), einzubringen. Die damit bezweckte Interoperabilität reicht von der Datenebene und den darauf operierenden Infrastrukturdiensten bis hin zu organisatorischen Rahmenbedingungen, die entweder über freiwillige Abstimmungsprozesse oder rechtliche Vorgaben geschaffen werden. INSPIRE benennt die zu erfassenden Fachthemen, die sogenannten *Annex-Themen*, wie z.B. die Themen *Transportnetzwerke*, *Hydrographie* oder *Schutzgebiete*. Die geforderten Inhalte sind in einer vorgegebenen Struktur, den Datenmodellen der *INSPIRE-Datenspezifikationen*, über Webdienste bereitzustellen und mittels Metadaten zu beschreiben. Für INSPIRE-Daten charakteristisch ist ein ausgeprägter Raumbezug, weshalb in erster Linie Strategien zum Umgang mit Geodaten angewendet werden. Das Potential, das von diesen thematisch aufbereiteten und miteinander verknüpften Datenbeständen ausgeht, ist sehr hoch gemessen an der geplanten Verfügbarkeit und Abdeckung über alle EU-Mitgliedsstaaten. Gleichsam vielversprechend ist die zu erwartende Qualität und Aktualität der aus öffentlicher Hand stammenden Datensätze. Seitens der INSPIRE-Beteiligten gilt es trotz Sprachbarrieren, verschiedenster nationaler und kleinstaatlicher IT-Landschaften und datenrechtlicher Bestimmungen einem hohen Anspruch gerecht zu werden. Dieser besteht darin, nicht nur der EU-Kommission oder öffentlichen Verwaltungen, sondern auch dem einfachen EU-Bürger eine möglichst **grenzenlose und kostenfreie Geodateninfrastruktur** über öffentliche Datenbestände anzubieten.

Eine ähnlich ambitionierte Entwicklung zeichnet sich mit dem Aufkommen des **Semantic Web** seit der Jahrtausendwende ab. In dessen vielfältigem Rahmen werden mit freiwilliger Beteiligung zahlreiche Wissensbestände aufgebaut, deren Konzepte bzw. Objektklassen in sogenannten Ontologien spezifiziert sind und die sich über komplexe regelbasierte Abfragesprachen aus dem Umfeld der Beschreibungslogik durchforsten lassen. Hitzler et al. [2008] umschreibt das wesentliche Anliegen des Semantic Web mit den knappen Worten: „*Finde Wege und Methoden, Informationen so zu repräsentieren, dass Maschinen damit in einer Art und Weise umgehen können, die aus menschlicher Sicht nützlich und sinnvoll erscheint*“. Damit grenzt sich das Semantic Web vom altbekannten Internet mit seinen Hyperlinks und textuellen Inhalten derart ab, dass nunmehr die eigentlichen Daten bzw. das Faktenwissen und nicht deren Textrepräsentationen im Mittelpunkt stehen. Ebenso erfolgt eine mit Weblinks gestützte, unmittelbare Verknüpfung der Daten miteinander und zu ihren Konzepten, was semantischen Suchzwecken und automatisierten Schlussfolgerungen zugutekommt. Nach anfänglich hoher Komplexität der Konzeptwelt, verlagert sich die Aufmerksamkeit zunehmend auf leichtgewichtige Ansätze. Deren bekannteste Initiative **Linked Data**[2] hat mit ihren simplen, aber essentiellen Paradigmen bereits viele wertvolle Datentöpfe erschlossen, die als *Linked (Open) Data* (LOD) bezeichnet werden. Die LOD-Paradigmen beruhen auf der langfristigen Erreichbarkeit und Eindeutigkeit von Webinhalten sowie Methoden zu ihrer Verknüpfung. Die eingebrachten LOD-Inhalte setzen sich u.a. aus den populären Daten der digitalen Enzyklopädie

[1] Projektseite: http://inspire.jrc.ec.europa.eu/; INSPIRE-Geoportal: http://inspire-geoportal.ec.europa.eu/
[2] Projektseite: http://linkeddata.org

Wikipedia[3] und der freien Weltkarte *OpenStreetMap* (OSM[4]) zusammen. Auch die öffentliche Hand ist mittlerweile auf den zukunftsträchtigen Zug des Semantic Web aufgesprungen, beispielsweise die britische Regierung mit der Freischaltung des Datenportals *data.gov.uk* oder die europäische Umweltagentur mit weitreichenden Plänen zur Informationsvernetzung im Rahmen des Projektes *Shared Environmental Information System* (SEIS) [Roug 2009].

Beide Datenwelten, INSPIRE und Linked Open Data, eint das Ziel der freien Datenverfügung und der prinzipiellen technischen Architektur über verteilte Dienste und Datenverlinkungen. Sie basieren jedoch auf unterschiedlichen technologischen Entwicklungen. Während sich INSPIRE an den geographischen Standards der *International Organization for Standardization* (ISO) und dem *Open Geospatial Consortium* (OGC[5]) orientiert, deren Umfeld im Folgenden als *Geo Web* bezeichnet wird, richtet sich die Initiative Linked Data nach den vom *World Wide Web Consortium* (W3C) ausgearbeiteten Semantic Web-Spezifikationen. **Gemeinsame Schnittmengen** ergeben sich durch die stets stärker werdende Präsenz geographischer Informationen im Internet und Semantic Web. Diese drückt sich durch das gezielte Sammeln topographischer Inhalte auf freiwilliger und gemeinnütziger Basis aus (Schlagwort: *Volunteered Geographical Information* (VGI) als spezielle Form des *Crowd Sourcing*; Beispiel: OpenStreetMap) oder aber in der Anreicherung bestehenden digitalen Kartenmaterials mit Sachwissen (Schlagwort: *Social Tagging*; Beispiel: Google Earth[6]). Das OGC und ISO haben technische Mittel und Wege standardisiert, die bei diesen Aufgaben anfallenden Geoinformationen professionell zu verarbeiten, ob in Bezug auf die Persistierung, den Transfer von Geodaten oder deren Auswertung und Darstellung. Semantic Web-Projekte bedienen sich bereits einfachster Georeferenzierungen, beschränkt zumeist auf geographische Punktkoordinaten oder anderen einfachen Geometrietypen. Das OGC ist bestrebt, die eigene Expertise in das Semantic Web einfließen zu lassen, ob mit früheren Ansätzen, z.B. der Newsfeed-Spracherweiterung *GeoRSS* [Reed et al. 2006], oder aber der aktuellen und aussichtsreichen Bemühung um die Erweiterung der Semantic Web-Anfragesprache SPARQL [Prud'hommeaux & Seaborne 2008] mit räumlichen Filterungen - das sogenannte *GeoSPARQL* [Perry & Herring 2011]. Im Gegenzug versucht die Geo Web-Gemeinde, Semantic Web-Auswertetechniken zu adaptieren (Schlagwort: *Semantic Enablement*), um von neuen Möglichkeiten der Datenintegration und -transformation, der Konzeptionalisierung und loseren Kopplung verknüpfter Datenmengen zu profitieren. Zu unterscheiden sind Ansätze, die auf interne Umsetzung semantischer Strategien innerhalb einer GDI abzielen, z.B. mittels *semantischer Annotationen*, von jenen, die Mediatoren bzw. Proxy-Applikationen zur Kommunikation mit Semantic Web-Diensten und -Clients einsetzen und damit die Kluft zwischen beiden Technikwelten überbrücken. Die Bereicherung durch das Semantic Web, die das OGC erfährt, ist eine logische Fortsetzung der eigenen Wissensstrukturierung in existierenden Gazetteer- und Thesauri-Diensten. Sie kann dazu beitragen, um u.a. Problematiken der Mehrsprachigkeit (Internationalisierung) und der Datenverschlagwortung zu lösen.

Betrachtet man die **Wirkbereiche beider Internettechnologien** Geo Web und Semantic Web, fällt das Geo Web durch seine fachlich-professionelle Bindung und den allgegenwärtigen Raumbezug auf. Stattdessen zeigt sich das Semantic Web neutraler. Erstens hinsichtlich des Benutzerkreises, denn Behörden, die Privatwirtschaft oder Privatanwender - organisiert in Crowd Sourcing-Projekten - erzeugen gleichsam ähnlich dimensionierte Datenbeiträge. Zweitens angesichts der Dateninhalte, die sowohl materielle und topographische Objekte als auch Elemente der Gedankenwelt, Geschichte etc. einbeziehen. Die Koexistenz beider Welten und ihrer Besonderheiten ist als Chance anzusehen, Synergieeffekte zu nutzen und erfolgreiche technologische Ansätze sowie verfügbare Dateninhalte auszutauschen.

[3] Projektseite: http://de.wikipedia.org/wiki
[4] Projektseite: http://www.openstreetmap.org/
[5] Webauftritt: http://www.opengeospatial.org/
[6] Produktseite: http://www.google.de/intl/de/earth/

1.2 Motivation, Problemstellung

Motiviert wird die vorliegende Arbeit durch die Aussicht, zwei Systemwelten ohne großen organisatorischen Aufwand miteinander zu verzahnen. Aus technischer Sicht ist eine **Verknüpfung der Technologiestränge** verlockend. Auf der einen Seite die Spezifizierungen von OGC und ISO, letztere als De-jure-Normen durch geschlossene Gremien und sorgfältigste Standardisierungsprozesse erarbeitet. Auf der anderen die von der Internetgemeinde getriebenen Empfehlungen bzw. De-facto-Normen des W3C, die sich dank offener Gremienarbeit durch tendenziell höheren Praxisbezug und schnellere Reflektionen aktueller Entwicklungen auszeichnen. Welche Dienste sich zur Datenprozessierung beider Systemwelten koppeln und zur Mehrwertgenerierung in Reihe schalten lassen, wo Parallelen in der Informationsverarbeitung aufzufinden sind, all dies wurde bereits 2006 in einem OGC-Interoperabilitätsexperiment erforscht [Lieberman 2006] und ist für den Verwendungskontext INSPIRE erneut aufzufrischen und einer speziellen Untersuchung zu unterziehen.

Darüber hinaus ist es reizvoll, das Semantic Web durch die **Erschließung von INSPIRE** als eine multithematische und wohlstrukturierte Informationsquelle zu bereichern. Die laufenden INSPIRE-Abstimmungsprozesse begünstigen diesen Schritt, da sie keine isolierten und applikationsspezifischen, sondern harmonisierte Datenmodelle schaffen, die im Semantic Web als fachlische Konzeptwelten, sogenannte *Domänenontologien*, gut etabliert werden können. Die Themenvielfalt von INSPIRE mit insgesamt 34 Annex-Themen, u.a. Berichtsdaten zu europäischen Richtlinien, Katasterinformationen oder Erkundungs- und Forschungsdaten, die von Satelliten und Bodenobservationen für das europäische Programm GMES[7] und das internationale Netzwerk GEOSS[8] gewonnen werden, dürfte auf allgemeines Interesse stoßen. Ein INSPIRE-Datenzugriff bzw. Vertriebskanal über das Semantic Web könnte den INSPIRE-Bemühungen größere Aufmerksamkeit bescheren und INSPIRE speziell auch für den einfachen EU-Bürger attraktiver machen. Auch wird der Endanwender mit Hilfe von semantischen Anfragemitteln, z.B. mit der Sprache SPARQL, in die Lage versetzt, komplexe Datenanalysen zu starten und interessante Anwendungsfälle durchzuspielen. Denn unter bestimmten Voraussetzungen der Informationsvernetzung lassen sich beispielsweise auch Konzeptzusammenhänge schlussfolgern oder Anfragen verteilen, um mehrere Datenquellen gleichzeitig zu durchforsten und Resultate kombiniert an den Klienten zurückliefern. Wie konkrete **Anwendungsszenarien** für die INSPIRE-Datenzugriffe und verteilte Suchvorgänge aussehen können, wird im Verlaufe der Arbeit näher betrachtet und die technische Machbarkeit eingehend geprüft (vgl. Kapitel 3 *Anforderungsanaylse*, 5.3 *Implementierung eines Prototypen* und 6 *Ergebnis und Diskussion*).

Die Motivation speist sich auch daraus, dass INSPIRE in seinen wesentlichen Strukturen fertig spezifiziert ist, d.h. es liegen bereits für die Infrastrukturelemente wie Such- und Darstellungsdienste und die angesprochenen Datenspezifikationen **stabile INSPIRE-Implementierungsanweisungen** oder -empfehlungen vor. Darauf kann sich die vorliegende Arbeit stützen und als eine der ersten Beiträge nicht nur der zukünftigen Projektumsetzung vorweggreifen, sondern sogar die Nachnutzung der INSPIRE-Strukturen in das Semantic Web als einen weiteren Anwendungskontext diskutieren. Dem Autor ist keine ähnliche Arbeit bekannt, die diesen Themenkomplex annähernd intensiv beleuchtet. Einhergend mit der exklusiven Themenwahl ergibt sich aber auch die Problematik, das INSPIRE noch nicht den endgültigen Sprung von der Theorie in die Praxis geschafft hat, konforme Datensätze bislang Mangelware sind und fertige INSPIRE-Dienste geschweigedenn lauffähige Testumgebungen kaum zur Verfügung stehen. Zudem reagieren die Software-Hersteller recht zögerlich auf die Spezifikationsentwicklungen und brachten erst wenige kompatible Produkte auf den Markt. Für die Machbarkeitsuntersuchung im Rahmen dieser Arbeit ist deshalb der Aufbau einer eigenen INSPIRE-Testplattform und die Suche nach oder die Herleitung von repräsentativen INSPIRE-Testdatensätzen erforderlich.

[7] Global Monitoring for Environment and Security; Projektseite: http://www.gmes.info/
[8] Global Earth Observation System of Systems; Projektseite: http://www.earthobservations.org/geoss.shtml

Letztlich liegen diverse Motivationsgründe vor, die allesamt für die Erschließung der INSPIRE-Infrastruktur und Publikation der INSPIRE-Daten im Semantic Web sprechen. Um dieses Ziel zu verfolgen und eine Realisierung anzustreben, kommt man nicht umhin, sich mit **zwei zentralen Problemstellungen** zu befassen:

1. die **Umformung von INSPIRE-Daten in ein gängiges Semantic Web-Format**: im Umfeld von Geodateninfrastrukturen und im Speziellen im INSPIRE-Projekt werden Geodaten standardkonform zu der *Geography Markup Language* (GML) [Portele 2007] gespeichert und transferiert. Das XML-Format GML ist in ein Semantic Web-Format zu überführen. Hierfür bieten sich die weitverbreiteten Wissensrepräsentationssprachen *Resource Description Framework* (RDF) [Manola & Miller 2004] und die *Web Ontology Language* (OWL) [Hitzler et al. 2009] an. Die Herausforderung besteht darin, die Konvertierung möglichst effizient und ohne Informationsverlust zu bewerkstelligen. Zugleich sollten wesentliche Semantic Web-Prinzipien zur Anwendung kommen, um eine vielfältige Nachnutzung zu sichern, u.a. durch Integrieren semantischer Bezüge und Querverweise in den resultierenden Daten.

2. die **Abbildung von Semantic Web- auf Geo Web-Datenanfragen**: um INSPIRE-Quellen für das Semantic Web zu öffnen und dabei intelligente Suchvorgänge starten zu können, die ohne einen vollständigen Datendownload auskommen, werden Regeln benötigt, die die automatisierte Abbildung von Semantic Web- auf Geo Web-Anfragen erlauben. Ziel ist ein transparentes Geo Web, das ohne manuellen Eingriff ad-hoc-Anfragen von Semantic Web-Klienten bedienen kann. Die Wahl der Mittel bzw. der zu untersuchenden Anfragensprachen für eine Sprachabbildung fällt relativ leicht. Denn das Geo Web stellt seine Geodaten in der Regel über Webdienste des OGC-Diensttyps *Web Feature Service* (WFS) [Vretanos 2005*b*] zum Download bereit. WFS-Dienste beherrschen zumeist nur die Basisfiltersprache, das sogenannte *Filter Encoding* (FE) [Vretanos 2005*a*], weshalb als Zielsprache der Sprachabbildung nur Filter Encoding in Frage kommt. Ähnlich exklusiv sieht es im Bereich des Semantic Web aus. Die bereits erwähnte Sprache *SPARQL* hat sich - nach dem Grad ihrer Entwicklung und der Verbreitung zu urteilen - zur wichtigsten Anfrage- und Filtersprache von ontologischen Wissensbasen *gemausert*. Aus diesem Grund fällt der SPARQL - inklusive der räumlichen Erweiterung *GeoSPARQL* - zweckmäßigerweise die Rolle der Ausgangssprache zu. Folglich konzentriert sich die Aufgabenstellung auf das Definieren von Regeln, die eine durchgängige und verlustfreie Sprachabbildung bewirken. Dazu sind trotz verschiedener Verwendungskontexte und der dadurch bedingten sprachlichen Differenzen möglichst alle SPARQL-Sprachkonstrukte und -Filteroperatoren in FE-Entsprechungen zu übersetzen.

Abgesehen von den fachlich abzustimmenden Inhalten und technischen wie organisatorischen Einrichtungen ist ebenso auf *weiche Kriterien* eines *Semantic Enablement* zu achten. Dazu zählen beispielsweise die Einhaltung der Datenkonsistenz oder ein geringer Konfigurationsaufwand für die INSPIRE-Datenhalter bzw. Administratoren von Datenschnittstellen.

1.3 Zielsetzung und Hypothesen

Untersucht werden die Mittel und Wege, einen Semantic Web-Zugriff auf INSPIRE-Daten zu bewerkstelligen. Dafür ist die Frage zu klären, ob und in welchem Grad sich Geodaten in ein semantisches Format überführen lassen und dabei den Prinzipien des vom W3C geförderten Semantic Web bzw. der populären Linked Data Initiative entsprechen können.

Die einzelnen **Ziele** sind:

- die allgemeine Erkundung der Gemeinsamkeiten des Semantic Web und Geo Web hinsichtlich der Informationsstrukturen und insbesondere der Geometriespeicherung und -verarbeitung
- die Entwicklung einer Methodik, wie Geodaten des Formats GML in das Semantic Web-Format RDF bzw. OWL transformiert werden können. Hierzu ist ein genereller Lösungsweg zur Modellierung von OWL-Konzepten als Bestandteile von INSPIRE-Ontologien zu entwickeln und mit repräsentativen INSPIRE-Testdatensätzen zu prüfen.
- die Konzeptionierung einer geosemantischen Anfrageschnittstelle von OGC-Datendiensten mit der Einschränkung auf den OGC-Diensttyp Web Feature Service (WFS), der auch von INSPIRE vorrangig adressiert wird. Nicht nur der direkte Datenabruf, sondern auch eine intelligente Suche bzw. Filterung nach OGC-Geodaten gilt es zu unterstützen. Dafür ist eine unidirektionale Abbildung von SPARQL-Anfragen (Semantic Web) auf Filter Encoding-Ausdrücke (Geo Web) zu entwerfen.
- die Implementierung eines Prototypen als technischer Machbarkeitsnachweis (*Proof of Concept*)

An dieser Stelle soll auch darauf hingewiesen werden, dass das anfängliche Arbeitsthema vorsah, im Rahmen der Master Thesis nur eine Ontologie zum INSPIRE-Thema *Schutzgebiete* zu entwickeln. Zwar wird das Thema *Schutzgebiete* nach wie vor exemplarisch in den Grundlagenthemen betrachtet und zur Ontologie-Modellierung herangezogen. Es hat sich jedoch im Laufe der Bearbeitung herausgestellt, dass die INSPIRE-Datenspezifikationen große Gemeinsamkeiten aufweisen und auf klaren Prinzipien beruhen, so dass der jetzige Lösungsansatz (siehe Kapitel 4) allgemeine Modellierungsregeln definiert, die auf alle bisher fertig spezifizierten Annex-I-Themen und vergleichbare zukünftige INSPIRE-Themen aus Annex-II/III angewendet werden können. Die Flexibilität wird dank einer weitestgehend syntaktischen Umformung der INSPIRE-Datenmodelle (GML-Schemata) in INSPIRE-Ontologien (RDF/OWL) erreicht.

Um eine zweckmäßige thematische Abgrenzung vorzunehmen, sind als **Nicht-Ziele** zu nennen:

- die Metadatenebene, bestehend aus Metadatensätzen und -diensten, wird nicht berücksichtigt und der Fokus allein auf die Erschließung der INSPIRE-Datenschicht gelegt. Zweifellos spielen Metadaten eine bedeutende Rolle zu Zwecken der Datenbeschreibung und -suche, jedoch sprechen zwei gewichtige Gründe gegen die Behandlung des Metadatenthemas innerhalb dieser Arbeit. Einerseits machen die harmonisierten INSPIRE-Datenspezifikationen Metadaten bis zu einem gewissen Grad überflüssig, indem inhaltliche und qualitative Vorgaben (z.B. eingegrenzte Auflösungsstufen, einheitliche Kategorisierungen etc.) vergleichbare Datentöpfe entstehen lassen. Zum anderen trennt das Semantic Web nicht strikt zwischen der Daten- und der Metadatenebene, wie das im Geo Web praktiziert wird. Stattdessen speichert man Metadaten als Aussagen und Annotierungen zusammen mit den eigentlichen Dateninhalten und überwindet dadurch die auch im Geo Web oft fließenden Grenzen. Allerdings ist die bisherige Praxis mit Vorsicht zu genießen, denn es hat sich im Semantic Web- als auch in der kleineren Linked Data-Gemeinde bislang noch keine abgestimmte Vorgehensweise zur Speicherung von Metadaten herauskristallisiert. Allenfalls verschiedene Empfehlungen und Vorschläge sind publiziert [Hartig & Zhao 2010], [Linking Open Data Community Project 2007]. Diese und andere Ansätze bedürfen einer gründlichen Überprüfung, um eine Best-Practice-Lösung der Überführung von Metadateninhalten aus dem Geo Web zu ermitteln, und sind deshalb als eigenständiges Arbeitsthema anzusehen.

- es werden nur die Server-seitigen Transformations- und Suchprozesse tangiert, eine Einbindung in oder gar Implementierung von GUI-gestützten Web-Clients erfolgt nicht. Für die Evaluierung der prototypischen Anfrageschnittstelle ist ein rudimentärer SPARQL-Client ohne GUI ausreichend. Das Ziel ist es, in einem ersten Schritt eine zweckmäßige Dienstekommunikation zwischen dem Semantic Web und dem Geo Web herzustellen und dadurch den Zugriff auf INSPIRE-Ressourcen zu ermöglichen. Weitere Schritte könnten darin bestehen, eine Client-seitige Suchunterstützung über benutzerfreundliche Weboberflächen bereitzustellen.

Im Hinblick auf die Untersuchung werden folgende **Hypothesen** formuliert:

- es treffen genügend begünstigende Faktoren - ausgelöst durch etablierte Standards und dem allgemein fortgeschrittenen Stand in Wissenschaft, Technik und Anwendung - zusammen, um INSPIRE-Geodaten effizient und interoperabel im Semantic Web nutzbar zu machen.
- die Überführung von INSPIRE-Datenmodellen in semantische Ontologien ist lohnenswert und unkompliziert, weil jede INSPIRE-Datenspezifikation ein bestimmtes Fachgebiet abdeckt und somit Qualitäten einer Domänenontologie aufweist.

1.4 Struktur der Thesis

Die Struktur der Arbeit ist in Abbildung 1.1 wiedergegeben. Die Abbildung führt die hohe Komplexität der Grundlagenthemen in **Kapitel 2** vor Augen, die sich aus den beiden maßgeblichen Wissensgebieten Semantic Web und Geo Web speisen. Für einen einfachen Einstieg werden zu jedem Wissensgebiet zunächst die wesentlichen Begriffe definiert (Abschnitt 2.1) und im Anschluss mitsamt bedeutenden Normen und Strategien vorgestellt (Abschnitte 2.2 und 2.3). Der Abschnitt 2.4 zeigt die Berührungspunkte beider Wissensgebiete und nennt die existierenden Semantic Web-Ansätze zur Geodatenspeicherung und -abfrage. Ebenso wird die neueste Entwicklung, der im Juli 2011 veröffentlichte Standardentwurf GeoSPARQL, betrachtet und im weiteren Verlauf der Arbeit umfangreich diskutiert. Der letzte Abschnitt 2.5 des Grundlagenkapitels führt in das Projekt der europäischen Geodateninfrastruktur INSPIRE ein, das gewissermaßen die Datenquelle darstellt, mit der sich die vorliegende Arbeit beschäftigt.

Das anschließende **Kapitel 3** zeigt den Verwendungskontext von INSPIRE-Daten im Semantic Web, indem potentielle Anwendungsszenarien für die Suche und Verarbeitung von INSPIRE-Schutzgebietedaten im Zusammenspiel mit anderen Semantic Web-Quellen aufgezeigt werden. Aus den Anschauungsbeispielen werden im nächsten Abschnitt 3.2 Nutzeranforderungen abgeleitet. Sowohl die Anwendungsszenarien als auch die Nutzeranforderungen dienen der späteren Evaluierung des Lösungsansatzes.

Der Lösungsansatz folgt in **Kapitel 4**, bestehend aus zwei Abschnitten. Erstens einer Beschreibung des Inhaltsmodells bzw. der wichtigsten Design-Ziele für die Modellierung von INSPIRE-Ontologien (Abschnitt 4.1). Zweitens werden Systemarchitekturen für geosemantische INSPIRE-Anfrageschnittstellen verglichen und Argumente für die in dieser Arbeit gewählte Architektur vorgebracht (Abschnitt 4.2).

Der Lösungsansatz wird konkretisiert in **Kapitel 5**. Zunächst wird die favorisierte Methodik zur Ontologieentwicklung vorgestellt (Abschnitt 5.1). Anschließend werden die einzelnen Prozesse und Komponenten einer geosemantischen Anfrageschnittstelle auf INSPIRE-Downloaddienste besprochen (Abschnitt 5.2), die das Konzept für die prototypische Implementierung bilden. Die letztliche Implementierung sowie die dafür notwendige INSPIRE-Testplattform und die verwendeten Testdatensätze sind Thema des Abschnittes 5.3.

Das **Kapitel 6** enthält eine kritische Betrachtung der erarbeiteten Konzepte und des Prototypen. In diesem Zusammenhang werden die in der Einleitung getroffenen Hypothesen und die analysierten Nutzeranforderungen geprüft. Die Arbeit schließt in **Kapitel 7** mit der Zusammenfassung der Ergebnisse und dem Ausblick auf potentielle Anschlussthemen und zukünftige Entwicklungen.

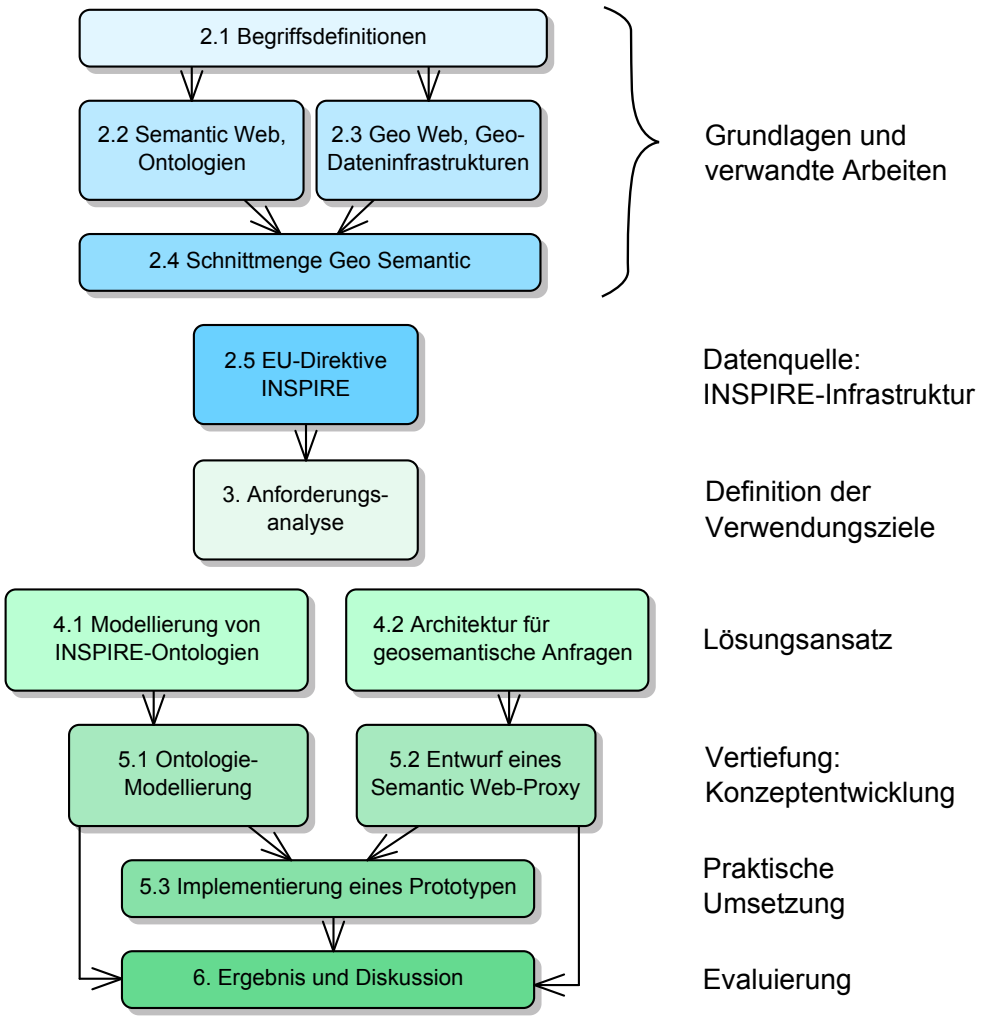


Abbildung 1.1 – Struktur der Arbeit

2 Grundlagen und verwandte Arbeiten

Die nächsten Abschnitte überblicken die nötigen Grundlagenthemen, relevanten Standardisierungen und verwandten Fachbeiträge, in die sich die vorliegende Arbeit einbettet. Angefangen mit Begriffsdefinitionen, die ein Durcheinander an Namen und gleichbedeutenden Begriffen vermeiden sollen, führt das Kapitel ein in die beiden Disziplinen Semantic Web und Geo Web und beachtet dabei die im Wachsen begriffenen technologischen Überschneidungen.

Angesichts der umfänglichen Grundlagen möchte der Autor darauf verzichten, allgegenwärtige IT-Technologien vorzustellen. Dazu zählen insbesondere die Auszeichnungssprache *Extensible Markup Language* (XML) und die zur XML-Sprachfamilie gehörenden Analysetechniken zur Inhaltstransformation (XSLT), -Selektion und -Abfrage (XPath und XQuery) und der Informationsvernetzung per XLink. Ebenso werden Kenntnisse zur konzeptionellen Modellierungssprache *Unified Modeling Language* (UML), zur bekannten Datenbanksprache *Structured Query Language* (SQL) und zu Formen und Kommunikationswegen des heutigen Internets vorausgesetzt, u.a. also auch die Verwendung der Netzwerkprotokolle *Hypertext Transfer Protocol* (HTTP) und *Simple Object Access Protocol* (SOAP).

2.1 Begriffsdefinitionen

Die nun folgenden Begriffsdefinitionen sind nach den Disziplinen Semantic Web und Geo Web unterschieden, um die namentliche Herkunft und den Verwendungskontext hervorzuheben. Es werden überdies nur die grundlegendsten Begriffe erläutert, die die jeweiligen Informationsstrukturen prägen. Diese und weitere Begriffe werden im Anschluss anhand von Beispielen in Beziehung gesetzt.

Für die verwendeten Begriffe wird die Konvention getroffen, dass bei nahezu gleicher Schreibweise deutscher und englischer Fachbegriffe die deutsche Bezeichnung bevorzugt wird. Sollten im Umkehrschluss englische Fachbegriffe deutlich von ihren deutschen Übersetzungen abweichen oder mehrdeutige Übersetzungen haben, werden die englischen Bezeichner gewählt. Wörter, die in `Schreibmaschinenschrift` dargestellt sind, verweisen auf eine separate Begriffsdefinition.

Begriffe aus dem Semantic Web-Umfeld

ABox
Menge an Fakten- bzw. Instanzwissen. Das *A* in ABox steht für den philosophischen Begriff *Assertion* und bedeutet „*feststellende Behauptung*“ oder „*Versicherung*“[9]. In einer `Wissensbasis` bezeichnet die ABox alle Aussagen bzw. Instanzdaten, während die zugehörigen `Konzepte` Elemente der sogenannten `TBox` sind.

Annotation
Aussagen, die meist zur Beschriftung von `Klassen` und `Prädikaten` dienen, die `Wissensbasis` jedoch nicht mit weiteren logischen Aussagen anreichern. Annotationen werden beispielsweise mit den Prädikaten *rdfs:label*, *rdfs:comment* und *owl:versionInfo* eingeleitet. Solche Zusatzinformationen werden von vielen semantischen Programmen in graphischen Oberflächen als Benutzerhilfen eingesetzt. Eine spezielle Annotationsform sind die Axiom-Annotationen, siehe auch `Reifikation`.

[9] Quelle: Duden online; siehe: http://www.duden.de

Beschreibungslogik (engl. Description Logic)
Neben der Graphentheorie die zweite grundlegende Wissenschaft, auf der das Semantic Web basiert. Als Einschränkung der Prädikatenlogik erster Stufe (engl. *First Order Logic*, FOL) ist die Beschreibungslogik relevant für die `Inferenzbildung`.

Blank Nodes (dt. auch *Leere Knoten*)
Blank Nodes sind `Ressourcen`, die nicht eindeutig identifiziert sind. Sie sind gebräuchlich zur Gruppierung von zusammengehörigen Elementen, die jedoch als Komposition keine gesonderte Identität über einen konkreten Bezeichner benötigen.

Graph
Gerichteter Graph gemäß der Graphentheorie, der `Ressourcen` als Graphenknoten und `Prädikate` als Graphenkanten beinhaltet.

Individuum (Synonym: Objekt, Instanz)
Repräsentation eines realen Objektes, das ein `Konzept` aus der Ideenwelt instanziiert. Der `ABox` zugehörig. Äquivalent zum `Feature` im Geo Web, ohne einen Raumbezug vorauszusetzen.

Inferenzbildung (engl. Reasoning)
Schlussfolgerung impliziten Wissens in der `Wissensbasis`. Gibt es beispielsweise in der `TBox` voneinander abgeleitete `Konzepte` oder `Prädikate` mit Einschränkungen von Datentypen und deren Wertebereichen, dann kann ein sogenannter *Reasoner* die Instanzdaten in der `Wissensbasis` daraufhin validieren und neue Aussagen ableiten.

Klasse
Merkmalstyp bzw. Kategorie inhaltlich gleichartiger `Ressourcen`.

Konzept
Der Begriff Konzept umfasst Definitionen von `Klassen`, `Prädikaten` und Datentypen aus der `TBox`. Das Semantic Web bedient sich vornehmlich zweier Wissensrepräsentationssprachen zur Beschreibung von Konzepten: das Resource Description Framework (RDF) [Manola & Miller 2004] und die darauf aufbauende Web Ontology Language (OWL) [Hitzler et al. 2009]. Vorsicht ist bei manchen `Vokabularen` für Taxonomien und Thesauri geboten, die den Begriff Konzept auf andere Weise definieren und verwenden als in RDF und OWL üblich, so z.B. durch das Vokabular *Simple Knowledge Organization System* (SKOS) [Miles & Bechhofer 2009].

Literal
Sachattributinformation zur Beschreibung von `Ressourcen`.

Ontologie (Synonym: Wissensbasis, im Speziellen auch: Vokabular, Wissensmodell)
Die Philosophie versteht unter einer Ontologie allgemein die „*Lehre vom Sein*“[10]. Nach Tom Gruber [Gruber 1993] ist eine Ontologie eine „*formale und explizite Spezifikation der Konzeptionalisierung eines gemeinsamen Wissensgebietes*“. Damit wird ausgedrückt, dass die verwendeten Konzepte für eine Fachdomäne gültig, bekannt und mit derselben formalen Sprache beschrieben werden. Demnach beschränkt sich der Begriff im ursprünglichen Sinn auf die Konzeptebene der `TBox` (`Vokabular`), allerdings ist mit dem Begriff Ontologie häufig auch die ganze `Wissensbasis` gemeint - inklusive der `ABox`.

Prädikat (ähnliche Begriffe: Role, Relation, Property; dt. Rolle, Eigenschaft, Attribut)
Beziehung zwischen einer `Ressource` als Subjekt und einer weiteren `Ressource` oder einem `Literal` als Objekt. Ein Prädikat verknüpft eine `Ressource` mit sich selbst oder beliebig vielen anderen `Ressourcen`. Hingegen besteht eine *Relation* aus mindestens zwei oder mehr voneinander verschiedenen `Ressourcen`. OWL unterscheidet im Gegensatz zu RDF zwischen einem `Literal` als Objekt - dann wird das Prädikat *DatatypeProperty* genannt - oder einer weiteren `Ressource` als Objekt - dies nennt sich *ObjectProperty*.

[10] Quelle: Duden online; siehe: http://www.duden.de

Quad
Hierbei handelt es sich um ein Aussage (`Statement`), die zusätzlich zu den Strukturelementen eines `Triples` - Subjekt, Prädikat, Objekt - auch eine Kontextangabe enthält. Zumeist wird zu diesem Zweck eine `Graphen`-URI verwendet, über die die Angabe der Datenquelle geschieht, z.B. *http://dbpedia.org* als Graphen-URI für Semantic Web-Daten der freien Enzyklopädie *Wikipedia*.

Reifikation
Aussage, die sich explizit auf eine weitere Aussage bezieht. Beispielsweise ließe sich der primären Aussage A: *Firma X hat eine Filiale in Dresden* eine sekundäre Aussage anheften: *Zeitungsagentur Y hat verkündet, dass 'die Firma X eine Filiale in Dresden hat'*. Diese Art der abhängigen Aussagen heißen in der Ontologiesprache OWL Axiom-Annotationen.

Ressource (Synonym: Entität)
Ressourcen kommen entweder eindeutig identifiziert, als `Individuen` oder `Konzepte`, oder aber ohne Bezeichner als `Blank Node` vor. In OWL [Hitzler et al. 2009] heißen Ressourcen auch *Entitäten*.

Social Tagging
Verschlagwortung insbesondere räumlicher `Ressourcen` (`Features`) im Social Web auf gemeinschaftlicher freier Basis. Sammlungen dieser Schlagwörter (*Tags*) bilden die sogenannten *Folksonomien*, die als Wortschatz betrachtet meist recht unstrukturiert sind.

Statement (Synonym: Fact, Axiom, Assertion; dt. Fakt(um), Aussage)
Elementare Aussage bzw. ausgedrückter Sachverhalt, meist als `Tripel`, manchmal auch als `Quad` vorkommend.

TBox (Synonym: Vokabular)
Menge an terminologischem (Schema-) Wissen, womit die `Konzepte` einer `Wissensbasis` gemeint sind.

Tripel
Ein Tripel besteht aus einem Subjekt, d.h. einer zu beschreibenden `Ressource`, einem `Prädikat`, das die Eigenschaft des Subjekts angibt, und einem Objekt als Argument des `Prädikats`. Das Objekt kann eine weitere `Ressource` oder ein `Literal` sein.

Vokabular (Synonym: TBox)
Siehe `TBox`.

Wissensbasis (engl. Knowledge base (KB), Repository)
Menge aller Aussagen (`Statements`), die in einem physikalischen oder virtuellen Datenspeicher zusammengefasst sind. Beispielsweise kommen Wissenbasen häufig in Form von Datenbank-gestützten Datenhaltungen vor, sogenannte *RDF-Triplestores*.

Begriffe aus dem Geo Web-Umfeld

Applikationsschema
Spezifikation der Struktur eines Transferformates bzw. einer Datenbasis mittels formaler Beschreibung von Objekttypen (siehe `Feature Type`), deren Eigenschaften und Werteeinschränkungen. Die physische Modellierung von Transferformaten geschieht üblicherweise mit *XML-Schema* (XSD) [Fallside & Walmsley 2004], der Strukturbeschreibungssprache für Dokumente im XML-Format.

Codelist
Vordefinierte Werteliste, die technische Codes auf sprechende Namen abbildet und zur eingeschränkten Attributierung von Objekten dient.

Datensatz
Identifizierbare Datenmenge aus Einheiten gleichen Daten- bzw. Objekttyps, die in Dateien oder in einer relationalen Datenbank abgelegt ist.

Datenprodukt
Räumlicher `Datensatz` bzw. Datensatzserie, die konform zu ISO 19131 [ISO/TC 211 2007*a*] spezifiziert ist. ISO 19131 verlangt u.a. eine präzise Darstellung des Verwendungszwecks und der Anwendungsszenarien, der Datenqualität und der Datenerfassungs- sowie Datenpflegevorgänge. Die INSPIRE-Datenspezifikationen sind Beispiele konformer ISO 19131-Datenprodukte.

Enumeration
Spezielle `Codelist`, die in sich abgeschlossen ist, d.h. die Definition weiterer Codelist-Werte ausschließt. Diese Unterscheidung wird insbesondere im Geodatenformat *Geography Markup Language* (GML) [Portele 2007] und im INSPIRE Konzeptdokument *Generic Conceptual Model* [INSPIRE Drafting Team „Data Specifications“ 2010*a*] vorgenommen.

Feature (Synonyme: Objekt, Entität, (Objekt-)Instanz)
Repräsentation eines realen Objektes, dessen thematische, zeitliche und inbesondere räumliche Charakteristiken festgehalten werden. In ISO 19101 [ISO/TC 211 2002*a*] kurz als „*Abstraktion eines Realwelt-Phänomens*“ bezeichnet. Ein geographisches Feature wird zusätzlich dargestellt als „*Feature assoziiert mit einer Lage relativ zur Erde*“.

Feature concept
Element eines `konzeptionellen Schemas`, meist in Form einer UML-Klasse. Aus dem Feature concept können ein oder mehrere `Feature types` (in XML-Kodierung: Complex-types) auf der physischen Ebene des (XML-) `Applikationsschemas` resultieren. Damit sind Feature concepts als semantisches Bindeglied verschiedener `Applikationsschemata` geeignet, um beispielsweise wie im INSPIRE-Prozess Vereinfachungen, Abstraktionen oder Harmonisierungen herbeizuführen.

Feature catalogue (dt. Objekttypenkatalog)
Registrierung der Bestandteile eines `Applikationsschemas` (Feature-/Objekttypen, Attribute, Wertelisten etc.), um neben der technischen Form des `Applikationsschemas` eine rein textuell, leicht verständliche und gegebenenfalls multilinguale Definition von Informationseinheiten vorzuhalten und deren schnelles Auffinden zu ermöglichen. Das Katalogisieren wird beispielsweise durch ISO 19110 [ISO/TC 211 2005*b*] standardisiert und lässt sich in einem Registrierdienst in einem oder mehreren untergeordneten Registern organisieren, z.B. konform zu ISO-19126 [ISO/TC 211 2009] und ISO 19135 [ISO/TC 211 2005*c*].

Feature concept dictionary
Leichtverständliche textuelle Definition von `Feature concepts`, die aus dem `konzeptionellen Schema` stammen. Ein Feature concept dictionary entspricht dem `Feature catalogue` auf konzeptioneller Ebene. Ebenso lassen sich Registrierdienste nach ISO-19126 und ISO-19135 einrichten.

Feature type
Klasse zur Kategorisierung von Objekten (siehe `Feature`) mit ähnlichen Objektmerkmalen.

Geobasisdaten (Synonym: Geobasisinformationen)
`Geodaten`, die in der Gestalt von Infrastruktur- und topographischen Daten einen Bezugsrahmen für Fachinformationen (*Geofachdaten*) schaffen.

Geodaten (Synonym: Geoinformationen)
Daten mit Raumbezug bzw. einer `Georeferenzierung`.

Georeferenzierung
Erlaubt den Raumbezug von Objekten (`Features`), indem die räumliche Dimension mit Koordinaten oder geographischen Namen (*Toponymen*, z.B. Stadt- oder Flussnamen) angefügt wird. Koordinaten stellen eine direkte und geographische Namen eine indirekte Georeferenzierung dar.

Konzeptionelles Schema
Definiert Eigenschaften von Konzepten (siehe `Feature concept`) und deren Zusammenhänge in einer formalen Sprache. Als konzeptionelle Schemasprache wird in der Geoinformatik überwiegend die *Unified Modeling Language* (UML) eingesetzt.

Koordinatenreferenzsystem (Synonym: (Lage-)Bezugssystem)
Metainformation für Koordinaten, ohne die kein eindeutiger Lagebezug von Koordinaten möglich ist. Ein Koordinatenreferenzsystem besteht erstens aus einem *geographischem Datum* zur Lagerung eines Referenzkörpers (Spheroid oder Ellipsoid) relativ zum Erdzentrum und zweitens einer *Projektion* (auch *Koordinatensystem* genannt) der Oberfläche des Referenzkörpers in die Planare.

Metadaten
Sind laut ISO 19115 [ISO/TC 211 2003*b*] „*Daten über Daten*“. ISO-Metadaten beschreiben konkrete Geodaten bzw. Geodatendienste und liefern u.a. deren identifizierende und kategorisierende Information, Kontaktdetails zum Datenhalter und -erfasser, Datenqualität und -aktualität. Sie werden in einer XML-Syntax nach ISO 19139 [ISO/TC 211 2007*b*] serialisiert und als separater Informationslayer in Metadatenkatalogen gemäß der OGC-Spezifikation *Catalogue Service - Web* (CS-W) [Nebert et al. 2007] gepflegt.

Prädikat (engl. Property)
Eigenschaft eines Objektes (`Feature`). Besonders im Umgang mit der XML-Sprachfamilie und den XML-Derivaten der *Geography Markup Language* (GML) gebräuchlich. In der GML wird ein Prädikat als *GML-Property* bezeichnet und dient der Verlinkung zweier Objekte - äquivalent zu einem `Prädikat` im Semantic Web.

Spatial object (Synonym: Feature)
Begriff aus dem Projekt INSPIRE, gleichbedeutend mit einem `Feature`. Leider ist der Begriff *Spatial object* in der ISO 19100-Standardserie andersweitig vorbelegt. Dort wird er nur für die geometrische Information, d.h. die rein räumliche Objektbeschreibung, und nicht umfassend für ein Feature gebraucht (siehe ISO 19107 [ISO/TC 211 2003*a*]).

Spatial object type (Synonym: Feature type)
Begriff aus dem Projekt INSPIRE, gleichbedeutend mit einem `Feature type`.

Topologie
Die „*Lehre von der Lage und Anordnung der Gebilde im Raum*“[11]. Topologische Beziehungen beschreiben die Lage jeweils zweier Objekte (`Features`) zueinander, nicht aber deren absolute Positionierung im Raum (siehe `Georeferenzierung`). Beispiele sind die Überschneidung, das Enthaltensein und die Nachbarschaft von Objekten.

[11] Quelle: Duden online; siehe: http://www.duden.de

2.2 Semantic Web, Ontologien

2.2.1 Semantic Web - Grundsätze

Noch bevor der Abschnitt auf Vokabulare, Speicherformen und andere weiterführende Details zu sprechen kommt, soll an dieser Stelle erstmal möglichst scharf umrissen werden, was das Semantic Web prinzipiell auszeichnet. In *Semantic Web* steckt das Wort *Semantik*. Die **Semantik** hat mit der Bedeutung von Zeichen und Symbolen, Wörtern und deren Zusammenhang zu tun. Die Semantik lässt sich unterscheiden von der **Syntax**, mit deren Hilfe sich der reguläre Aufbau von Zeichenketten und höheren Informationseinheiten organisieren und spezifizieren lässt. Die Syntax ist als Grundvoraussetzung dafür anzusehen, dass digitale Ressourcen maschinenlesbar sind. Zeichenketten sind deshalb noch nicht semantisch angereichert. Ob ein XML-Element einen verständlichen Namen besitzt (z.B. *Verkehrsdichte*) oder einen eher kryptischen (z.B. *aux_1_x*) ist - im Gegensatz zur menschlichen Interpretation - für das maschinelle Auslesen im Grunde gleichgültig. Wie also wird einer Zeichenfolge Bedeutung beigemessen?

Während sich der Internetnutzer im *World Wide Web* (WWW) nach interessanten Hyperlinks richtet und selbst darüber entscheidet, von Webseite A nach B zu navigieren, stützt sich das Semantic Web vermehrt auf Maschinen, die eine eigenständige Linkverfolgung im Dienste des Menschen durchführen. Dabei spielen die textuellen, überwiegend rein Menschen-verständlichen Inhalte von Webseiten eine untergeordnete Rolle, vielmehr rücken Daten und deren Verknüpfungen in den Mittelpunkt des Interesses. Hitzler et al. [2008] weist darauf hin, dass das Semantic Web nicht zu verwechseln ist mit dem Forschungsfeld der künstlichen Intelligenz. Letzteres versucht, mit künstlichem Leben neues Wissen zu generieren. Hingegen ist das Semantic Web damit beschäftigt, aus vorhandenen Informationen **implizites Wissen** abzuleiten, d.h. aus einer gegebenen Menge an Faktenwissen selbstständig Schlussfolgerungen herbeizuführen. Ein Beispiel hierzu: aus den Aussagen *Peter ist ein Mann* und *Lukas ist Peters Sohn* kann z.B. geschlussfolgert werden: *Peter ist der Vater von Lukas* und *Lukas ist ein Mensch*. Allerdings trifft dies nur zu, sofern schematische Regeln als Hintergrundwissen zur maschinellen Prozessierung herangezogen werden können, wie z.B. *ein Mann ist ein Mensch* oder *ein Mann, der die Rolle eines Elternteiles ausübt, ist gleichbedeutend mit einem Vater*.

Anhand des Beispieles lassen sich verschiedene Beobachtungen machen. Einerseits müssen die Fakten bzw. Aussagen, wie *Lukas ist Peters Sohn*, in einer maschinenlesbaren Form, also einer definierten Syntax, vorliegen. Andererseits sind die Konzeptwörter, wie *Vater* und *Sohn*, in einen **Bedeutungskontext** zu setzen, so dass Programme befähigt werden, diese Informationseinheiten weiterzuverarbeiten. Beide Herausforderungen oder Hürden, erstens die der Definition von grammatikalischen Regeln als eine Art Basissyntax wie auch zweitens die der Einbettung von Begriffen in einen semantischen Kontext, gilt es zu meistern. Erst dann lässt sich die Vision über einen nahezu grenzenlosen Informationszugriff von WWW-Wegbereiter Tim-Berners-Lee [Berners-Lee et al. 2001] in Ansätzen verwirklichen, dessen Vorschlag aus dem Jahr 2001 die heutigen Strukturen des Semantic Web mitsamt seiner Internetvernetzung und kollaboralen Wissensvermehrung ins Leben rief.

Wenn nun in der weiteren Arbeit von Semantic Web die Rede ist, so ist damit die Eingrenzung auf die Entwicklungsschiene der **W3C-Initiative Semantic Web Activity** gemeint. Es gibt auch andersartige Entwicklungen im Bereich der semantischen Technologien und Wissensspeicherung - siehe z.B. die Sprachen *Frame-Logic*[12] (F-Logic) oder *XML Topic Maps* (XTM)[13]. Jedoch ist die vom W3C eingeleitete Entwicklung diejenige, die in den letzten Jahren die meiste Verbreitung erfuhr und u.a. durch die Linked Open Data Initiative neuen Schub erhält. Die Semantic Web Activity hat bereits verschiedenste Spezifikationen - beim W3C *Recommendations* genannt (dt. Empfehlungen) - hervorgebracht. Am bedeutendsten ist das *Resource Description Framework* (RDF) [Manola & Miller 2004], das mit seinen syntaktischen Regeln die wesentliche Infrastruktur des Semantic Web vorgibt und damit die erste, oben beschriebene Hürde überwinden hilft. Das RDF wird im nächsten Kapitel näher durchleuchtet. Als grammatikalische Erweiterung zu RDF ist vor allem die *Web Ontology Language* (OWL) [Hitzler et al. 2009] zu nennen, die ein exakteres Definieren von

[12] Originalpublikation der Sprachentwickler M.Kiefer, G.Lausen und J.Wu unter: http://www.cs.sunysb.edu/~kifer/TechReports/flogic.pdf
[13] Informationen zu Topic Maps und den zugehörigen ISO-Standardisierungen unter: http://www.topicmaps.org/

schematischem Wissen, den *Vokabularen*, zulässt. RDF- und OWL-**Vokabulare** sind in den letzten Jahren *wie Pilze aus dem Boden geschossen*, es gibt sie für etliche wissenschaftliche und technische Domänen, z.B. für Kontaktdaten in Freunschaftsnetzwerken des Social Web (siehe Projekt *Friend of a Friend*[14], FOAF) oder Verwaltungsinformationen von administrativen Gebietseinheiten des *Ordnance Survey*, der britischen Landesvermessung. Vokabulare bilden im übertragenen Sinne *kleine und größere Sprachinseln im Meer der Kommunikation.* Auf derem Terrain sind Begriffe bzw. die damit benannten Konzepte durch präzise Definitionen und Relationen *auf dem Trockenen.* Werden Ressourcen einer Wissensbasis mit Vokabularkonzepten in Beziehung gebracht, so sind sie semantisch in das Vokabular integriert. Je mehr Verknüpfungen eine Ressource zu anderen Ressourcen aufweist (Realisierung über Prädikate), umso mehr Rückschlüsse lassen andere Ressourcen auf die Natur der ersten zu. Synonyme und Homonyme sind beispielsweise eine häufige Ursache für Verwechslungen und Fehlinterpretation. So kann der Begriff *Werkzeug* unterschiedliche Assoziationen hervorrufen. Man könnte darunter z.B. ein Utensil für den Heimwerkerbedarf oder auch ein Software-Tool verstehen. Sobald aber *Werkzeug* mit den Wörtern *Entwicklungsumgebung* und *Programmierung* näher umschrieben steht oder aber der homonyme Charakter des Wortes durch weitere Aussagen kenntlich gemacht ist, wird die Semantik des Wortes *Werkzeug* besser verständlich. In besonderem Maße trifft dies für den Gebrauch von Konzepten eines Vokabulars zu, sofern dieses vielfache Anwendung in diversen semantischen Werkzeugen und Wissensbasen findet. Vokabulare fungieren also als eine gemeinsame Sprache der jeweiligen Wissensdomäne und bilden neben Verknüpfungen von Begriffen respektive Ressourcen den *semantischen Unterbau* des Semantic Web.

Eine weitere Eigenschaft des Semantic Web ist die **datenzentrische Sicht**, für die auch der Ausdruck *Resource orientated architecture* (ROA) oder *Web der Daten* existiert. Der Zugriff auf Ressourcen ist meist direkt über die Ressourcen-URI und in verschiedenen Rückgabeformaten möglich. Häufig werden einfachste auf dem Internet-Transferprotokoll *Hyper Text Transfer Protocol* (HTTP) aufsetzende Dienste verwendet, die sofortige Aktionen auf die Ressource erlauben, wie das Erstellen, Lesen, Löschen und Aktualisieren. Dieser technische Ansatz wird als *CRUD* (Create, Read, Update, Delete) bezeichnet und nach Webber et al. [2010] vielfach mit dem Modewort und der intelligenteren Software-Architektur des *Representational State Transfer* (REST) verwechselt. Die datenzentrische Sichtweise spiegelt sich auch in den Verknüpfungen der Ressourcen untereinander wieder. Die Ressourcen selbst bilden die Knoten in einem unternehmensweiten oder sogar globalen Wissensnetz. Die Datenfokusierung hat auch Auswirkungen auf die Software-Programmierung. Mit semantischen Formaten kann ein Großteil der Software-Logik ausgedrückt und eingespart werden, was wiederum die semantische Software entlastet und zugleich flexibler für weitere Verwendungsszenarien macht. Damit bieten sich semantische Formate insbesondere für Zwecke der Datenintegration und -zusammenführung an.

In der herkömmlichen Datenverarbeitung und -speicherung werden IT-Systeme mit einer begrenzten Anzahl an Informationen und Datensätzen konfrontiert. Das zeigt sich in der Speicherung von Tupeln in relationalen Datenbanken oder anhand des Transfers von überschaubaren Datenpacketen und zumeist geringen Dateigrößen. Diese Beschränkung auf die Informationen nur eines Datentopfes gleich welchen Formates nennt sich *Closed World Assumption* (CWA), also ein begrenzter Blickwinkel auf die - eigene, selbstgeschaffene oder abstrahierte - Realität. Das Semantic Web versucht in vielen Belangen genau den konträren Standpunkt einzunehmen, gemäß der **Open World Assumption** (OWA). Das bedeutet, dass grundsätzlich angenommen wird, dass die eigene Faktensammlung nicht alle Informationen enthält, somit auch *nicht das letzte Wort gesprochen* oder die allumfassende Beschreibung einer Ressource erbracht werden kann. Irgendwo entfernt im Internet, in dem das Semantic Web naturgemäß fest verankert ist, kann durchaus eine weitere Ressourcen-Zusatzinformation erfasst sein. Vor dem Hintergrund des *Web 2.0* und dank der heutigen Formen gemeinnütziger und freiwilliger Zuarbeit (*Crowd Sourcing*) ist der Gedanke der OWA durchaus nachvollziehbar. Kompromisse gilt es dann einzugehen, sobald eine Datenauswertung zu erfolgen hat, eine Bilanz gezogen bzw. eine entscheidbare Antwort generiert werden soll, wofür in vielen Fällen doch die geschlossene Weltsicht nötig ist. Schließlich kann im Sinne der OWA ein vermeintlicher Sachverhalt, der sich aus lokalem Faktenwissen herauslesen lässt, durch Aussagen

[14] Projektseite: http://www.foaf-project.org/

entfernter Wissensbasen widerlegt werden. Positive Auswirkungen hat die OWA wiederum auf die Erstellung und Nutzung von Vokabularen, indem sie Anlass zu mehr Flexibilität und Wiederverwendbarkeit von Vokabularen gibt.

Ein anderes, nicht weniger unproblematisches Prinzip ist die **Non Unique Name Assumption** (NUNA)[15]. Die NUNA besagt, dass sich unterschiedliche Ressourcenbezeichner - üblicherweise in Form von URIs bzw. IRIs[16] - auf ein und dieselbe Ressource beziehen können. Die gerade angesprochene OWA liefert das Szenario, z.B. wird die Ressource *Berlin* unterschiedlich referenziert durch:

- Wikipedia (Semantic Web-Projekt *DBPedia*[17]): http://dbpedia.org/resource/Berlin
- OpenStreetMap (Semantic Web-Projekt *LinkedGeoData*[18]): http://linkedgeodata.org/page/node240109189
- GeoNames[19]: http://sws.geonames.org/2950159/

Die drei angeführten Identifikatoren lassen sich mit einer Gleichheitsaussage (gebräuchliches Prädikat: *owl:sameAs*) aufeinander abbilden, ebenso können zwei ungleiche Ressourcen mit dem Prädikat *owl:differentFrom* explizit semantisch voneinander getrennt werden. Die dafür notwendigen zusätzlichen Aussagen zum Attestieren der Gleichheit oder Ungleichheit sind aber nur begrenzt sinnvoll, hält man sich die wachsende Anzahl an semantischen Datentöpfen mit großen inhaltlichen Schnittmengen vor Augen.

2.2.2 RDF, RDFS, OWL

Das vorangegangene Kapitel präsentiert das Semantic Web von seiner theoretischen Seite. Nun folgt das praktische Grundgerüst, bestehend aus dem bereits erwähnten *Resource Description Framework* (RDF) [Manola & Miller 2004] als die gemeinsame formale Sprachbasis. RDF liefert ein einfaches, jedoch ausdrucksstarkes Datenmodell, das auf **gerichteten Graphen** beruht. Ein gerichteter Graph besteht aus Knoten und deren verbindenden Kanten. Dessen elementarste Information setzt sich aus zwei Knoten und einer Kante zusammen; auch als **Tripel** bezeichnet. Ein Tripel identifiziert ein Subjekt, ein Prädikat und ein Objekt. Diese Grammatikbausteine sollen anhand der Abbildung 2.1 erläutert werden:

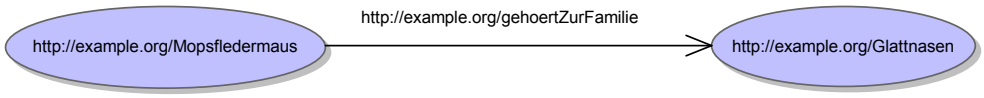


Abbildung 2.1 – Einfachster gerichteter Graph

Das Beispiel zeigt, wie die Zugehörigkeit einer europäischen Fledermausart namens *Mopsfledermaus* zu ihrer Tierfamilie der *Glattnasen* in einem kleinstmöglichen Graphen ausgedrückt werden kann. Alle Graphenbestandteile sind dem beispielhaften Namensraum `http://example.org` zugeordnet, der hier nur zur Vervollständigung des Graphen und zur eindeutigen Benennung der Ressourcen dienen soll. Sowohl `Mopsfledermaus` als auch `Glattnasen` stellen Ressourcen dar, die über das Prädikat `gehoertZurFamilie` in Beziehung gesetzt werden. Die Beziehung (Pfeil-Symbol) ist unidirektional und führt vom Subjekt `Mopsfledermaus` als Gegenstand der Beschreibung zum Objekt `Glattnasen`, das das Subjekt charakterisiert.

[15] der Umkehrschluss heißt *Unique Name Assumption* (UNA) und wird nur in konstruierten Sonderfällen angenommen
[16] Internationalized Resource Identifier: die internationalisierte Form einer URI, die UTF-8 kodierte Sonderzeichen enthalten darf; siehe RfC 3987 [Duerst & Suignard 2005]
[17] Projektseite: http://dbpedia.org
[18] Projektseite: http://linkedgeodata.org
[19] Dokumentation: http://www.geonames.org/ontology/documentation.html

Auf diese Weise lassen sich viele Aussagen zur Ressource `Mopsfledermaus` treffen, indem kurzerhand dem Subjektknoten weitere Knoten-Kanten-Beziehungen respektive Prädikat-Objekt-Paare angefügt werden. Hier offenbart sich die Flexibilität des RDF-Datenmodells. Tripel können willkürlich angeordnet sein, eine Ressource kann sowohl die Rolle eines Subjektes als auch die eines Objektes eingehen. Ein gerichteter Graph ist nicht hierarchisch aufgebaut, wie beispielsweise ein XML-Dokument, sondern als dreidimensionales Netz ohne eigentlichen Ursprungsknoten anzusehen.

Die Abbildung 2.2 enthält alle wesentlichen RDF-Sprachbausteine und einige weiterführende Merkmale des Semantic Web. Insbesondere geht das Diagramm auf den Unterschied zwischen einem Vokabular (TBox) und der restlichen Wissensbasis (ABox) ein. Anstelle der absoluten Schreibweise sind im Diagramm die exemplarischen Namensräume `http://example.org/` (für Individuen) und `http://example.org/concept/` (für Konzepte) durch die Präfixe `ex:` bzw. `excon:` abgekürzt. Zunächst seien die Individuen der **ABox** betrachtet, dargestellt im unteren Diagrammbereich. Der Graph zeigt die Ressource `ex:Mopsfledermaus`, die durch die Eigenschaft `excon:gehoertZurFamilie` in ihrer Familienzugehörigkeit weiterhin als Mitglied der `ex:Glattnasen` ausgewiesen wird. Die Ressource `ex:Glattnasen` ist ihrerseits als Subjekt über das Prädikat `foaf:page` mit einem weiteren Objekt beschrieben, einer Ressource namens `http://de.wikipedia.org/wiki/Glattnasen`, wobei es sich um die Aufrufadresse einer Wikipedia-Webseite handelt. Wie man am Beispiel `ex:Glattnasen` sieht, können **Ressourcen** (ovale Knotennotation) innerhalb von Tripeln entweder als Subjekte oder als Objekte vorkommen und werden dabei mit einer URI angegeben.

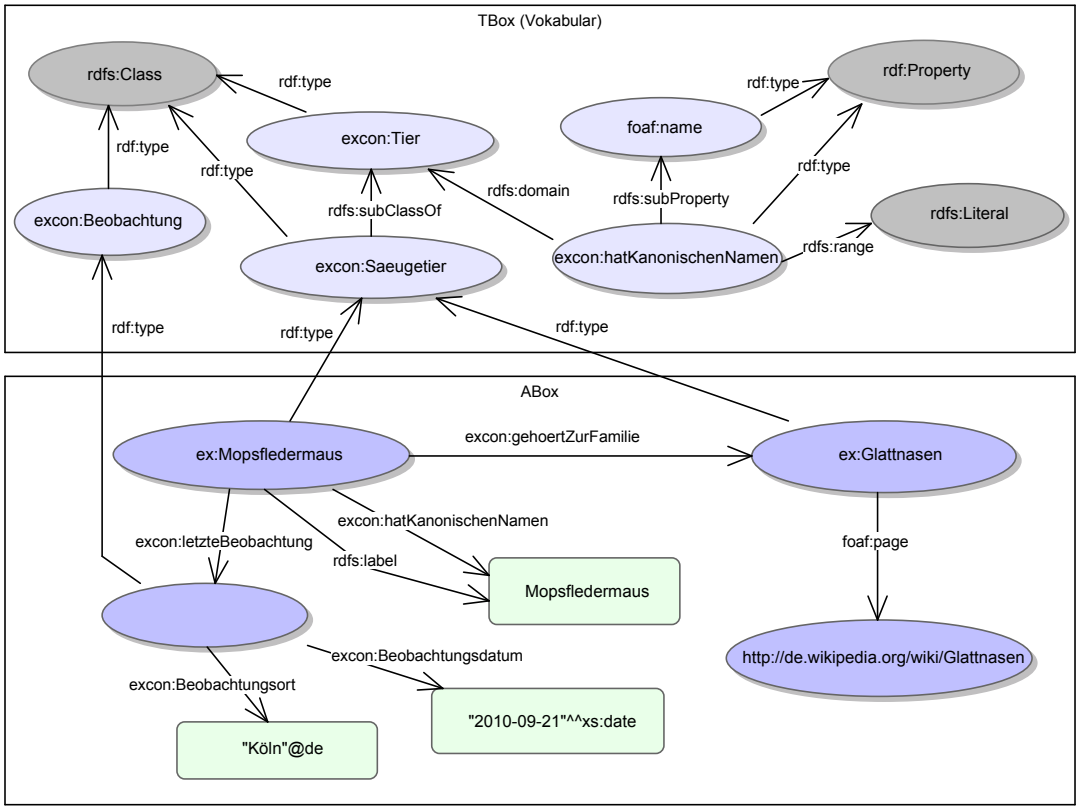


Abbildung 2.2 – Komplexes Graphenbeispiel, aufgeteilt nach TBox und ABox

Literale (rechteckige Knotennotation) enthalten reine Sachattributinformationen und dürfen nur als Objekt geführt werden. Der Datentyp von Literalen lässt sich auf bekannte textuelle, numerische und temporale Datentypen einschränken (sogenannte *typed literal*); so geschehen beim Literal `"2010-09-21"^^xs:date`, das mit dem XML-Schema

Datentyp für Datumsfelder `xs:date` markiert wurde. Darüber hinaus kann einem Literal ein Sprachkürzel (sogenanntes *language tag*) zugewiesen werden, z.B. `@de` zur Kennzeichnung des deutschsprachigen Literals `"Köln"@de`.

Eine Besonderheit in RDF ist der **Blank Node** (dt. *leerer Knoten*). Im Graphenbeispiel taucht ein Blank Node als Objekt des von `ex:Mopsfledermaus` ausgehenden Prädikates `excon:letzteBeobachtung` auf. Solche Blank Nodes dienen in RDF zum Gruppieren und Auflisten untergeordneter Knoten, ohne eine Identifizierung per URI erforderlich zu machen. Hier im Beispiel bündelt ein Blank Node zwei getrennte Informationselemente `excon:Beobachtungsdatum` und `excon:Beobachtungsort` zu einer Informationseinheit, einer Beobachtung, deren Benennung in diesem Graphenkontext nicht von Relevanz ist. Obwohl etliche Semantic Web-Projekte erheblichen Gebrauch von Blank Nodes machen, sollten jene nach Möglichkeit vermieden werden. Die Gründe hierfür werden im Abschnitt 2.2.4 erörtert.

Als nächstes ist die zugehörige Konzeptebene bzw. **TBox** (oberer Diagrammbereich) zu betrachten, die im Graphenbeispiel auf das Nötigste für Erläuterungszwecke reduziert wurde. Auf der linken Seite finden sich drei Ressourcen `excon:Beobachtung`, `excon:Saeugetier` und `excon:Tier`. Sie repräsentieren Konzepte bzw. Klassen (`rdfs:Class`), erkennbar an ihrem Prädikat `rdf:type` mit dem Argument `rdfs:Class`. Die Ressource `rdfs:Class` ist - wie auch alle anderen grau-hinterlegten Ressourcen der TBox - ein originäres Element aus den W3C-Spezifikationen RDF und RDF-Schema [Brickley & Guha 2004]. RDF-Schema legt als gesonderter Teil von RDF die primären Sprachbausteine fest. Die Individuen aus der ABox sind typisiert, z.B. ist `ex:Mopsfledermaus` der Klasse `excon:Saeugetier` zugehörig, welche per `rdfs:subClassOf` als Unterklasse von `excon:Tier` deklariert ist. Derartige Taxonomien - und sogar Mehrfachableitungen - kommen auch für Prädikate in Frage, wie sich dem Beispiel des Tripels `excon:hatKanonischenNamen rdfs:subProperty foaf:name` entnehmen lässt. Ein gängiges Mittel zur Verwendungseinschränkung von Prädikaten ist die Deklaration des Definitionsbereiches mit `rdfs:domain` und des Wertebereiches mit `rdfs:range`. Durch die exemplarischen Definitionsprädikate ist der Gebrauch von `excon:hatKanonischenNamen` auf Subjekte und Objekte der Ressourcentypen `excon:Tier` bzw. `rdfs:Literal` eingeschränkt.

Das Tripel `excon:hatKanonischenNamen rdfs:subProperty foaf:name` macht nebenbei deutlich, dass Vokabularkonzepte (hier `excon:hatKanonischenNamen`) auf **entfernte Konzepte** gründen, von jenen abgeleitet oder mit ihnen gleichgesetzt werden können. Das selbstdefinierte Konzept `excon:hatKanonischenNamen` aus dem Beispielsgraphen ist beispielsweise eine Spezialisierung von `foaf:name`, einem Prädikat, das aus dem Friend of a Friend-Vokabular (FOAF) stammt. Näheres hierzu folgt in Abschnitt 2.2.4.

Wo exakt die **Grenze zwischen Klassen** (Objekttypen) **und Individuen** (Objektinstanzen) verläuft, kommt auf die Zielsetzung der Modellierung an. Die Konzeptklasse `excon:Saeugetier` könnte im Rahmen einer biologischen Taxonomie auch als Instanz geführt werden. Gleichfalls wäre denkbar, die Gattung `ex:Mopsfledermaus` als Konzept für eine zoologische Datenbank zu modellieren und alle in einem bestimmten Zoo gehaltenen Fledermäuse dieser Tiergattung als Instanzen mit Namen oder Kennnummern zu führen. Der Vollständigkeit halber ist zu erwähnen, dass in RDF(S) und in der Spracherweiterung OWL in seiner Ausprägung OWL-Full Klassen nach Belieben auch anstelle von Individuen eingesetzt werden können. Dadurch werden Ressourcen aus unterschiedlichen Perspektiven betrachtet mal als Klassen und mal als Individuen behandelt. Dieses Mittel der Metamodellierung nennt sich *Punning*. Eine derartige Verschmelzung der TBox und der ABox ist aus modelltheoretischer Sicht wie auch aus Performanzgründen nur eingeschränkt sinnvoll. Eine weitere Erläuterung folgt nach einer kurzen Einführung in die Sprache OWL.

Die Wissensrepräsentationssprache **OWL** umfasst alle RDF(S)-Ausdrucksmittel, wie z.B. `rdfs:subClassOf` und `rdf:type`, und fügt neue Ausdrucksmittel hinzu, um Konzepte noch präziser definieren zu können. Mit den erweiterten Sprachmitteln lassen sich Klassen beispielsweise als Schnittmenge, Disjunktion oder Komplement weiterer Klassen definieren. Die Instanzen einer Klasse können mit Hilfe von Wertelisten beschränkt werden. Ebenso lässt sich mit OWL festlegen, dass Klasseninstanzen ausgewählte Prädikate enthalten müssen, wobei die Häufigkeit der Prädikate (Kardinalitätsrestriktionen) als auch deren mögliche Objektwerte (qualifizierende Restriktionen) näher bestimmt werden können. Es sind also umfängliche Rollen- oder Prädikat-Restriktionen für Klassen möglich. Prädikate hingegen lassen sich als transitiv, symmetrisch, invers oder funktional markieren. Die OWL enthält auch Ausdrucksmittel zu

Gleichheits- und Ungleichheitsaussagen für zwei oder mehr Individuen (Prädikate: `owl:sameAs`, `owl:differentFrom`) bzw. Konzepten (Prädikate: z.B. `owl:equivalentClass`, `owl:equivalentProperty`, `owl:disjointWith`). Insgesamt steigert OWL die sprachliche Mächtigkeit von RDF(S) derart, dass sich die Inferenzbildung abhängig von der Größe der Wissensbasis als hochkomplex erweisen kann. Um diesem Problem entgegen zu wirken, wurden drei **OWL-Teilsprachen** voneinander abgegrenzt: OWL-Full, OWL-DL und OWL-Lite. OWL-Full enthält alle OWL-Sprachmittel, ist also mit OWL gleichzusetzen. OWL-Lite wurde zunächst als einfache Einstiegshürde zur Entwicklung OWL-fähiger Programme geplant [Hebeler et al. 2009], hat sich in der Praxis allerdings nicht bewährt, weil OWL-Lite zu wenige OWL-Sprachmittel erlaubt.

Weniger eingrezend als OWL-Lite, aber deutlich restriktiver als OWL-Full ist **OWL-DL** (DL: Description Logic). Wie der Name verrät, ist OWL-DL angelehnt an Beschreibungslogiken. Damit verbunden ist die Einsicht, dass Beschreibungslogiken entscheidbar sind, weshalb Algorithmen der Inferenzbildung in absehbarer Laufzeit terminieren können. Nach Hitzler et al. [2008] „*realisieren Beschreibungslogiken einen Kompromiss zwischen Ausdrucksstärke und Skalierbarkeit*“, wobei mit Skalierbarkeit der Ressourcenbedarf gemeint ist, der für Auswertungen einer wachsenden Eingabemenge, hier der ontologische Aussagenmenge, erforderlich ist. Für die Entscheidbarkeit von Auswerteprozessen hat das W3C einige Kompromisse für OWL-DL beschlossen, u.a. die in der Beschreibungslogik übliche Differenzierung zwischen Klassen und Instanzen. Der wesentliche Vorteil besteht in der separaten Behandlung auf Konzept- bzw. Instanzebene: während auf der Konzeptebene z.B. Klassenkonsistenzprüfungen oder Ableitungen von Klassenhierarchien (*Subsumption-Reasoning*) vorgenommen werden, findet auf Instanzebene z.B. das Schlussfolgern von Ressourcenähnlichkeiten und Konzeptzugehörigkeitsprüfungen statt.[20] Desweiteren lässt OWL-DL keine Typisierungen mit `rdf:Property` zu, sondern unterscheidet in `owl:DatatypeProperty` und `owl:ObjectProperty`, d.h. das Objekt eines Prädikates ist verbindlich entweder ein Literal oder eine weitere Ressource. Weitere Einschränkungen betreffen z.B. die Verwendung von Häufigkeitsrestriktionen im Kontext transitiver oder inverser Prädikate. Die Begrenzung in den Ausdrucksmitteln verhelfen OWL-DL strukturierten Wissensbasen zu einer performanten Auswertung, weshalb sehr viele der heutigen Semantic Web-Werkzeuge OWL-DL konform sind.

In der Version **OWL 2** wurden darüber hinaus drei Sprachprofile zu OWL-DL gebildet: 1. EL 2. QL 3. RL. Sie gruppieren bestimmte Sprachmittel, die 1. großen Taxonomien gelegen kommen, 2. die effektive Suchen über große Instanzdaten erlauben und 3. das regelbasierte Reasoning fördern. Die drei OWL2-Profile sind für die vorliegende Arbeit nicht weiter von Interesse, hingegen ist die pauschale Eingrenzung auf die Sprachmittel von OWL-DL bezweckt, um ein weites Spektrum an Verwendungsmöglichkeiten dank der breiten Software-Unterstützung anzustreben.

Im vorherigen Abschnitt 2.2.1 wurde darauf hingewiesen, dass sich das Semantic Web mit dem Ableiten impliziten Wissens beschäftigt - auch **Inferenzbildung** oder *Reasoning* genannt. Die dazu nötigen Sprachbausteine aus den Sprachfamilien RDF, RDFS und OWL sind oben genannt. Nun fehlt noch das schematische Regelwerk, an dem sich Reasoner-Programme bei der Inferenzbildung orientieren können, um die Wissensbasis um weitere gültige Fakten bzw. Interpretationen der T-Box zu erweitern. Es gibt nicht ein einziges, sondern mehrere jener Regelwerke, die innerhalb der Sprachspezifikationen definiert sind und sich **Entailment-Regime** (kurz: Entailments) nennen. Sie bauen aufeinander auf, indem höhere Entailment-Regime die Ableitungsregeln der untergeordneten Entailment-Regime einbeziehen und um neue erweitern. Die Reihenfolge der Entailment-Regime lautet (von einfach bis komplex): 1. Simple Entailment 2. RDF-Entailment 3. RDFS-Entailment 4. Datatype- oder D-Entailment 5. OWL-Lite-Entailment 6. OWL-DL-Entailment 7. OWL-Full-Entailment. Das Simple Entailment führt Ableitungen herbei, indem allein die identifizierbaren Ressourcen durch Blank Nodes ersetzt werden (z.B. aus `ex:Mopsfledermaus rdfs:label "Mopsfledermaus"` folgt zusätzlich `_:blank1 rdfs:label "Mopsfledermaus"`[21]). Das RDF-Entailment interpretiert u.a. `foaf:name` aus der Aussage `ex:Glattnasen foaf:name "Glattnasen"` als RDF-Property, d.h. es wird die Aussage `foaf:name rdf:type rdf:Property` angefügt. Viel aufschlussreicher sind RDFS- oder OWL-Entailments. RDFS-Entailment er-

[20] Weitere Aspekte zur Unterscheidung zwischen der TBox und der ABox findet man z.B. unter: http://www.mkbergman.com/489/ontology-best-practices-for-data-driven-applications-part-2/

[21] In der Tripel-Notation, die am Ende des Abschnitts vorgestellt wird, ist eine mit '_:' eingeleitete Ressource als Blank Node zu lesen

möglicht bereits das sogenannte *Subsumption-Reasoning* zum Auflösen von Klassenhierarchien, z.B. werden beide Aussagen `ex:Mopsfledermaus rdf:type excon:Saeugetier` und `excon:Saeugetier rdfs:subClassOf excon:Tier` ausgewertet und um `ex:Mopsfledermaus rdf:type excon:Tier` ergänzt. Die OWL-Entailments beachten schließlich die komplexen OWL-Ausdrucksmittel, u.a. die oben angesprochenen Klassen- und Prädikatrestriktionen. Durch die Ableitungsregeln, den *Entailment Rules* wird die Wissensbasis angereichert, so dass deutlich ausgefallenere Fragestellungen möglich sind. Für eine weitere Vertiefung wird auf die Spezifikation RDF-Schema [Brickley & Guha 2004] verwiesen.

Abschließend sollen die in der Praxis geläufigen **RDF-Formatierungen** genannt werden, die natürlich auch OWL-Inhalte als gültiges RDF serialisieren. Weil RDF-Ressourcen willkürlich verlinkt sein können und visuelle Formate zur Massenspeicherung ungeeignet sind, dienen gerichtete Graphen (siehe Abbildung 2.2) nur zu Anschauungszwecken. Ein Tripel stellt die RDF-Basiseinheit dar, deshalb muss eine effiziente Serialisierung RDF-Inhalte in Tripel zerteilen und diese nacheinander speichern. Eine Möglichkeit und auch die offizielle RDF-Formatierung ist das XML-kodierte *RDF/XML*. Im Codebeispiel 1 sind die Konstrukte aus Graphenbeispiel 2.2 als OWL-Konzepte und -Instanzen formatiert in RDF/XML wiedergegeben. Eine andere Formatierung ist die Triple-Notation *N3* oder ihre verwandten und einfacheren Alternativen *N-Triples* und *Terse RDF Triple Language* (Turtle). Letztere erfreut sich im Semantic Web zunehmender Beliebtheit. Ein Auszug, d.h. die OWL-Instanz `ex:Mopsfledermaus` mitsamt ihren Prädikat-Objekt-Verknüpfungen, ist in Turtle-Formatierung dargestellt in Codebeispiel 2. Auffällig sind die Zeilenendezeichen in Tripel-Notationen, die aus einem Punkt, einem Semikolon oder Komma bestehen können. Mit einem Punkt werden gewöhnliche Tripel (Subjekt-Prädikat-Objekt) abgeschlossen, mit einem Semikolon gleiche Subjekte für unterschiedliche Prädikat-Objekt-Paare abgekürzt, ein Komma wiederholt gleiche Subjekt-Prädikat-Kombinationen bei unterschiedlichen Objekten.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY ex "http://example.org/" >
  <!ENTITY excon "http://example.org/concept/" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xs "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY foaf "http://xmlns.com/foaf/0.1/" > ]>
<rdf:RDF xmlns="http://example.org/concept/" xmlns:ex="http://example.org/"
  xmlns:excon="http://example.org/concept/" xmlns:xs="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <owl:Ontology rdf:about="http://example.org/"/>

  <!-- Konzepte -->
  <owl:ObjectProperty rdf:about="&excon;gehoertZurFamilie"/>
  <owl:ObjectProperty rdf:about="&excon;letzteBeobachtung"/>

  <owl:DatatypeProperty rdf:about="&excon;Beobachtungsdatum"/>
  <owl:DatatypeProperty rdf:about="&excon;Beobachtungsort"/>
  <owl:DatatypeProperty rdf:about="&excon;hatKanonischenNamen">
    <rdfs:subPropertyOf rdf:resource="&foaf;name"/>
    <rdfs:domain rdf:resource="&excon;Tier"/>
    <rdfs:range rdf:resource="&rdfs;Literal"/>
  </owl:DatatypeProperty>

  <owl:Class rdf:about="&excon;Beobachtung"/>
  <owl:Class rdf:about="&excon;Tier"/>
  <owl:Class rdf:about="&excon;Saeugetier">
```

```
    <rdfs:subClassOf rdf:resource="&excon;Tier"/>
  </owl:Class>

  <!-- Individuen -->
  <Saeugetier rdf:about="&ex;Glattnasen">
    <rdf:type rdf:resource="&owl;NamedIndividual"/>
    <foaf:page rdf:resource="http://de.wikipedia.org/wiki/Glattnasen"/>
  </Saeugetier>
  <Saeugetier rdf:about="&ex;Mopsfledermaus">
    <rdf:type rdf:resource="&owl;NamedIndividual"/>
    <rdfs:label>Mopsfledermaus</rdfs:label>
    <hatKanonischenNamen>Mopsfledermaus</hatKanonischenNamen>
    <gehoertZurFamilie rdf:resource="&ex;Glattnasen"/>
    <letzteBeobachtung>
      <Beobachtung>
        <Beobachtungsort xml:lang="de">Köln</Beobachtungsort>
        <Beobachtungsdatum rdf:datatype="&xs;date">2010-09-21</Beobachtungsdatum>
      </Beobachtung>
    </letzteBeobachtung>
  </Saeugetier>
</rdf:RDF>
```

Codebeispiel 1 – RDF/XML-Formatierung

```
...
ex:Mopsfledermaus rdf:type excon:Saeugetier ,
                           owl:NamedIndividual ;
                  rdfs:label "Mopsfledermaus" ;
                  excon:hatKanonischenNamen "Mopsfledermaus" ;
                  excon:gehoertZurFamilie ex:Glattnasen ;
                  excon:letzteBeobachtung _:id1 .
_:id1 rdf:type excon:Beobachtung ;
      excon:Beobachtungsdatum "2010-09-21"^^xs:date ;
      excon:Beobachtungsort "Köln"@de .
...
```

Codebeispiel 2 – RDF-Formatierung mit der Triple-Notation Turtle

2.2.3 Kommunikationswege

Die verfügbaren Online-Quellen, die nach Semantic Web-Kriterien strukturiert angeboten werden, lassen sich auf verschiedenen Wegen durchsuchen. Hebeler et al. [2009] identifizieren drei Kommunikationswege eines Anwenders: 1. das Navigieren 2. das Suchen und 3. das Filtern von Semantic Web-Ressourcen. Das Navigieren wird auch als **RDF-Browsing** bezeichnet, weil hierzu sogenannte *RDF-Browser* eingesetzt werden, ob als Plugin regulärer Browser, z.B. *Tabulator*[22] für den Mozilla Firefox-Browser, oder als Web- oder Desktopapplikationen. RDF-Browser stellen Tripel-Informationen und ganze RDF-Sammlungen anschaulich dar und heben Ressourcenverlinkungen (URIs) hervor. Die Verlinkungen wiederum laden wie herkömmliche Internet-Hyperlinks zum Verfolgen bzw. Navigieren zu verknüpften Inhalten ein. Trotz der graphischen Aufbereitung, die leichter zugänglich ist als die Serialisierungen aus dem vorangegangenen Abschnitt 2.2.2, ist das RDF-Browsen recht unübersichtlich, weil das Semantic Web als reines Datennetzwerk im Gegensatz zum klassischen Internet auf kontextbezogenes Layout verzichtet. Gewohnter ist dagegen die **Semantische Suche** über Semantic Web-Suchmaschinen vergleichbar zu *Google*, wie z.B. Swoogle[23]

[22] Projektseite: http://www.w3.org/2005/ajar/tab
[23] Swoogle-Rechercheoberfläche: http://swoogle.umbc.edu/

oder Sindice [24]. Beide semantischen Suchmaschinen durchkämmen das Internet nach Semantic Web-Inhalten (RDF-Datensammlungen, OWL-Ontologien) und indexieren diese textuell. Inferenzbildungen sollten bei den riesigen Online-Datenmengen jedoch ausgeschlossen sein. Der Hauptverwendungszweck von semantischen Suchmaschinen ist deshalb nach Hebeler et al. [2009] auch eher darin zu sehen, mittels einer Stichwortsuche nach den bereits verfügbaren Vokabularen bzw. ontologischen Konzepten recherchieren zu können, mit dem Ziel, diese Konzepte nachzunutzen.

Sofern die Datenquelle bekannt ist, wird in der Regel die dritte Kommunikationsart gewählt: das gezielte Filtern über deklarative Anfragesprachen, auch als **RDF-Querying** bezeichnet. Das Semantic Web hat diverse Anfragesprachen hervorgebracht, z.B. *OWL-QL*[25] oder *SeRQL*[26]. Für die Abfrage von RDF/RDFS/OWL-Datenquellen hat sich die Anfragesprache *SPARQL*[27] [Prud'hommeaux & Seaborne 2008] als De-facto-Standard etabliert. Die Beliebtheit von SPARQL lässt sich u.a. auf die vielen im Internet kursierenden RDF-Datenbanken (*RDF-Triplestores*) zurückführen. Dank der leicht verständlichen SPARQL-Syntax, die eine große Ähnlichkeit mit der bekannten Datenbanksprache *Structured Query Language* (SQL) hat, können RDF-Datenbanken recht unkompliziert mit Anfrageschnittstellen ausgestattet werden, den sogenannten *SPARQL-Endpoints*. Um dieser Fokusierung in der Semantic Web-Gemeinde gerecht zu werden, beschränkt sich nun die weitere Betrachtung auf SPARQL.

Anfragen in **SPARQL** führen auf einer gegebenen Menge an RDF-Tripeln Graphenvergleiche durch, d.h. sie erkennen Tripel und ihre Verknüpfungen als *Graphenmuster* (engl. Graph Pattern). Die stabile SPARQL-Version 1.0 aus dem Jahr 2008 lässt den Implementierungen offen, ob während der Anfrage Inferenzbildungen auf dem Graphen der Wissensbasis stattfinden oder ob nur die reinen RDF-Tripel als Bestandteile des Graphen durchsucht und verglichen werden.[28] Die Graphenvergleiche führen zu temporären mengenorientierten Zwischenergebnissen (*SPARQL-Solutions* oder *RDF-Tupel*), die sich wiederum mit Hilfe von Mengenoperatoren (SPARQL-Operatoren und -Filter) weiter verschneiden und filtern lassen. SPARQL-Ergebnisse (*Bindings*) werden je nach Anfrageart - den sogenannten *SPARQL Query forms* - unterschiedlich ausgegeben. Es gibt die vier Anfragearten `Select`, `Construct`, `Describe` und `Ask`, wobei die ersten beiden die gebräuchlichen sind. Mit der Anfrageart `Select` erhält man Resultate in einer relationalen Form als XML-Format kodiert (Format: *SPARQL Query Results XML Format*). `Construct` hingegen stellt abhängig von der Benutzereingabe aus den Anfrageresultaten neue RDF-Tripel zu einem Ergebnisgraphen zusammen, dessen Format somit regulärem RDF entspricht (z.B. in der Serialisierung RDF/XML).

Beispiel 3 zeigt eine SPARQL-Anfrage, die eine RDF-Datenhaltung nach Konzepten und Individuen aus Abbildung 2.2 durchsucht. Die Zeilen 1 bis 4 bilden RDF-Namensräume auf Präfixe ab, die in den weiteren Fragekonstrukten Verwendung finden. In Zeile 6 wird die SPARQL-Anfrageart `Select` gewählt und wie in SQL eine `Select`-Klausel angegeben, die die gewünschten Ausgabevariablen benennt. Mit der optionalen `From`-Klausel kann ein Standard-Graph als Suchziel festgelegt werden, meistens beinhaltet die RDF-Datenhaltung aber bereits einen Standard-Graphen und die `From`-Klausel kann entfallen. Ist die RDF-Datenhaltung in mehrere Graphen unterteilt, können bestimmte Graphen mit `From Named`-Klauseln ausgewählt bzw. unterschieden werden, vergleichbar den Tabellen in der SQL `From`-Klausel. Die Zeilen 9 bis 13 geben nun das Graphenmuster vor, dem alle Lösungen entsprechen sollen. Das Graphenmuster besteht aus mehreren Tripelmustern, die Variablen enthalten. Die Variablen-Notation beginnt mit einem Fragezeichen, z.B. `?tier`. Das hier dargestellte Graphenmuster filtert alle Tierarten, die vom Typ `excon:Saeugetier` sind (Zeile 9), zur Tiergattung der `ex:Glattnasen` gehören (Zeile 10), einen kanonischen Namen haben (Zeile 11) und an einem bestimmten Datum beobachtet wurden (Zeile 12/13). Ein optionales Graphenmuster wird mit dem `Optional`-Operator eingeleitet (Zeile 14). Hier im Beispiel 3 werden optional Ortsangaben auf die Ergebnis-Variable `?ort` abgebildet, sofern für die Tierarten, die zuvor mit dem verpflichtenden Graphenmuster aufgefunden wurden, auch Informationen zu Beobachtungsorten erfasst sind. Sind schließlich alle Tripel durch die Graphenmuster ermittelt und mengenorientiert

[24] Sindice-Startseite: http://sindice.com/
[25] Projekt des Stanford Knowledge Systems Laboratory; siehe: http://www.ksl.stanford.edu/projects/owl-ql/
[26] Filtersprache des Semantic Web-Frameworks *Sesame* der Firma *Aduna*; Tutorial: http://www.openrdf.org/doc/sesame2/users/ch09.html
[27] beim Namen *SPARQL* handelt es sich um ein rekursives Akronym: **S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage
[28] die momentan in Bearbeitung befindliche Nachfolgeversion SPARQL 1.1 [Harris & Seaborne 2012] enthält neuerdings explizite Verweise auf SPARQL Entailment-Regime

zwischengespeichert, können relationale Mengenoperatoren, wie z.B. `Union` oder `Join`, angewandt werden. Ebenso lassen sich die gefundenen Tripel, insbesondere RDF-Literale mit Sachattributinformationen, durch `Filter`-Operatoren filtern, wie dies z.B. in Zeile 15 mit der Angabe des Beobachtungsdatums geschieht. Dazu wird der Pfad der Variable `?datum` aufgelöst, indem das Graphenmuster mit den Pfadvergleichen in den Zeilen 12 und 13 die benötigte Datumsinformation in der RDF-Datenhaltung identifiziert. Abschließend können sogenannte *SPARQL-Ergebnismodifikatoren* (engl. *Solution modifier*) eingesetzt werden, wie z.B. `Limit` (Zeile 17) zur Begrenzung der Lösungsmenge oder eine `Order By`-Klausel zum Sortieren der Lösungen.

```
1  @prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix excon:  <http://example.org/concept/> .
3  @prefix ex:     <http://example.org/> .
4  @prefix xs:     <http://www.w3.org/2001/XMLSchema#> .
5
6  SELECT ?name ?datum ?ort
7  FROM <http://www.ex.org/graph1>
8  WHERE{
9    ?tier  rdf:type  excon:Saeugetier .
10   ?tier  excon:gehoertZurFamilie  ex:Glattnasen .
11   ?tier  excon:hatKanonischenNamen  ?name .
12   ?tier  excon:letzteBeobachtung  ?beobachtung .
13   ?beobachtung  excon:Beobachtungsdatum  ?datum .
14   OPTIONAL { ?beobachtung  excon:Beobachtungsort  ?ort } .
15   FILTER ( ?datum > "2010-01-01T00:00:00Z"^^xs:dateTime )
16 }
17 LIMIT 5
```

Codebeispiel 3 – Beispiel einer SPARQL-Anfrage

Richtet man die Anfrage aus Beispiel 3 gegen den Graphen in Abbildung 2.2, so erhält man eine einzelne Lösung mit drei Variablen-Bindings: ?name = `"Mopsfledermaus"`, ?datum = `"2010-09-21"^^xs:date` und ?ort = `"Köln"@de`.

Während des Auswertevorganges wird die SPARQL-Syntax der Benutzeranfrage (Beispiel 3) geparst und umgeformt in Ausdrücke der sogenannten **SPARQL-Algebra**. Die SPARQL-Algebra definiert die formale Semantik von SPARQL und enthält Ausdrücke, aus denen exakte Berechnungsanweisungen zur Reproduzierbarkeit von SPARQL-Ergebnissen gefolgert werden können. Die Überführung der syntaktischen Anfrageform (Benutzeranfrage) in die SPARQL-Algebra ist wohldefiniert und liefert eine Anfrage, die strukturell optimiert ist, indem z.B. URI-Präfixe aufgelöst und zusammengehörige Graphenmuster optimal gruppiert sind. Die SPARQL-Algebra Operatoren ähneln jenen der Relationalen Algebra, auf der insbesondere SQL beruht. Beispielsweise gleichen die SPARQL-Algebra Operatoren `Join` und `Left Join` den Operatoren `Equi Join` bzw. `Left Outer Join` der relationalen Algebra. Für eine tiefergehende Betrachtung wird auf die SPARQL-Spezifikation [Prud'hommeaux & Seaborne 2008] Kapitel 12 *Definition of SPARQL* verwiesen. Ein beispielhafter SPARQL-Auswertevorgang inklusive der Umformung in die SPARQL-Algebra ist in Anhang A wiedergegeben.

2.2.4 Vernetztes Wissen

Das Semantic Web hat diverse Ontologien bzw. Vokabulare hervorgebracht, die globale oder auch nur partielle Wissensbereiche mit Konzepten modellieren. In der Regel werden drei **Vokabulartypen** unterschieden: 1. *Upper-Ontologien* 2. *Domänenontologien* und 3. *Applikations-* oder *Problemontologien*. Upper-Ontologien (auch *Top-Level-* oder *Foundation-Ontologien* genannt), wie bspw. die *Descriptive Ontology for Linguistic and Cognitive Engineering*

(Dolce[29]) oder Cyc[30], bilden übergeordnete Vokabulare, die auf philosophischen Ansätzen beruhen und eine globale Sicht der Dinge darstellen. Sie umfassen meist mehrere tausend Konzepte, die von einer abstrakten, theoretischen Ebene bis hin zur Modellierung alltäglicher Objekte der Real- und Gedankenwelt reichen. Im Gegensatz dazu werden Domänenontologien entwickelt, um die Konzepte eines Wissensgebietes (Domäne), wie z.B. die Hydrologie, die Geologie oder die multimediale Informationstechnik, zu erfassen, damit Fachbegriffe und Semantiken innerhalb der jeweiligen Wissensgemeinde eindeutig festgelegt sind. Zu den bekanntesten Domänenontologien zählen sicherlich das Vokabular *Friend of a Friend* (FOAF)[31] zur Beschreibung von Freundschaftsverhältnissen und Kontaktdaten, das Vokabular *Dublin Core* (DC)[32] zum Erfassen von Publikationsmetadaten und die *Time-Ontologie*[33] für zeitliche Zusammenhänge. Applikationsontologien bleiben stattdessen einem sehr beschränkten Anwenderkreis, z.B. einem einzelnen Unternehmen, vorbehalten und dienen im Rahmen einer benutzerspezifischen Problemstellung mit programmatischer Lösung als eine Art semantische Datengrundlage.

Wie eingangs im Abschnitt 2.2.1 erläutert, sind **Verlinkungen** im Semantic Web essentiell, um den semantischen Gehalt von Ressourcen zu steigern und dank begrifflicher Verknüpfungen maschinengestützte Auswertungen zu erleichtern. Die Wissensvernetzung kann entweder über verknüpfte Vokabularkonzepte oder Instanzen stattfinden. Die Realisierung auf Instanzebene wird gegen Ende dieses Abschnittes insbesondere am Beispiel *Linked Open Data* betrachtet. Vokabularkonzepte werden miteinander verknüpft oder aufeinander gemappt, indem OWL-Sprachmittel explizit deren Gleichheit, z.B. `<http://ex.org/concept1> owl:equivalentClass <http://ex2.org/conceptA>`, oder Spezialisierung attestieren, z.B. `<http://ex.org/concept1> rdfs:subClassOf <http://ex2.org/conceptA>`. Mit derartigen Mapping-Aussagen können zwei und mehr Domänen- oder Problemontologien direkt aufeinander abgebildet werden, falls eine bilinguale Transformation erwünscht ist. Es kann aber auch der *Umweg* über eine Upper-Ontologie bestritten werden, die dann als Bindeglied für die semantische Interoperabilität mehrerer Ontologien genutzt wird. Das ontologische Mapping wird häufig auch als **Ontologie-Alignment** bezeichnet.

Wie Domänenontologien, so bilden herkömmlicherweise Taxonomien und Thesauri Ordnungssysteme für Fachbegriffe einer Wissensdomäne. Weil kontrolliertes Namensgut in INSPIRE eine wichtige Rolle spielt, folgt hier nun eine kurze Exkursion zu diesem Thema. Im Allgemeinen lassen sich Taxonomien und Thesauri als simple Ontologien einstufen, da sie selten mehr als zwei Verknüpfungsrelationen - Synonyme und Ähnlichkeit - und meistens nur eine primäre Hierarchie ihrer Begriffe, den *Deskriptoren*, aufweisen. Die Deskriptoren werden als einfache Referenzinformation im Sinne eines Suchindexes, eines Ordnungskriteriums oder zur Übersicht über die Begriffswelt der Domäne gebraucht. Nicht aber dienen sie zur formalen ontologischen Beschreibung eines Konzeptes und zum Hinzufügen neuer Aussagen, so wie es die OWL vorsieht. Dennoch bieten sich derartige kontrollierte Vokabulare als Referenzinformationen zur Verknüpfung mit Semantic Web-Ressourcen an. Mit dem **Simple Knowledge Organization System** (SKOS) [Miles & Bechhofer 2009] hat das W3C ein OWL-Vokabular abgestimmt, das einen praktikablen Migrationspfad von Thesauri-Inhalten zu OWL-Individuen aufzeigt - nicht zu OWL-Klassenkonzepten aus besagten Gründen. Die mit den SKOS-Sprachmitteln gepflegten OWL-Individuen sind typisiert (mit `rdf:type`) als a) `skos:ConceptScheme` oder b) `skos:Concept`, d.h. sie entsprechen entweder a) einem Thesaurus, einer Taxonomie bzw. einem Klassifikationsschema oder b) deren Fachbegriffen bzw. Deskriptoren. Diverse Prädikate werden eingeführt, und zwar für die Namensgebung der Fachbegriffe (z.B. `skos:prefLabel`, `skos:altLabel`), für die Hierarchisierung der Fachbegriffe untereinander (z.B. `skos:narrower`, `skos:broader`, `skos:hasTopConcept`), ihre Zuordnung zum jeweiligen kontrollierten Vokabular (`skos:inScheme`) und Relationen zwischen den Fachbegriffen verschiedener Vokabulare (z.B. `skos:narrowMatch`, `skos:exactMatch`). SKOS erlaubt also die Überführung von kontrolliertem Namensgut in eine Semantic Web-Repräsentation mit einfachsten standardisierten Mitteln und ist demnach prädestiniert für die Semantic Web-Erschließung von ISO-Registern wie auch INSPIRE-Wortschätzen. Mehr dazu folgt in Abschnitt 5.1.3.

[29] Dokumentation: http://www.loa-cnr.it/DOLCE.html
[30] abgeleitet aus dem engl. en**cyc**lopedia; siehe: http://www.cyc.com/
[31] Projektseite: http://www.foaf-project.org/
[32] Dokumentation: http://dublincore.org/documents/dcmes-xml/
[33] Dokumentation: http://www.w3.org/TR/owl-time/

Das Projekt **Linked Open Data** (LOD) bezweckt die Online-Verfügbarmachung und Wissensvernetzung gemeinnützig erfasster oder behördlich freigestellter Wissensdatenbanken, die zusammen als *LOD-Cloud* bezeichnet werden.[34]. Die gegenseitige Verknüpfung der Datentöpfe erfolgt auf Instanzebene in den meisten Fällen per OWL-Gleichheitsaussage `owl:sameAs`, aber auch durch RDF(S)-Prädikate (z.B. `rdfs:isDefinedBy`, `rdfs:seeAlso`) oder beispielsweise mittels gebräuchlicher Kontaktverlinkungen des FOAF-Vokabulars (z.B. `foaf:knows`, `foaf:based_near`). Linked Open Data stützt sich dabei lediglich auf die RDF-Semantik, ohne RDFS/OWL Schema-Semantiken einzubeziehen bzw. zu fordern, so dass ontologische Schlussfolgerungen selten möglich sind [Jain et al. 2010]. Der Linked Open Data-Initiative ist eher an der einfachen Verfügbarmachung von Daten gelegen. In seinen *Design Issues* als eine Art LOD-Präambel notiert Tim Berners-Lee vier wesentliche Regeln[35] zur LOD-Datenveröffentlichung: 1. verwende URIs (eindeutige Bezeichner) für Semantic Web-Ressourcen 2. verwende HTTP-URIs, so dass URIs aufgelöst werden können 3. werden URIs aufgelöst, so liefere an der Zieladresse sinnvolle Ressourcenbeschreibungen in standardisierter Form (RDF, SPARQL) und 4. füge Verlinkungen zu anderen Semantic Web-Ressourcen ein, so dass dorthin navigiert und darüber weitere Informationen bezogen werden können.

Diese vier Regeln sind aber nur die primären LOD-Prinzipien, andere Prinzipien als auch Best-Practice-Empfehlungen stehen zusammengefasst im How-To Leitpfaden [Linking Open Data Community Project 2007]. Dort wird insbesondere logisch differenziert in *Informationsressourcen* und *Nicht-Informationsressourcen*. Während letztere die realen Dinge mit URIs identifizieren, sollen Informationsressourcen zugehörige Beschreibungen (sogenannte *Associated Descriptions*) liefern, und zwar in den üblichen Formaten (MIME-Typen) HTML, RDF/XML, RDF-Turtle etc., wobei RDF/XML obligatorisch ist. Benutzeranfragen auf Nicht-Informationsressourcen werden über das sog. **Content Negotiation** (dt. *Aushandeln der Inhalte*) auf Informationsressourcen abgebildet. Der angefragte Datenserver antwortet vorzugsweise mit einer HTTP 303 *See Other*-Statusmeldung und weist den Client damit auf die zugehörige Informationsressource bzw. deren Adresse hin. Der Client befolgt den Redirect und nennt im HTTP-Header der neuerlichen Anfrage das gewünschte Benutzerformat, das die HTTP 303-Nachricht zur Auswahl stellt. Das Content Negotiation setzt voraus, dass sowohl die (Nicht-Informations-)Ressource als auch deren Beschreibungen jeweils mit eigenen URIs identifiziert werden (z.B. `http://dbpedia.org/resource/Berlin` für die Nicht-Informationsressource und `http://dbpedia.org/page/Berlin` für deren Beschreibung als HTML-Seite).

Weitere LOD-Prinzipien schränken den Gebrauch von RDF-Sprachmitteln ein, z.B. wird ein Verzicht von Blank Nodes propagiert, da die eindeutige Ressourcenidentifikation charakteristisch für Linked Data ist. Blank Nodes enthalten jedoch per Definition keine URI, d.h. sie lassen sich nicht direkt verknüpfen bzw. veröffentlichen. Dennoch gibt es etliche LOD-Datenprojekte, die viele ihrer Ressourcen aus fachlichen Gesichtspunkten als Blank Nodes pflegen und diesem Mangel gegebenenfalls mittels synthetischer URIs, generiert aus Hash-Werten, beizukommen versuchen. Desweiteren wird von Axiom-Annotationen (Stichwort: *Reifikation*) oder z.B. RDF-Collection/Container und damit von Mitteln der Metamodellierung und weitschweifigen RDF-Sprachkonstrukten abgeraten, die speziell im Umgang mit SPARQL und Graphenvergleichen hinderlich sind. Das LOD How-To unterbreitet Vorschläge, wie statische RDF-Dateien, RDF-Datenbanken oder fremde Formate und Dienste mittels Wrapper-Anwendungen LOD-fähig gemacht werden können. Das Ziel sind entweder Anfrageschnittstellen für den Navigationsfall (sogenannte *Linked Data-Endpoints*) oder jene für Filteranfragen (LOD adressiert hierfür im Wesentlichen *SPARQL-Endpoints*; siehe Abschnitt 2.2.3). Die Linked Open Data Initiative ist als sehr erfolgreicher Ansatz anzusehen, einfachste Datenverfügbarkeit im Semantic Web zu erreichen. Trotz mancher Schwächen, z.B. hinsichtlich der mangelnden schematischen Unterstützung oder fehleranfälligen Bemühungen zur automatisierten Ressourcenverlinkung, ist mit Linked Data der bisher weitreichendste Impuls und Anreiz für Semantic Web-Datenbereitsteller wie -Anwender geglückt und soll in dieser Arbeit besondere Beachtung finden.

[34] Projektseite: http://linkeddata.org/; Übersichtsdiagramm der LOD-Cloud: http://richard.cyganiak.de/2007/10/lod/; der Name *Linked Open Data* wird häufig auch mit *Linked Data* abgekürzt

[35] zu finden auf der LOD-Projektseite: http://www.w3.org/DesignIssues/LinkedData.html

2.3 Geo Web, Geodateninfrastrukturen

2.3.1 Geo Web - Einführung

Das Geo Web beschäftigt sich mit der Auslagerung von Desktop-gestützter GIS-Funktionalität auf internetfähige Geodienste. **Geographische Informationssysteme** (GIS) sind seit den 70er Jahren im Gebrauch, um u.a. die Anwendungsfelder Kartographie, Kataster und Planungswesen mit digitalen Werkzeugen der Erfassung, Speicherung, Analyse, Aufbereitung und Visualisierung von Geoinformationen voranzutreiben. GIS-Software bestand bis in die 90er Jahre aus recht monolithisch aufgebauten Desktop-Applikationen, die dem Datenaustausch hinderlich waren und dementsprechend auch zu Insellösungen bzgl. der Transfer- und Speicherformate führte. Eine bekannte Open-Source Initiative zwecks einem kostenfreien, Quell-offenen GIS namens *GRASS GIS*[36] konnte diese geschlossene Systemwelt auch nicht zufriedenstellend revolutionieren. Die rein technisch-offene Lösung konnte den nötigen inhaltlichen Abstimmungsbedarf innerhalb der GIS-Anwendergemeinde nicht erübrigen.[37] Mittlerweile verfolgt man mit dem Geo Web eine andere Strategie, die gemeinschaftliche Standardisierungsprozesse vorsieht. Dabei gilt die Aufmerksamkeit den webfähigen Geodiensten, deren Web-Schnittstellen z.B. für die Abfrage von Kartenbildern oder Sensordaten spezifiziert werden. Dadurch lässt sich sowohl Open-Source als auch kommerzielle, neue wie bereits bestehende GIS-Software zur Sicherung getätigter Investitionen mit normierten Schnittstellen ausstatten und in eine Architektur von intern oder öffentlich zugänglichen Geodiensten einklinken, sogenannten Geodateninfrastrukturen (GDI).

Das ähnelt dem Grundgedanken des Internet-Hypes *Service Orientated Architecture* (SOA) mit der Verkettung von Unterstützerdiensten für die Abbildung von Geschäftsprozessen. Zwar eint SOA und Geo Web prinzipiell der **Ansatz der verteilten Dienstelandschaft**, in der Praxis beschränkt sich SOA jedoch meist auf Firmennetzwerke, während das Geo Web heutzutage über jegliche Systemgrenzen hinweg Geodienste bereitstellt. Andererseits ist das Geo Web in der jetzigen Ausprägung noch stark auf die Vernetzung von Datenquellen und deren Recherche ausgelegt und beinhaltet weniger Diensttypen für die in der SOA bedeutungsvollen Funktions- und Prozessschichten. Das Geo Web ist grundlegend mit dem Anwendungsmuster *Publish/Find/Bind/Chain* [Bernard et al. 2005] verbunden, das eine logische Handlungskette zum Datenaustausch ausdrückt. Danach werden Metadaten zur Beschreibung der Geo-Ressourcen (Geodaten & Geodienste) veröffentlicht (*Publish*), die der Recherche der Geo-Ressourcen dienen (*Find*) und jene zur unmittelbaren, sinnvollen Anwendung in GIS-Clients geladen werden können (*Bind*). Zusätzlich gibt es die seltener unterstütze Aktion *Chain*, die auf Unterstützungsdienste und Wertschöpfungsketten hinweist. Der klassische Anwendungsfall für eine Wertschöpfungskette ist eine Kaskadierung von Kartendiensten, deren Produkte je Verarbeitungsschritt mit weiteren Informationslayern im *Overlay*-Verfahren angereichert werden. Ein anderer Wesenszug des Geo Web ist der Verbleib der originären Geodaten beim Datenerfasser bzw. -publizierer und der damit einhergehenden qualitativ wertvolleren Datenpflege durch den Urheber selbst. Dadurch kann nicht zuletzt das Datenmanagement effizienter und kostengünstiger gestaltet werden. Ebenso lassen sich Redundanzen und Synchronisierungsaufwände vermeiden.

Die in der Praxis noch unterrepräsentierte Funktions- und Prozesschicht erlebt derzeit einen deutlichen Entwicklungsschub. Hier sind vor allem die Web Processing Services (WPS) zu nennen, die allgemeine oder fachspezifische GIS-Analyseprozeduren anbieten. Speziell zur Rasterdatenverarbeitung ist seit 2009 die genormte Schnittstelle des Web Coverage Processing Service (WCPS) verfügbar. Ebenso gibt es bereits Pilotprojekte zur Orchestrierung von verschiedensten Geodiensten, z.B. für das Szenario des Katastrophenmanagements [Weiser & Zipf 2007]. Eine **inhaltliche Erweiterung** erfährt das Geo Web im Laufe der letzten Jahre auch auf dem Gebiet der Datenmodelle und -formate. Die im Geo Web gebräuchlichen XML-basierten Transferformate entwickeln sich von flach-strukturierten Einzelformaten hin zu komplexeren und fachlich-differenzierten Formaten als Ergebnis langwieriger Konsensbildung. Ein gutes Beispiel hierfür ist das Projekt INSPIRE (siehe Abschnitt 2.5).

[36] Projektseite: http://grass.fbk.eu/
[37] Quelle: Kurzer geschichtlicher Abriss des OGC; siehe: http://www.opengeospatial.org/ogc/history

Treibende Kräfte hinter dem Geo Web sind das Open Geospatial Consortium (OGC) und die International Organization for Standardization (ISO). Das OGC ist ein 1994 gegründetes, gemeinnütziges Industriekonsortium, das sich derzeit aus 445 weltweit agierenden Mitgliedern (Stand: Januar 2012[38]) aus den Bereichen Forschung, Wirtschaft und Regierungsinstitutionen zusammensetzt. Es erarbeitet in Spezifierungsprogrammen freie De-facto-Standards für Dienstschnittstellen, Datenmodelle und -kodierungen im Bereich der Geodatenverarbeitung. Ausserdem prüft das OGC in Interoperabilitätsprogrammen neue Technologien und deren Einsatz in realen Szenarien mit existierenden Bordmitteln der OGC-Spezifikationen. Zur Abstimmung mit anderen Standardisierungsorganisationen geht das OGC strategische Partnerschaften ein, so z.B. die sehr weitreichende Liaison mit der ISO, speziell dem ISO Technical Committee (TC) 211. Das ISO TC 211 beschäftigt sich ebenfalls mit der Geodatenverarbeitung und verabschiedet De-jure-Standards (Normenreihe 19100), die thematisch nicht exakt mit den OGC-Spezifikationen deckungsgleich sein müssen. Doch dank der engen und eher ergänzenden Zusammenarbeit, mit zahlreicher personeller Beteiligung in beiden Gremien, werden OGC-Spezifikationen häufig 1:1 auch als ISO-Standards übernommen. Im Gegenzug führt das OGC grundlegende ISO-Standards in Auszügen bzw. in fortgeschrittenem Stadium ein. Diese dienen neben anderen Positionierungsdokumenten des OGC unter der Bezeichnung *Abstract Specification*[39] zur Orientierung für die Entwicklungsarbeiten an den *Implementation Specifications*, den Spezifikationen für Dienstschnittstellen und Geodatensprachen.

2.3.2 Datenmodellierung

Der zentrale ISO-Standard für Modellierungszwecke ist ISO 19109 *Geographic information – Rules for application schema* [ISO/TC 211 2005*a*]. Er schafft Regularien für die Erstellung von Profilen standardisierter Datenmodelle und davon abgeleiteter Anwendungsschemata (*Rules for application schema*). So schreibt ISO 19109 vor, eine formale Sprache zur Modellierung einzusetzen, empfohlen ist die UML. Darüber hinaus wird in ISO 19109 das Meta-Modell **General Feature Model** (GFM) aufgestellt, auf das konzeptionell jedes Datenmodell der Normenreihe 19100 basiert. Das GFM definiert für Feature Types (`GF_FeatureType`) Eigenschaften (`GF_PropertyType`), die über Attribute, Operationen oder Assoziationen realisiert werden können. Bei den Attributen (`GF_AttributeType`) kann es sich um a) zeitliche, b) räumliche, c) im Sinne eines indirekten Raumbezuges lokalisierende, d) Metadaten- und e) thematische Attributtypen handeln. Mit Ausnahme der thematischen Attribute verweist das GFM für jeden Attributtyp auf die jeweilige Ausgestaltung in vier weiteren ISO-Standards, so z.B. auf das *Temporal Schema* in ISO 19108 [ISO/TC 211 2002*b*] zur Beschreibung zeitlicher Charakteristiken oder auf das für die Geodatenverarbeitung immanente Thema der Geometriemodellierung in ISO 19107 *Spatial Schema* [ISO/TC 211 2003*a*]. Da die Geodatenverarbeitung bzw. -speicherung für die vorliegende Arbeit auch im Kontext des Semantic Web von besonderem Interesse ist, werden im Folgenden die von OGC und ISO entworfenen Geometriemodelle anhand der Abbildung 2.3 vorgestellt.

Parallel zum GFM exisitert beim OGC ein Pendant namens *OGC Abstract Specification Topic 5 Feature*. Die beiden Basismodelle sind zwar nicht harmonisiert, sie referenzieren jedoch gemeinsam das **ISO 19107 Spatial Schema**. Das Spatial Schema definiert ein Geometrietypenmodell (*Geometry Packages*), d.h eine Hierarchie an Geometrietypen bzw. -klassen mit a) geometrischen Primitiven: u.a. Punkte, Kurven, Obeflächen, Volumenkörper, b) komplexen Typen (Sammlung geometrisch-disjunkter Primitive) und c) Aggregationen (Sammlung sich potentiell überlappender Primitive). Außerdem beinhaltet das Spatial Schema ein Modell für Topologytypen (*Topology Packages*) und stellt für beide Modelle Basisfunktionen zur geometrischen Manipulation bzw. Verarbeitung und topologischen Vergleichen konkreter Objektinstanzen bereit.

Während das ISO Spatial Schema mit 3D Geometrien, Kurven u.a. sehr weitreichende Ausdrucksmöglichkeiten aufzubieten hat, erfolgte beim OGC schon früher eine ähnliche, wenn auch deutlich einfachere Ausarbeitung namens

[38] Quelle: OGC-Webseite, http://www.opengeospatial.org/ogc/members
[39] die Auflistung aller OGC Abstract Specifications ist zu finden unter: http://www.opengeospatial.org/standards/as

Simple Feature Access (SFA) [ISO/TC 211 2004*a*]. Die Simple Feature-Spezifikation beschränkt sich auf zweidimensionale Vektorgeometrien, deren Stützpunkte von Linien- oder Polygongeometrien geradlinig miteinander verbunden sind und die sich nicht überlappen dürfen (vgl. Aggregationen des Spatial Schema). Um einen besseren Eindruck zu gewinnen, ist das SFA-Geometriemodell in Abbildung 2.4 dargestellt. Die Spezifikation ist unterteilt in einen konzeptionellen Teil *Part 1 - Common Architecture* und der Modellüberführung zur Unterstützung dreier Basistechnologien, von denen sich in der Praxis nur die Variante der allgemeinen Datenbanksprache *Structured Query Language* (SQL) bewährt hat: *Part 2 - SQL option* [ISO/TC 211 2004*b*]. Die auch als *Simple Features for SQL* bezeichnete Implementierungsspezifikation für Geometrietypen und -funktionen in relationalen Datenbanken ist eine vielfach verwendete, leichte Einstiegshürde für die Bereitstellung standardisierter Geodatenbanken. Es sei nochmal angemerkt, dass die Spezifikation nur die Zugriffsschnittstelle normiert und die Implementierungsdetails den Datenbankherstellern überlässt.

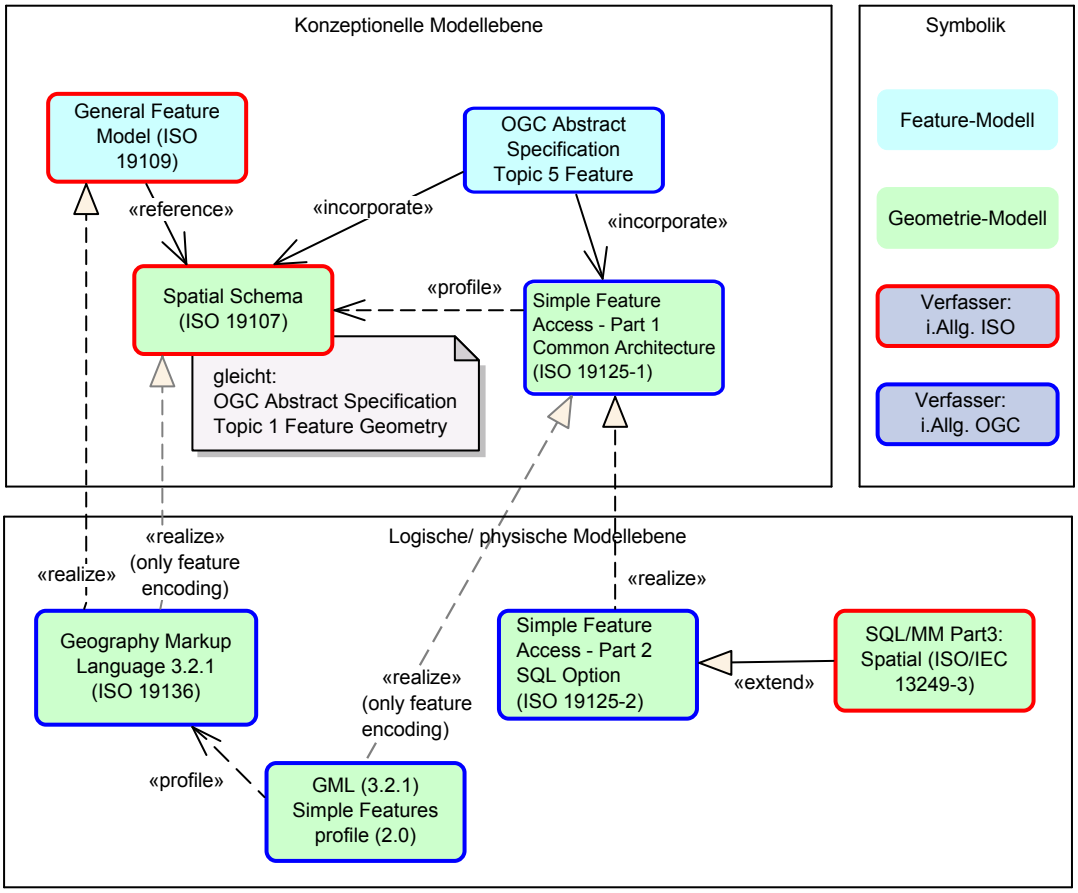


Abbildung 2.3 – ISO/OGC Feature- und Geometriemodelle[40]

Zeitgleich mit Simple Features for SQL wurde SQL92 zu SQL99 mitsamt dem *Multimedia-Package* (SQL/MM) weiterentwickelt. Das Multimedia-Package beinhaltet seitdem die räumliche Erweiterung **SQL/MM Part3: Spatial** mit einem Großteil an übernommenen SFA-Definitionen, wie Stolze [2003] beschreibt. Ebenso sind die Bezeichner an jene des SFA angelehnt, allerdings mit kleinen namentlichen Abwandlungen und können daher leicht verwirren. Aufgrund geringer funktionaler Neudefinitionen ist *SQL/MM Part3: Spatial* eine Obermenge von *Simple Features for SQL*.

[40] Diagrammminhalte entnommen aus den Quellen: ISO [ISO/TC 211 2005*a*], [ISO/TC 211 2003*a*], [ISO/TC 211 2004*a*], [ISO/TC 211 2004*b*], OGC [Open Geospatial Consortium Inc. 2008], [Portele 2007] sowie OGC Abstract Specifications (siehe unter: http://www.opengeospatial.org/standards/as) und Stolze [2003]

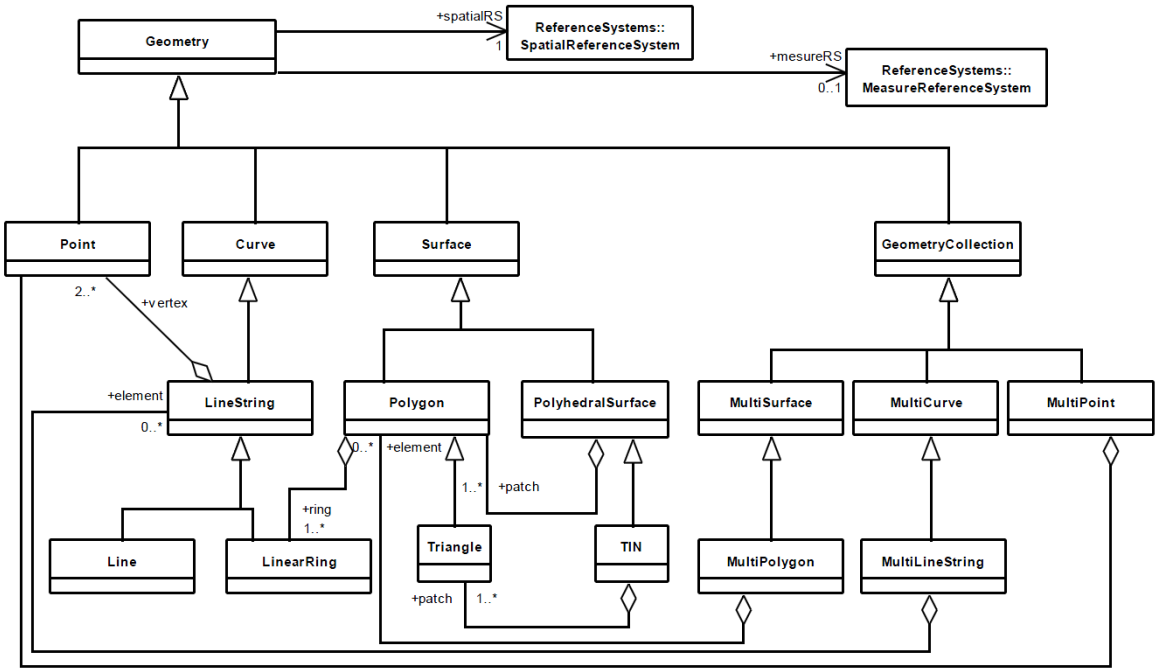


Abbildung 2.4 – Geometrieklassen aus SFA - Part 2 - Common Architecture [ISO/TC 211 2004*a*]

Beide Implementierungsspezifikationen, *Simple Features for SQL* und *SQL/MM Part3: Spatial*, sind mit ihren logischen Modellen auf SQL und damit auf Geodatenbanken ausgerichtet. Die **Geography Markup Language** (GML) [Portele 2007] ist hingegen eine XML-Auszeichnungssprache für Geodaten, deren Definition auf XML-Schemata (XSD) und Kodierungsregeln (Umformung UML → XSD) beruht. GML ist das Standardausgabeformat des *Web Feature Service* (WFS; siehe Abschnitt 2.3.4 *Verteilte Dienstearchitektur*) und kommt überwiegend beim Datenaustausch zwischen Geodiensten und Web-Clients zum Einsatz. Die GML-Version 3.2 ist mittlerweile bei der ISO auch als ISO-Standard 19136 geführt. Die Aufnahme in die ISO-Normenreihe 19100 bewirkte nebenbei, dass GML zu einer (partiellen) Realisierung des umfangreichen ISO 19107 Spatial Schema wurde. Eine neue Profilierung von GML 3.2.1, das sogenannte *GML Simple Features profile*, ist eine *schlanke* Realisierung von ISO 19107, dessen Profillevel *Full* die Konformität zum SFA-Geometriemodell anstrebt (die SFA-Zugriffsfunktionen bleiben unberücksichtigt). Das GML Simple Features profile harmonisiert damit ansatzweise die beiden Welten der XML-Verarbeitung (GML) und der Geodatenbanken (Simple Features). Es sei auch erwähnt, dass die von Google eingeführte und zwischenzeitlich als OGC-Spezifikation akzeptierte *Keyhole Markup Language* (KML) als Visualisierungsnotation im Umfeld von Earth-Browsern (z.B. Google Earth u.a.) ebenfalls einige wenige Geometrietypen anbietet. Diese sind mitsamt Verweisen auf Koordinatenreferenzsysteme der Vorgängerversion GML 2.1.2 entlehnt. Ob und inwieweit KML zukünftig konform zu GML bzw. der ISO-Standardisierungsreihe 19100 weiterentwickelt wird, bleibt abzuwarten.

Geography Markup Language - Kurzporträt

Das XML-kodierte Geography Markup Language dient zum Speichern von abstrahierten Realweltobjekten, den *Features*. Die Sprache ist objektorientiert, da sich auf Features Prinzipien objektorientierter Programmiersprachen anwenden lassen, wie z.B. Abstraktion, Vererbung und Aggregatbildungen. GML gibt drei wesentliche **abstrakte Elementtypen** vor: `gml:AbstractGMLType`, `gml:AbstractFeatureType` und `gml:AbstractGeometryType`. Der Typ `gml:AbstractGMLType` enthält die allgemeingültigen optionalen Attribute zur Benennung und Beschreibung aller GML-Elemente. Die beiden anderen abstrakten Typen erben die Eigenschaften von `gml:AbstractGMLType` und zeigen

die klare definitionsbedingte Trennung zwischen Features und deren Geometrien auf. Die in GML-Anwendungsschemata definierten Objekttypen müssen über herkömmliche XML-Vererbungsmechanismen (`substitutionGroup`) von `gml:AbstractFeatureType` oder `gml:AbstractGeometryType` abgeleitet werden. GML in der Version 2.x kam mit nur drei normativen XML-Schemata aus, mittlerweile besitzt GML 3.x über 30 Schemata. Davon sind viele Schemata optional und liefern z.B. Definitionen für 3D Geometrien, Coverages für kontinuierliche (Raster-) Informationen und topologische Inhalte. Der Anwendungsmodellierer kann sich daraus wie aus einem Werkzeugkasten bedienen und bindet zusätzlich zu den ausgewählten optionalen Schemata die GML-Kernschemata ein.

Zwei wichtige und im Vergleich mit RDF verwandte **Modellierungsmuster** sind die Konzeption als *gerichteter Graph* und das *Objekt-Property-Muster*. Gut herausgestellt werden die beiden Parallelen zwischen GML und RDF in der Arbeit von Schade et al. [2010]. Wie RDF definiert auch GML Attribute für Querbeziehungen zwischen Objekten. Schade et al. zeigen die Entsprechungen zwischen GML- und RDF-Referenzelementen wie folgt: `xlink:href` (GML) entspricht `rdf:resource` (RDF) und `gml:identifier` (GML) gleicht `rdf:about` (RDF). Wie Schade et al. betonen, fällt bei Betrachtung von GML auf, dass Theorie und Praxis leider weit auseinander klaffen. Von GML-Objektreferenzen als Merkmal der Informationsstruktur eines gerichteten Graphen wird nur geringfügig Gebrauch gemacht. Vielmehr dominieren GML-Anwendungsschemata und -Instanzdokumente mit einer einfachen XML-Elementverschachtelung und mit den Zwängen der üblichen XML-Hierarchisierung. Modellierte Querbeziehungen sind eher die Ausnahme.

Das **Objekt-Property-Muster**[41] zeugt vom Werdegang der Sprache GML, das in GML-Version 1.0 eine explizite RDF-Formatierung enthielt. In GML sind Objekte entweder Features oder die das Feature beschreibenden Objekteigenschaften. GML-Objekte (Upper-Camel-Case Schreibweise, z.B. `ProtectedSite`) werden jeweils durch eine GML-Property miteinander verknüpft (Lower-Camel-Case Schreibweise, z.B. `legalFoundationDate`). Der Name einer GML-Property soll möglichst sprechend sein und auf die Semantik des verbundenen Unterelementes hinweisen. Folglich lassen sich die Vergleiche zu RDF darin finden, dass GML-Objekte den Subjekten und Objekten in RDF-Tripels gleichkommen, während GML-Properties den RDF-Prädikaten entsprechen.

Abschließend soll ein kleines Beispiel beide Modellierungsmuster anhand eines GML-Instanzdokumentes demonstrieren. Es enthält eine übergeordnete `gml:FeatureCollection` (GML-Kopfelement, Container von Features) und Details eines INSPIRE Schutzgebiete-Features (hier `ps-f:ProtectedSite`, siehe auch Abschnitt 2.5.2 *Datenspezifikationen*). Das Objekt-Property-Muster ist im Beispiel allgegenwärtig. Eine INSPIRE-Querbeziehung auf ein weiteres INSPIRE-Feature ist mit der GML-Property `ps-f:isInRegion` angedeutet, die eine Referenz des Schutzgebietes auf die sie umgebende biogeographische Region einführt. Die Referenz lässt sich mit Hilfe des `xlink:href`-Attributes über einen Webdienst-Aufruf auflösen.

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
  xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
  xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0"
  xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  gml:id="FC_DEBWPS1" xsi:schemaLocation="urn:x-inspire:specification:gmlas:ProtectedSites:3.0 ProtectedSites.xsd">
  <gml:boundedBy>
    <gml:Envelope srsName="urn:ogc:def:crs:epsg::4326">
      <gml:lowerCorner>47.5 8.1</gml:lowerCorner>
      <gml:upperCorner>48.0 8.25</gml:upperCorner>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <ps-f:ProtectedSite gml:id="DE.BW.PS.335_809028000001">
      <ps:geometry>
```

[41] im Kontext von GML spricht man anstatt von einem *Prädikat* eher von einer *Property*, meint aber denselben Sachverhalt

```
        <gml:Surface gml:id="geomid_335_809028000001_1" srsName="urn:ogc:def:crs:epsg::4326" srsDimension="2">
          <gml:patches>
            <gml:PolygonPatch>
              <gml:exterior>
                <gml:LinearRing><gml:posList>47.542 8.191 47.541 8.193 ... 47.542 8.191</gml:posList></gml:LinearRing>
              </gml:exterior>
            </gml:PolygonPatch>
          </gml:patches>
        </gml:Surface>
      </ps:geometry>
      ...
      <ps:legalFoundationDocument>
        <gmd:CI_Citation>
          <gmd:title>
            <gco:CharacterString>Landesanstalt für Umwelt, Messungen und Naturschutz vom 31.01.08</gco:CharacterString>
          </gmd:title>
          <gmd:date>
            <gmd:CI_Date>
              <gmd:date><gco:Date>2008-01-31</gco:Date></gmd:date>
              <gmd:dateType/>
            </gmd:CI_Date>
          </gmd:date>
        </gmd:CI_Citation>
      </ps:legalFoundationDocument>
      <ps:siteDesignation>
        <ps:DesignationType>
          <ps:designationScheme>UNESCOManAndBiospereProgramme</ps:designationScheme>
          <ps:designation>BiosphereReserve</ps:designation>
          <ps:percentageUnderDesignation>100</ps:percentageUnderDesignation>
        </ps:DesignationType>
      </ps:siteDesignation>
      ...
      <ps-f:isInRegion xlink:href="http://example.wfs.serviceurl?SERVICE=WFS&VERSION=1.1.0
      &REQUEST=GetGmlObject&OUTPUTFORMAT=text%2Fxml%3B+subtype%3Dgml%2F3.2.1&TRAVERSEXLINKDEPTH=0
      &GMLOBJECTID=DE.BW.BGR.100_901028000001"/>
      ...
    </ps-f:ProtectedSite>
  </gml:featureMember>
</gml:FeatureCollection>
```

Codebeispiel 4 – Beispiel eines GML-Dokumentes

2.3.3 Topologische Beziehungen

Der Begriff *Topologie* wird bereits im Abschnitt 2.1 *Begriffsdefinitionen* als „*Lehre von der Lage und Anordnung der Gebilde im Raum*“[42] beschrieben. Worboys & Duckham [2004, S.99] liefern ein einprägsames Beispiel, welche Eigenschaften topologischer und welche nicht-topologischer Natur sind: auf einem Gummituch[43] (*rubber sheet*) sind mehrere Objekte aufgemalt, die sich z.B. überschneiden oder sich gegenseitig enthalten. Dehnt man das Gummituch und damit auch die abgebildeten Objekte, bleiben die topologischen Eigenschaften bzw. Objektbeziehungen der *Überschneidung*, des *Enthaltenseins* etc. gleich. Es ändern sich jedoch alle nicht-topologischen Eigenschaften, wie die Objektdistanzen, -winkel, -längen, -flächen, -umfänge, Zentroidpunkte etc. Worboys & Duckham [2004] nennen eine derartige, durch das Dehnen des Gummituches hervorgerufene Transformation eine **topologische Transformation** bzw. *Homöomorphismus*.

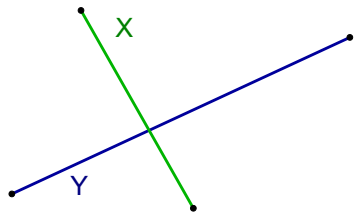
[42] Quelle: Duden online; siehe: http://www.duden.de
[43] stellvertretend für den zweidimensionalen Euklidischen Raum

Topologische Beziehungen können mit Hilfe von Linien- und Polgyon-**Topologietabellen** wie in herkömmlichen Desktop-GIS-Programmen temporär gespeichert werden, um Eigenschaften wie die *Verknüpfung* (Konnektivität) oder die Nachbarschaft von Geoobjekten zur Laufzeit ohne Neuberechnung abfragen zu können. Die Verwendungsmöglichkeiten dieser vorprozessierten Ansätze beschränken sich zumeist auf Darstellungsoptimierungen, die z.B. ein doppeltes Rendering von Grenzlinien benachbarter Flächen vermeiden. Letztlich ist aber die statische Abbildung der topologischen Zusammenhänge aller Geoobjekte innerhalb eines GIS-Projektes oder einer Geodatenbank aufgrund der enorm vielen Kombinationsmöglichkeiten nicht realisierbar. Deshalb müssen konkrete topologische Fragestellungen zur Laufzeit geklärt werden, indem sehr häufig zwei Geoobjekte oder deren Aggregationen miteinander verglichen werden.

Topologische Objekteigenschaften bilden den Gegenstand für ein **weites Forschungsfeld**, das sich nach Worboys & Duckham [2004, S.100] unterteilt in die Betrachtung der „*Topologie von Punktmengen [...] im Speziellen um Konzepte wie die Nachbarschaft, Nähe und offene Mengen*“ und die „*kombinatorische Topologie*“. Letztere schafft die Grundlagen zum Vergleichen diskreter Geoobjekte, die scharfe Umgrenzungen haben und zumeist als Vektorgeometrien vorkommen. Am weitesten erforscht und angewendet im Geo Web- bzw. INSPIRE-Umfeld sind topologische Beziehungen zweidimensionaler Vektorgeometrien, worauf sich auch die weitere Betrachtung beschränkt.

Wie in der SFA-Spezifikation [ISO/TC 211 2004*a*] erläutert wird, lässt sich ein diskretes Geoobjekt einteilen in sein Inneres (Interior, Symbol: °), seine Grenze[44] (Boundary, Symbol: ’) und den, das Geoobjekt umgebenden Raum (Exterior, Symbol: ⁻). Die paarweise Verschneidung der Bereiche `X°`, `X’` und `X⁻` respektive `Y°`, `Y’` und `Y⁻` zweier zu vergleichender Geoobjekte `X` und `Y` führt zu einem **9 Intersection Model** (9IM) [Egenhofer & Herring 1992]. Das 9IM bildet 2^9 verschiedene Möglichkeiten ab, je nachdem, ob die jeweiligen Bereiche miteinander Schnittmengen aufweisen (Ergebniswert = 1) oder nicht (Wert = 0). Eine konkrete topologische Beziehung kann in einer 9IM Matrix dargestellt werden; siehe hierzu Tabelle 2.1, die den Schnitt zweier Linien zeigt. Zusätzlich kann die topologische Dimension[45] der in der Schnittmenge befindlichen Schnittgeometrien berücksichtigt werden. Beispielsweise schneiden sich zwei Flächen zu einer Schnittfläche (2D) oder 2 Linien in einem Schnittpunkt (0D). Hierfür wird das 9IM um die Ergebnisdimension zum **Dimensionally Extended 9 Intersection Model** (DE-9IM) erweitert [Clementini & Di Felice 1995].[46] DE-9IM-Ergebniswerte sind in der Tabelle 2.1 in Klammern wiedergegeben.

Tabelle 2.1 – Topologische Schnittmatrix der dargestellten Linien X und Y



9IM(DE)	`Y°`	`Y’`	`Y⁻`
`X°`	1 (0)	0 (F)	1 (1)
`X’`	0 (F)	0 (F)	1 (0)
`X⁻`	1 (1)	1 (0)	1 (2)

9IM-Matrix Pattern: 1 0 1 0 0 1 1 1 1

DE-9IM-Matrix Pattern: 0 F 1 F F 0 1 0 2

Die 9IM und DE-9IM Werte können zeilenorientiert hintereinander geschrieben werden und ergeben die in Tabelle 2.1 unten dargestellten (DE-)9IM-Muster (Pattern). Diese Matrix-Pattern lassen sich zur Parameterübergabe an eine

[44] von Bedeutung ist die genaue Definition der Grenze: z.B. besitzen Punkte keine Grenze, Linien werden durch ihre Endpunkte begrenzt
[45] die topologische Dimension ist abhängig vom Geometrietyp: Punkt: 0D; Linie: 1D; und Polgyon: 2D
[46] Die Matrix des DE-9IM führt die Schnittmengen-Ergebniswerte `-1,0,1,2,T,F` ein: `-1` bzw. `F`: bedeutet eine leere Schnittmenge; `0`: Schnittpunkt(e); `1`: Schnittlinie(n); `2`: Schnittfläche(n); `T`: beliebiger Wert aus `{0,1,2}`

allgemeingültige topologische Funktion (z.B. die Simple Features-Funktion `relate`) verwenden, um jegliche topologische Beziehungen zweier Geoobjekte abzufragen. Darüber hinaus können aus den zahlreichen, schlecht interpretierbaren Matrix-Pattern solche identifiziert werden, für die **natürlichsprachige Ausdrücke**, z.B. *innerhalb von* (inside) oder *schneidet* (crosses), passend erscheinen. Das Beispiel in Tabelle 2.1 zeigt das DE-9IM Pattern `0 F 1 F F 0 1 0 2` für den Sachverhalt `X schneidet Y` bzw. `Y schneidet X`, das Pattern `2 F F 1 F F 2 1 2` repräsentiert den Sachverhalt `X innerhalb von Y` bzw. `Y enthält X` usw.[47]

Es gibt drei geläufige Familien natürlichsprachiger topologischer Bezeichner:

1. Egenhofer-Relationen
2. Simple Features-Relationen
3. Region Connection Calculus-Relationen (RCC)

Alle drei Relationenfamilien sind in Abbildung 2.5 unterschiedlich eingefärbt wiedergegeben. Das 9IM-Model wurde von Egenhofer & Herring [1992] für GIS-Fragestellungen entwickelt, woraus sich für einfache zweidimensionale Geometrien[48] 8 **Egenhofer-Basisrelationen** einführen lassen (siehe 8 Relationen von Egenhofer in Abbildung 2.5). Weite Verbreitung in der Geo Web-Gemeinde erfahren die topologischen Relationen von Simple Features (eingeführt in *Simple Feature Access*; siehe Abschnitt 2.3.2). Sie leiten sich ab aus den Egenhofer-Basisrelationen und enthalten mit den Relationen `touches`, `crosses` und `overlaps` Einschränkungen auf Geometrietypen der Ausgangsgeometrien (z.B. L/P: Relationsprüfung zwischen Linie und Punkt; A/A: zwischen 2 Flächen etc.). Die dritte Relationenfamilie **Region Connection Calculus** (RCC) ist eine unabhängige Entwicklung von Randell et al. [1992] mit einem formalen theoretischen Ansatz, der sich auf die Beziehungen zwischen Regionen konzentriert. Die acht wichtigsten benannten Relationen sind aus der `connected`-Relation (dt. *verknüpft mit*) abgeleitet (siehe die RCC8-Relationen) [Worboys & Duckham 2004]. Auffallend ist die in Abbildung 2.5 dargestellte Übereinstimmung der Egenhofer-Basisrelationen mit den RCC8-Relationen. Zwei unterschiedliche Theorien führen demnach zu einem gleichen topologischen Relationensatz.

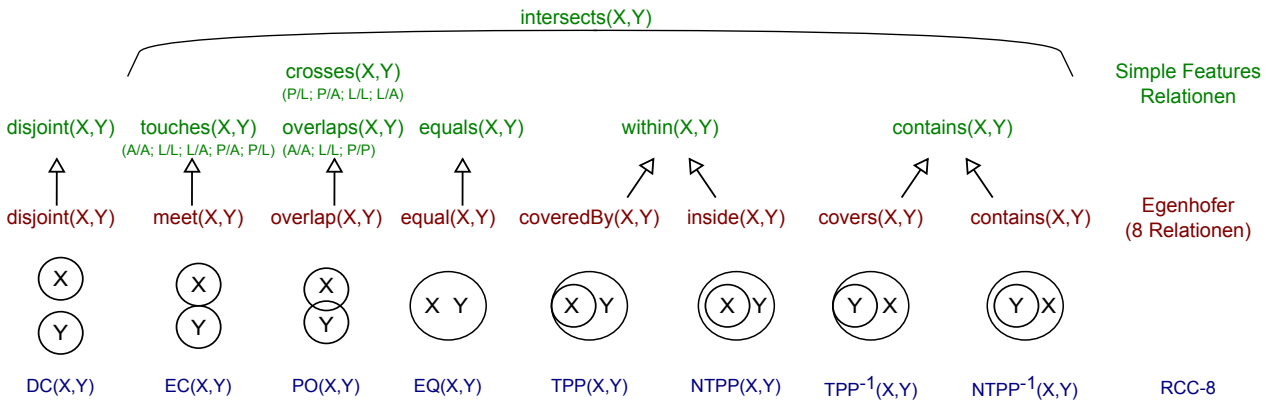


Abbildung 2.5 – Topologische Prädikate nach RCC, Egenhofer und Simple Features [49]

[47] die hier angeführten DE-9IM Pattern sind abhängig von den zu vergleichenden Ausgangsgeometrien. Das erste Beispiele ist nur für zwei sich schneidende Linien gültig, das zweite für ineinander liegende Polygone.

[48] Geometrien im zweidimensionalen Euklidischen Raum, deren Bereiche `X°,X' und X⁻` zusammenhängend sind und die nicht in mehrere Komponenten unterteilt sind (keine Komplexe bzw. Aggregationen)

[49] Quelle: Vergleich von Egenhofer- und RCC8-Relationen aus Knauff et al. [1998], Zuordnung der Simple Features-Relationen gemäß der SFA-Definitionen [ISO/TC 211 2004*a*]

2.3.4 Verteilte Dienstearchitektur

Wie eingangs erläutert, setzt das OGC aus Interoperabilitätsgründen auf die Spezifizierung von Dienstschnittstellen, den sogenannten *OGC Web Services* (OWS). Der Übersichtlichkeit halber sind in Abbildung 2.6 die derzeit häufigsten in Gebrauch befindlichen OWS[50] aufgeführt, die im Weiteren jeweils kurz herausgegriffen werden. Für eine tiefergehende Betrachtung wird auf den gut komprimierten Leitpfaden des OGC, dem *OGC Referenzmodell* [Open Geospatial Consortium Inc. 2008], oder auf das ausführliche Grundlagenbuch *Geodateninfrastrukturen* [Bernard et al. 2005] verwiesen.

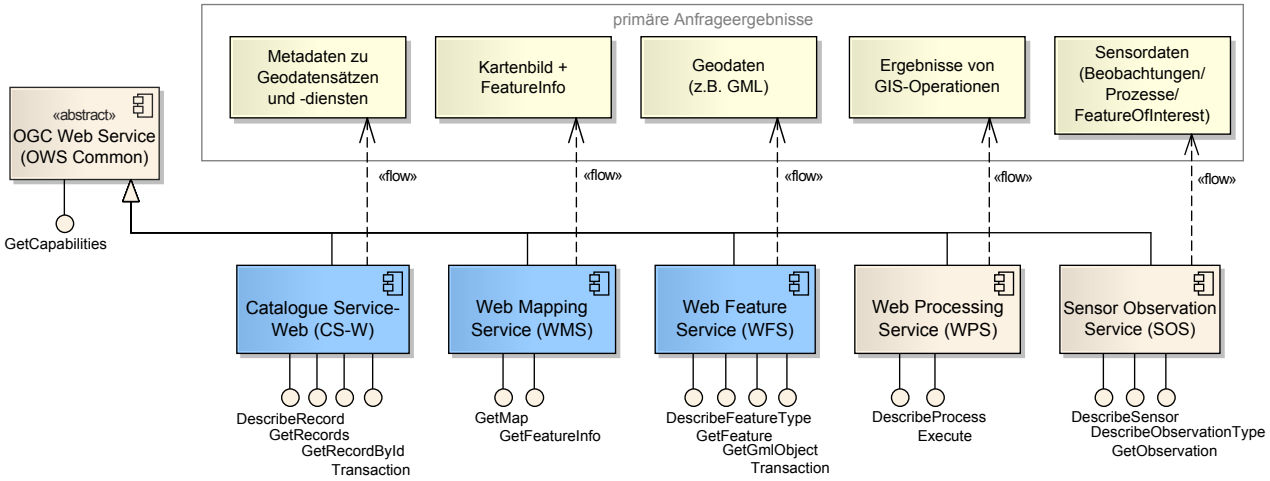


Abbildung 2.6 – Essentielle OGC-Dienstschnittstellen und ihre primären Produkte

Jeder OWS gehorcht den im **OGC Web Services Common Standard** [Whiteside & Greenwood 2010] festgelegten allgemeinen Dienstanforderungen. Danach muss jede OWS-Implementierung verpflichtend die Operation *GetCapabilities* anbieten. Dahinter verbirgt sich - vergleichbar der WSDL (Web Service Description Language) in W3C-Webdiensten - eine Auflistung der Dienstkontaktadressen (Operationen) und -fähigkeiten sowie eine Beschreibung der Datenquellen, auf denen der OWS operiert. Ein Operationsaufruf geschieht entweder über das Protokoll HTTP-Get (Request-Parameter angehängt an die Operationsadresse), per HTTP-POST (Request-Parameter als Key-Value-Pairs oder XML-kodiert im Anhang der HTTP-POST-Nachricht) oder mittels HTTP-POST/SOAP (mit in den SOAP-Envelope eingepacktem XML-strukturierten Request-Block und optionalen Details zum Routing, zur Verschlüsselung und Transaktion).

Kartendienst: Web Mapping Service

Der verbreitetste OGC-Diensttyp ist der *Web Mapping Service* (WMS) [de la Beaujardiere 2006], ein Kartendienst mit vorkonfigurierten Kartenebenen bzw. *Layern*, die in gängigen Bildformaten (z.B. PNG, JPG) ausgegeben werden. Abbildung 2.7 verdeutlicht den Informationsfluss während der Kommunikation zwischen WMS-Cient und -Server. Der WMS-Client informiert sich per Operation *GetCapabilities* über die Fähigkeiten und Inhalte eines WMS-Dienstes. Um die Kartenbilder abzurufen, startet der WMS-Client eine *GetMap*-Anfrage. Erfüllt der WMS-Dienst das optionale Profil *Styled Layer Descriptor* (SLD) kann der Client sogar die Symbolisierungsvorschriften der zu rendernden Kartenbildelemente diktieren. Eine weitere Operation namens *GetFeatureInfo* ermöglicht dem Anwender, Zusatzinformationen zu einzelnen Kartenelementen (Features) zu erfragen, indem die Koordinaten an der Cursor-Position relativ zur

[50] Einschätzung des Autors. Konkrete Statistikzahlen sind nicht bekannt noch zu erwarten, schließlich werden OWS vielfach lokal, in einem Unternehmens- oder Behörden-Intranet eingesetzt. Zu den wichtigsten OWS-Diensttypen zählen sicherlich der Metadatendienst (CS-W), der Kartendienst (WMS) und der Downloaddienst (WFS), da sie die Kernfunktionalitäten einer GDI bieten.

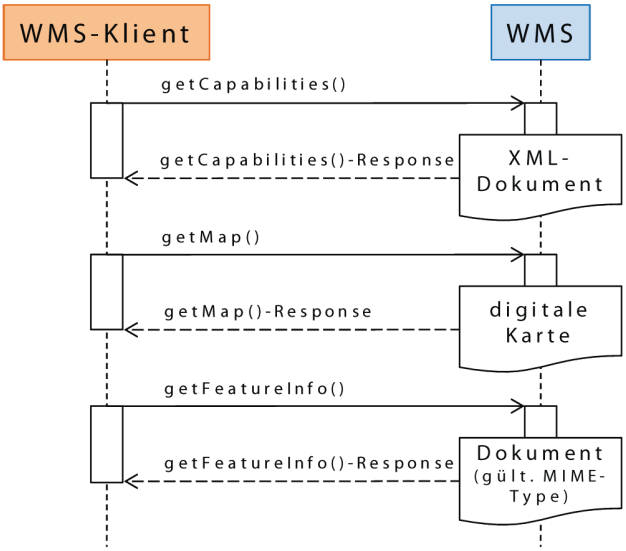


Abbildung 2.7 – OWS-Client/Server-Kommunikation am Beispiel WMS; Quelle: Staub [2009, S.57]

Darstellung im Kartenfenster aufgelöst und in eine GetFeatureInfo-Anfrage umgewandelt werden. Üblicherweise sind dies Sachattribute aus der jeweiligen Feature-Tabelle der Geodatenbank.

Der hier beschriebene Workflow dient also der anfänglichen Orientierung und der nachfolgenden Informationsverdichtung im WMS-Client. Ähnlich sieht die Kommunikationsabwicklung bei anderen OWS aus. Das unten stehende Beispiel veranschaulicht exemplarisch einen WMS-Request, hier im Speziellen ein GetMap-Abruf eines Kartenbildes (per HTTP-GET, ohne optionale Request-Parameter):

```
http://example.wms.serviceurl?service=WMS&version=1.3.0&request=GetMap&layers=Landesgrenze
    &bbox=50.2,9.1,52.3,10&crs=EPSG:4326&width=300&height=450&format=image/png&styles=
```

Der obige Aufruf ist an einen WMS-Server mit spezieller Spezifikationsversion gerichtet (`service=WMS` und `version=1.3.0`) und fordert ein Kartenbild (`request=GetMap`) zur Darstellung von Landesgrenzen an (`layers=Landesgrenze`). Weiter definiert der Request den zu betrachtenden Weltausschnitt über eine rechteckige räumliche Ausdehnung (`bbox=50.2,9.1,52.3,10`), angegeben in geographischen Koordinaten im Koordinatenreferenzsystem WGS84. Zuletzt folgen die gewünschten Kartenbildeigenschaften wie die Angabe des Zielreferenzsystems (`crs=EPSG:4326`), die Bildhöhe und -breite in Pixel (`width=300&height=450`), die Wahl des Bildformates (`format=image/png`) und die von der Render-Engine anzuwendende Symbolisierungsvorschrift (Server eigener Default-Stil mit `styles=`).

Katalogdienst: Catalogue Service - Web

Gemäß dem Leitgedanken *Publish/Find/Bind* werden in einem *Catalogue Service - Web* (CS-W) [Nebert et al. 2007] Metadaten über Geodaten und -dienste registriert, die sich anschließend recherchieren lassen.[51] Ungeachtet möglicher Geo Web-Benutzeroberflächen stellt sich die technische Kommunikation zwischen CS-W-Server und -Client wie folgt dar. Ein Metadaten-Client informiert sich über die Fähigkeiten des CS-W mit der *GetCapabilities*-Operation, die Auskunft über angebotenen Eintragstypen der Metadaten gibt (*Record Types* genannt; z.B. Metadatensätze nach ISO 19115/19- oder der US-amerikanischen *FGDC*-Metadatennorm). Der Client kann sich die Metadaten-Syntax in Form von XML-Schemata mit der Operation *DescribeRecord* abrufen, z.B. um Metadaten zu validieren. Die zentrale

[51] hier ist das Paradigma aus Abschnitt 2.3.1 verkürzt wiedergegeben, die Aktion *Chain* ist üblicherweise nicht im Aufgabenspektrum eines Katalogdientes enthalten

Operation *GetRecords* dient dem Abruf und der gleichzeitigen Filterung von Metadatensätzen. Es können auch gezielt einzelne Metadatensätze anhand ihrer ID aufgefunden werden, hierfür ist die Operation *GetRecordsById* gedacht.

Downloaddienst: Web Feature Service

Der *Web Feature Service* (WFS) [Vretanos 2010*b*] zählt zusammen mit WMS und CS-W zu den Kernkomponenten einer GDI. Der WFS hat die Aufgabe, die unbehandelten Geodatensätze aus diversen Datenquellen über eine Webschnittstelle zum Download anzubieten. Die Datenquellen - als Teil der Datenhaltungsschicht in der gängigen 3-Tier-Architektur[52] - sind entweder dateibasierte Geodatenformate (z.B. *GML*, *ESRI Shape*, *KML*) oder Geodatenbanken. Jede Datenquelle ist eigens als *Feature Type* in der GetCapabilities-Dienstbeschreibung erfasst. Vor dem Export kann sich der WFS-Client auf die Syntax (XML/GML-Schemata) der jeweiligen Feature Types einstellen. Der Download einer Feature-Menge - als *Feature Collection* bezeichnet - erfolgt mit der Operation *GetFeature*. Einzelne Features können mit der Operation *GetGmlObject* (WFS-Version 1.1) oder der leicht umdefinierten Operation *GetPropertyValue* (WFS-Version 2.0) separat heruntergeladen werden. Das WFS-Standardausgabeformat ist GML.

Filterung von Anfragen: Filter Encoding

Wie der Katalogdienst CS-W, so beherrscht auch der WFS eine Filtersprache für den eingeschränkten Download. Der CS-W operiert dabei auf der Filtersprache *OGC Common query language* (CQL)[53]. Die wichtigste CQL-Realisierung ist die XML-kodierte Sprache *OGC Filter Encoding* (FE) [Vretanos 2005*a*], die auch nativ vom WFS verwendet wird. In dieser Konstellation - WFS-Dienst + Filter Encoding - spielen die OGC-Techniken eine zentrale Rolle im Lösungsansatz dieser Arbeit.

Angewendet innerhalb einer WFS GetFeature-Anfrage, filtert ein Filter Encoding-Ausdruck die Informationseinheiten (Features) einer oder mehrerer Objektkategorien (Feature Types). Für diese Aufgabe stehen vier Operatorenklassen zur Auswahl:

- **räumliche Operatoren** (spatial operators): FE verwendet die 8 topologischen Relationen von Simple Features (siehe Abschnitt 2.3.3 *Topologische Beziehungen*) und fügt drei neue hinzu: `ogc:DWithin`, `ogc:Beyond` und `ogc:BBox`. Die ersten beiden Operatoren prüfen, ob sich eine Geometrie innerhalb einer bestimmten Distanz zu einer Vergleichsgeometrie (`ogc:DWithin`) bzw. sich außerhalb jener Distanz befindet (`ogc:Beyond`). Der wohl am häufigsten gebrauchte räumliche Operator `ogc:BBox` erleichtert die Eingabe von Vergleichsgeometrien, die eine kleinste rechteckige Ausdehnung darstellen - *Bounding Box* bzw. auch *Minimal Bounding Rectangle* genannt. Seine Funktionsweise ist äquivalent zur Operatorenkombination `ogc:Not` und `Disjoint`, denn er identifiziert gemäß [Vretanos 2005*a*] „*alle Geometrien, die mit der Bounding Box interagieren*“, d.h. sie schneiden oder von der Bounding Box umschlossen werden. Im Gegensatz zu Simple Features beherrscht FE keine nicht-topologischen Operationen, wie beispielsweise solche für Distanzermittlungen oder Zentroidpunktberechnungen.
- **vergleichende Operatoren** (comparison operators): führen textuelle und numerische Wertevergleiche durch (z.B. `ogc:PropertyIsEqualTo` bzw. `ogc:PropertyIsGreaterThan`). Mit dem Operator `ogc:PropertyIsLike` können Platzhalter für 0-n beliebige Zeichen in den Vergleichstext aufgenommen werden, FE unterstützt hingegen auch in der neuesten Version 2.0 keine regulären Ausdrücke. Dynamische Objektwertvergleiche lassen sich entweder durch neudefinierte Funktionen oder mit Hilfe der vorhandenen arithmetischen FE-Ausdrücke (z.B. `ogc:Add` und `ogc:Mul`) bewerkstelligen.
- **logische Operatoren** (logical operators): inbegriffen sind logische Ausdrücke für Vereinigung (`ogc:Or`, logisches Oder), Durchschnitt (`ogc:And`, logisches Und) und die Komplementärmenge (`ogc:Not`).

[52] 3-Tier: Server-Schichtenarchitektur in 3 Ebenen 1. Client-Schicht, 2. Applikations(server)-Schicht, 3. Datenhaltungsschicht

[53] Verwechslungsgefahr besteht mit der Common- oder Contextual Query Language aus dem Bibliothekarswesen, gleichfalls durch CQL abgekürzt. Hier ist die Rede von der OGC-Anfragesprache, spezifiziert von Nebert et al. [2007]

- **identifizierende Operatoren**: erlauben die Identifizierung und zugleich die Einschränkung von Features (z.B. filtert der Operator `ogc:GetGMLObjectId` über die Feature-Id, die per Attribut `gml:id` angegeben wird).

Über welche der Operatoren eine WFS-Implementierung verfügt, teilt sie in ihrer Dienstbeschreibung (GetCapabilities) in der Sektion *Filter Capabilities* mit. Das nun folgende Beispiel 5 enthält in den Zeilen 12-21 einen FE-Filterausdruck zum räumlichen Eingrenzen mit einer Vergleichsgeometrie (hier ein Polygon). Die Zeilen 22-25 filtern zusätzlich nur die Features, die ein Sachattribut `au:nationalCode` mit dem Wert `100` haben. Anhand der XML-Elementverschachtelung wird die Auswertereihenfolge ersichtlich, zunächst werden die inneren (`ogc:Within` und `ogc:PropertyIsEqualTo`; Zeilen 12 bzw. 22) und danach die äußeren Operatoren aufgerufen (hier `ogc:And`; Zeile 11). Ein Feature-Attribut, dessen Wert als Filterkriterium dienen soll, wird mit Hilfe der FE-Anweisung `ogc:PropertyName` identifiziert (siehe Beispiel 5; Zeilen 13 und 23). Anstelle eines einfachen Attributnamens kann in eine `ogc:PropertyName`-Anweisung auch ein XPath-Ausdruck eingetragen werden, um ein Feature-Attribut zu benennen, das im XML-Elementbaum tiefer geschachtelt ist. Wollte man beispielsweise Schutzgebiete-Features aus dem GML-Dokument in Beispiel 4 nach ihrem Schutzziel einschränken, wäre der folgende XPath-Ausdruck in eine `ogc:PropertyName`-Anweisung einzusetzen: `ps:siteDesignation/ps:DesignationType/ps:designation`.

```
1  <?xml version="1.0"?>
2  <wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" version="1.1.0"
3  xmlns:au="urn:x-inspire:specification:gmlas:AdministrativeUnits:3.0"
4  xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
5  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
7  resultType="results" outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">
8    <wfs:Query xmlns:au="urn:x-inspire:specification:gmlas:AdministrativeUnits:3.0"
9        typeName="au:AdministrativeUnit">
10     <ogc:Filter>
11       <ogc:And>
12         <ogc:Within>
13           <ogc:PropertyName>au:geometry</ogc:PropertyName>
14           <gml:Polygon srsName="urn:ogc:def:crs:epsg::4326">
15             <gml:outerBoundaryIs>
16               <gml:LinearRing>
17                 <gml:posList>50.23,7.53 ...</gml:posList>
18               </gml:LinearRing>
19             </gml:outerBoundaryIs>
20           </gml:Polygon>
21         </ogc:Within>
22         <ogc:PropertyIsEqualTo>
23           <ogc:PropertyName>au:nationalCode</ogc:PropertyName>
24           <ogc:Literal>100</ogc:Literal>
25         </ogc:PropertyIsEqualTo>
26       </ogc:And>
27     </ogc:Filter>
28   </wfs:Query>
29 </wfs:GetFeature>
```

Codebeispiel 5 – Beispiel einer WFS GetFeature-Anfrage inklusive Filter Encoding-Ausdrücke

Rundumblick: Weitere OGC-Diensttypen, Geoportalfunktionen

In den vergangengen Jahren fand eine rasanter **Ausbau der OGC-Dienstepalette** statt. So wurden insbesondere Diensttypen wie der *Web Processing Service* (WPS) oder der *Sensor Observation Service* (SOS) spezifiziert und mit Profilen spezialisiert (siehe Abbildung 2.6). Ein WPS informiert mit der Operation *DescribeProcess* über die möglichen GIS-Analyseprozesse (mitsamt Ein- und Ausgabeparameter), die mittels *Execute* angestoßen werden können. Der SOS gehört zur OGC-Initiative *Sensor Web Enablement* (oder kurz: *Sensor Web*), die das Umfeld der Sensordatenverarbeitung mit Diensten und Transfersprachen unterstützt. Der SOS selbst ist das Frontend eines Sensornetzwerkes. Die Spezifika der Sensoren können mit der Operation *DescribeSensor* erfragt und beobachtete Messwerte mit der Operation *GetObservation* im Standardformat *Observation & Measurement* (O&M) exportiert werden. O&M ist ein GML-Profil und wird momentan in verschiedenen Domänen - organisiert in den sogenannten OGC-Domain Working Groups (DWG) - auf individuelle Nutzerszenarien adaptiert. So entsteht beispielsweise in der Hydrologie die *Water Markup Language 2.0* (WaterML 2.0).

Ein anderer sehr agiler Technologiestrang trägt den Namen *OpenLS* und kümmert sich um *Open Location-based Services* (LBS). Dahinter verbergen sich Geodienste, die im Umfeld der Mobilfunk-Architekturen beheimatet sind und interaktive Kartendarstellungen, Lokalisierungs- und Routingaufgaben für leistungsschwache mobile Endgeräte zu leisten imstande sind. Zusammen mit Formaten wie *KML*, *GeoRSS* (siehe Abschnitt 2.4) oder *GeoSMS*, eine leichtgewichtige Erweiterung des bekannten Short Message Service-Formats (SMS) mit erweiterten Lokalisierungsinhalten, ist auch Open-LS Teil der **OGC Mass Market Services**, die die Mainstream-IT-Welt mit ubiquitären geographischen Funktionalitäten ausstattet. Umrahmt wird die OGC-Dienstepalette durch einige eher allgemeingültige Diensttypen, wie z.B. Sicherungs- und Bepreisungsdienste, die jedoch zumeist nicht den Status einer OGC-Implementierungsspezifikation genießen.

Abschließend sei auch kurz die Verbraucherseite der OGC-Geodienste angesprochen. Häufig werden verschiedene Geo Web-Benutzeroberflächen in einem gemeinsamen Webauftritt eingebunden, sogenannte **Geoportale**, und agieren als Thin-Clients, d.h. sie lagern die grösste Prozesslast auf Server-seitige Prozeduren aus. Die Verwendung von graphischen Benutzeroberflächen umfasst die Kartenbetrachtung, interaktive Datenmanipulation, Metadatenerfassung und -recherche. Geoportale sind für gewöhnlich intuitiv, d.h. ohne technische Vorkenntnisse, bedienbar und erlauben - eventuell abgestuft mit Zugriffsrechten - verschiedenen Zielgruppen, vom GIS-Administrator bis zu anonymen Internetanwendern, einen Einblick in fremde bzw. externe Datenhaltungen. Geoportale runden damit die größtenteils auf den Serverbereich beschränkten OGC-Aktivitäten ab. Sie werden insbesondere auch im Zuge der europäischen Richtlinie INSPIRE (siehe Abschnitt 2.5) bei vielen Institutionen der öffentlichen Hand aufgebaut und per OWS miteinander vernetzt.

2.4 Schnittmenge Geo Semantic

Die Einführung zur GML in Abschnitt 2.3.2 deckt bereits mehrere Gemeinsamkeiten zwischen RDF und GML auf. Ebenso steht außer Frage, dass generelle IT-Techniken wie z.B. die XML-Kodierung bzw. das XML-Parsing von Transferformaten oder die Etablierung von Netzwerkdiensten von beiden Disziplinen - dem Geo Web und dem Semantic Web - gleichermaßen genutzt werden. Dieser Abschnitt soll einen Einblick in die voranschreitende Annäherung beider Disziplinen auf dem Feld der räumlichen Informationsverarbeitung geben, die für das Geo Web immanent wichtig ist und für das Semantic Web aufgrund moderner Geo Tagging-Anwendungsszenarien und digitalen Globen stets an Bedeutung gewinnt. Die Wörter *Geo Semantic* oder auch *Geospatial Semantic Web* (GSW) bezeichnen diese inhaltliche Schnittmenge. Sie wird insbesondere von aktuellen Aktivitäten seitens der OGC-Anwendergemeinde geprägt, die darauf abzielen, einerseits komplexere geodätische Ansätze im Semantic Web zu etablieren und andererseits OGC-Geodienste mit Techniken des Semantic Web anzureichern. Zunächst einmal soll aber der ursprüngliche Umgang mit räumlicher Information im Semantic Web betrachtet werden. Hierzu stellen die beiden folgenden Unterabschnitte 2.4.1 Geo-Vokabulare und 2.4.2 geosemantische Anfrageschnittstellen vor.

2.4.1 Beispiele semantischer Geo-Vokabulare

Im Semantic Web existieren verschiedene Ansätze, um räumliche Informationen in RDF und OWL zu speichern.[54] Es wird nun im Folgenden eine Auswahl an vielfach zitierten und verwendeten Vokabularen vorgenommen. Ein Manko besonders älterer Geo-Vokabulare besteht darin, dass Koordinaten häufig ohne ordnungsgemäße Angabe des Koordinatenreferenzsystems (CRS) gespeichert werden. So gibt das Vokabular *Earth Maps, Latitudes, and Longitudes*[55] die Kodierung in geographischen Koordinaten vor, es fehlt jedoch der nötige Bezug zu einem geodätischen Datum. Das sicherlich am weitesten verbreitete - wenn auch weiterhin informelle - Geo-Vokabular ist das W3C **Basic Geo** (WGS84 lat/long) [Brickley 2003], das per Definition festgelegt ist auf die Verwendung des CRS WGS84, welches im Umgang mit GPS-Koordinaten bekannt ist. Basic Geo führt eine Klasse namens `geo:SpatialThing` für geographische Objekte (Features) ein, die Koordinaten geographischer Länge und Breite über die Prädikate `wgs84_pos:lat` und `wgs84_pos:long` referenziert, z.B. als Tripel ausgedrückt: `ex:Feature1 wgs84_pos:lat "50.0"` und `ex:Feature1 wgs84_pos:long "8.27"`. Punkthöhen bezogen auf den WGS84-Referenzellipsoiden lassen sich mit `wgs84_pos:altitude` ausdrücken. Alles in allem ist das Basic Geo-Vokabular prädestiniert für Aufgaben des Geo Tagging, mit dem bevorzugt WGS84-Punktgeometrien erfasst werden und nicht etwa komplexere Geometrietypen bzw. Koordinaten in alternativen Referenzsystemen.

Das Projekt **Geonames**[56] veröffentlicht geographisches Namensgut, wie z.B. Städte- und Regionennamen. Das zugehörige Geonames-Vokabular verwendet Basic Geo-Konzepte zur Punktspeicherung und erweitert diese um einige topologische Prädikate, wie z.B. `gn:nearby` für nahe gelegene Features, `gn:neighbour` für benachbarte Features, `gn:parentFeature` für das administrativ übergeordnete Feature usw. Das Codebeispiel 6 ist im RDF/XML-Format angegeben und informiert über ein Feature, das den Bezirk *Brest* beschreibt (siehe Prädikat `gn:name` in Zeile 2). Die Zeilen 3 und 4 enthalten die Lagekoordinaten, angegeben per Basic Geo-Prädikate. Die eben erwähnten topologischen Prädikate weisen in den Zeilen 5 und 6 den Bezirk Brest als Teil des *Départements Finistère* aus und geben an, dass Brest in der Nähe der Stadt *Milizac* liegt. Geonames enthält auch Prädikate, die topologische Beziehungen eines Typus (z.B. `gn:nearby`) zu einer RDF-Sammlung zusammenfassen und per URI referenzieren (z.B. `gn:nearbyFeatures`; Zeile 7).

[54] eine Übersicht ist im W3C-Wiki zu finden unter: http://www.w3.org/wiki/GeoInfo
[55] siehe RDF-Schema unter: http://www.w3.org/2000/10/swap/pim/earthMap.n3
[56] Dokumentation: http://www.geonames.org/ontology/documentation.html

```
<gn:Feature rdf:about="http://sws.geonames.org/3030299/">
  <gn:name>Arrondissement de Brest</gn:name>
  <wgs84_pos:lat>48.5</wgs84_pos:lat>
  <wgs84_pos:long>-4.5</wgs84_pos:long>
  <gn:parentFeature rdf:resource="http://sws.geonames.org/3018471/"/>
  <gn:nearby rdf:resource="http://sws.geonames.org/6430974/"/>
  <gn:nearbyFeatures rdf:resource="http://sws.geonames.org/3030299/nearby.rdf"/>
  <gn:locationMap rdf:resource="http://www.geonames.org/3030299/arrondissement-de-brest.html"/>
  ...
```

Codebeispiel 6 – Beispiel eines Geonames-Features

Ein Versuch, das Vokabular Basic Geo für weitere Geometrietypen zu öffnen, wird mit **GeoRSS** [Reed et al. 2006] unternommen. GeoRSS ist beim OGC als *White Paper* geführt, was einem offiziellen Standpunkt ohne Spezifikationscharakter gleichkommt. Das Newsfeed-Format *RSS* basiert in der Version 1.0 auf RDF und bildet damit die Formatgrundlage für GeoRSS. GeoRSS ist in zwei Profile unterteilt, in ein *Simple-* und ein *Extended-* bzw. *GML 3.1.1-Profil*. Das Simple-Profil speichert mehrere zusammenhängende Koordinatentupel über jeweils ein Prädikat, wie z.B. `ex:Feature1 georss:line "45.256 -110.45 46.46 -109.48"`. Ebenso schränkt das Simple-Profil auf WGS84-Koordinaten und die Geometrietypen Punkt (Prädikat `georss:point`), Linie (`georss:line`), Polygon (`georss:polygon`) und Bounding Box ein (kleinste rechteckige Ausdehnung, `georss:box`). Das Extended- oder GML 3.1.1-Profil definiert nur ein Prädikat namens `georss:where`, das zum Speichern vollständiger GML 3.1.1-Geometriekodierungen dient. Das Beispiel 7 verdeutlicht diesen Sachverhalt: das `georss:where`-Prädikat in Zeile 3 inkludiert eine GML-Polygongeometrie mit allen untergeordneten XML-Elementen. Dadurch können letztlich beliebige GML-Geometrietypen eingesetzt und Koordinaten in alternativen CRS angegeben werden.

```
<entry>
  ...
  <georss:where>
    <gml:Polygon>
      <gml:exterior>
        <gml:LinearRing>
          <gml:posList>45.13 -109.49 45.38 -109.23 44.85
              -109.33 45.13 -109.49</gml:posList>
        </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </georss:where>
</entry>
```

Codebeispiel 7 – Beispiel eines Atom-Feeds mit GeoRSS-Zusatz

Gemäß dem Simple-Profil werden die Koordinatenwerte einer Geometrie zu einer Zeichenkette aneinander gereiht und sind dadurch nicht mehr isoliert als einzelne RDF-Literale interpretierbar, so dass einfachste numerische Koordinatenvergleiche (z.B. $\mathtt{X} > 50°$ nördlicher Breite) nicht mehr funktionieren. Jedoch können komplexere Geometrien sehr schnell Tausende von Koordinatentupeln als Stützpunktinformationen enthalten. Eine vereinzelte Koordinatenspeicherung mit den Prädikaten `wgs84_lat` und `wgs84_lon` erscheint unter diesen Umständen unzweckmäßig, da man Gefahr läuft, die geometrische Ordnung der komplexen Geometrien zu verlieren. Dies lässt den Schluss zu, dass Geometrien geschickterweise, so wie im GeoRSS Simple-Profil vorgeschlagen, in einer Zeichenkette (*Literal*) transferiert werden und nur ausgewählte Geometrie-Interpreter jene zusammenhängenden Zeichenketten einlesen und untersuchen sollten. Das Extended-Profil ist dem Simple-Profil in dieser Hinsicht sehr ähnlich, denn trotz der zusätzlichen mittels `georss:where` inkludierten XML-Elemente übernimmt das Extended-Profil die GML-Syntax für komprimierte Speicherungen von Koordinatentupeln, z.B. mit `gml:posList` (siehe Beispiel 7; Zeilen 7 und 8).

Ein prominentes Anwendungsbeispiel der oben genannten Geo-Vokabulare ist das Projekt **LinkedGeoData** (LGD) [Auer et al. 2009], das mit OpenStreetMap-Daten (OSM) eine der größten Geodatenbestände in die Linked Data-Cloud integriert. Für die Geometriespeicherung nutzt LinkedGeoData Basic Geo- und GeoRSS-Prädikate. Das Codebeispiel 8 zeigt ein LGD-Feature (gekennzeichnet mit `lgd:Way`; Zeile 1), das einen Parkplatz identifiziert (Zeile 2). Die Polygongeometrie des Parkplatzes ist einerseits nach der GeoRSS-Methode vollständig in einem Literal angegeben (Zeile 3) und andererseits in einer Knotensequenz festgehalten (Zeile 4; referenziert auf Zeile 8). Die Sequenz gibt die Knoten in den Zeilen 9 bis 12 an, wobei ein einzelner Knoten, referenziert aus Zeile 9, mit geographischer Breiten- und Längenangabe in den Zeilen 15 bis 18 dargestellt ist. LinkedGeoData schränkt die Wahl der Geometrietypen in Punkte bzw. Knoten (Node) und Linien und Polygone ein, die als Routen (Way) gespeichert werden. Knoten und Routen lassen sich beliebig annotieren und typisieren. Interessanterweise stützt sich die Datenhaltung von LinkedGeoData nicht nur auf einen RDF-Triplestore für Tagging-Informationen, sondern lagert unabhängig dazu Geometrien in eine relationale Datenbank mit QuadTree-Indizierung aus (in OSM *QuadTiles* genannt). Exportiert werden die Geodaten über den Datenbankaufsatz *Triplify*[57], eine leichtgewichtige Webschnittstelle zur Umwandlung von relationalen in RDF-Repräsentationen.

```
1  <lgd:Way rdf:about="http://linkedgeodata.org/triplify/way100000049">
2    <rdf:type rdf:resouce="http://linkedgeodata.org/ontology/Parking"/>
3    <georss:polygon>54.9449538 82.8725115 54.9448994 82.8733572 ... 54.9449538 82.8725115</georss:polygon>
4    <lgd:hasNodes rdf:resource="http://linkedgeodata.org/triplify/way100000049/nodes"/>
5    ...
6  </lgd:Way>
7
8  <rdf:Seq rdf:about="http://linkedgeodata.org/triplify/way100000049/nodes">
9    <rdf:li rdf:resource="http://linkedgeodata.org/triplify/node1156219772"/>
10   <rdf:li rdf:resource="http://linkedgeodata.org/triplify/node1156219641"/>
11   ...
12   <rdf:li rdf:resource="http://linkedgeodata.org/triplify/node1156219772"/>
13 </rdf:Seq>
14
15 <lgd:Node rdf:about="http://linkedgeodata.org/triplify/node1156219772">
16         <wgs84_pos:lat>54.9449538</wgs84_pos:lat>
17         <wgs84_pos:long>82.8725115</wgs84_pos:long>
18 </lgd:Node>
```

Codebeispiel 8 – LinkedGeoData-Beispiel für ein flächenhaftes Feature

2.4.2 Geosemantische Anfrageschnittstellen

Der Abschnitt 2.4.1 hat Vokabulare und Initiativen genannt, die Geoinformationen mit Semantic Web-Mitteln modellieren. Damit die Datenspeicherung nicht zu einem Selbstzweck verkommt, sollten auch Zugriffswege und Analysemöglichkeiten offen stehen. Eine adequate Anfrageschnittstelle erlaubt neben textuellen und temporalen auch räumliche Filterungen.

Die Semantic Web-Gemeinde beschäftigt sich seit geraumer Zeit mit räumlichen Anfrageschnittstellen, ohne dabei die OGC-Entwicklungen einzubeziehen. Beispielsweise verwenden Jain et al. [2009] **partonomische Beziehungen** (Teil-Ganzes-Relationen), die allgemein dem Wissensgebiet der Topologie zugehören. Die partonomischen Beziehungen werden statisch zwischen Instanzen der Wissensbasis eingefügt. Desweiteren machen Jain et al. Gebrauch von SPARQL und Methoden des *Query Rewriting*, worunter das Umformen von Benutzeranfragen zu verstehen ist. Mit Query Rewriting-Regeln werden insbesondere einfache SPARQL-Benutzeranfragen um komplexere Graphenmuster erweitert. Beispielsweise decken Query Rewriting-Regeln transitive Prädikate innerhalb einer SPARQL-Anfrage auf, wie z.B. eine

[57] Projektseite: http://triplify.org/

Verkettung der Beziehung `liegt innerhalb von`, und ergänzen sie zu Graphenmustern passend zu Speicherformen einer bestimmten Wissensbasis. Der nächste Abschnitt 2.4.3 *State of the art: GeoSPARQL* wird das Query Rewriting anhand eines Beispiels veranschaulichen.

Kolas [2008] verwendet hingegen die topologischen Beziehungen der Region Connection Calculus-Theorie (siehe Abschnitt 2.3.3). Kolas diskutiert die Analyse topologischer Objektbeziehungen mit der Anfragesprache SPARQL und stellt diesbezüglich die beiden zentralen SPARQL-Auswerteoptionen gegenüber: a) per topologischer Prädikate in den Graphenmustern oder b) über den Gebrauch von SPARQL-Filterfunktionen. Option a) setzt die statische Speicherung topologischer Prädikate in der Wissensbasis voraus, während Option b) die Möglichkeit bietet, topologische Beziehungen auch zur Laufzeit festzustellen. Kolas spricht sich für Lösungsweg a) aus, um SPARQL-Erweiterungen zu vermeiden, die mit der Neudefinition von räumlichen SPARQL-Filterfunktionen einhergehen würden. Nichtsdestotrotz ist Kolas auf andere SPARQL-Erweiterungen angewiesen, z.B. definiert Kolas einen SPARQL `Premise`-Ausdruck, um temporäre Vergleichsgeometrien, kodiert in GeoRSS, den SPARQL-Anfragen beizufügen. GeoSPARQL löst die Problemstellung eleganter. Und zwar unterstützt GeoSPARQL beide Auswerteoptionen a) und b) und schlägt zusätzlich Query Rewriting-Regeln vor, um Graphenmuster mit topologischen Prädikaten (a) nach Bedarf zu räumlichen Filterfunktionen (b) zu transformieren (siehe nächster Abschnitt 2.4.3).

Das OGC ist spätestens seit 2006 darum bemüht, die starren Grenzen zwischen dem Geo Web und dem Semantic Web zu überwinden. So wurde das Format GeoRSS zum Untersuchungsgegenstand im OGC-Interoperabilitätsprogramm *OWS-4* [Vretanos 2007], das sich primär mit der Tauglichkeit von OGC-Diensten als Massenmarktprodukte (*Mass Market*) beschäftigte. Dabei war GeoRSS, integriert in einem Atom Newsfeed, als leichtgewichtiges WFS-Rückgabeformat vorgesehen. Zur gleichen Zeit fand das OGC Interoperabilitätsexperiment *Geospatial Semantic Web* (GSW.IE) [Lieberman 2006] statt, um die semantische Verfügbarmachung bzw. Nutzung von OWS-Diensten zu testen, sogenanntes **Semantic Enablement**. Das GSW.IE liefert eine Referenzarchitektur und gibt semantische Profile bzw. verschiedene Stufen des Semantic Enablements für die OGC-Diensttypen CS-W und WFS vor.

Für den WFS werden vier semantische Level formuliert:

1. GetCapabilities-Ausgabe im Format OWL-S[58]
2. DescribeFeatureType-Ausgabe in OWL
3. GetFeature- und GetFeatureById-Ausgaben in OWL
4. Unterstützung einer RDF/OWL-Anfragesprache[59]

Die GSW.IE-Arbeitsgruppe schlägt vor, die vier semantischen Level entweder als integraler Bestandteil des WFS oder per WFS-Fassade bzw. -Proxy zu realisieren. Diese Ansätze waren mitunter eine wichtige Anregung für die vorliegende Arbeit, die dem zweiten Lösungsweg, die Realisierung eines WFS-Proxy, nachgeht.

Die im GSW.IE getroffenen Überlegungen werden von Janowicz et al. [2010] konsequent weitergedacht und semantische OWS-Profile (hier **Semantic Enablement Layer**: SEL) eingeführt. Die OWS-Profile sind a) ein CS-W-Profil *Web Ontology Service* (WOS) zum Management und Zugriff auf Ontologiekonzepte und b) ein WPS-Profil *Web Reasoning Service* (WRS) für die Steuerung lokaler oder entfernter Inferenzmaschinen und die Interaktion mit dem WOS. Für den Betrieb des WOS ist ein CS-W mit semantischen Annotierungen angedacht. Janowicz et al. [2010] beziehen sich dabei auf die Dissertation von Klien [2008]. Klien schlägt die Übersetzung von GML-Applikationsschemata in OWL-Ontologien vor, damit Letztere semantisch mit Konzepten aus Upper- bzw. Core-Ontologien (z.B. Dolce) annotiert werden können. Janowicz et al. und Klien beschäftigen sich im Wesentlichen damit, semantische Konzepte und Technologien zur Unterstützung der geographischen Datenrecherche in GDIs nutzbar zu machen. Der Export

[58] OWL-S ist ein OWL-Vokabular für die Beschreibung von Webdiensten inklusive Profile, Bindings, Ein- und Ausgabeparameter

[59] zusätzlich zur Standard-Filtersprache OGC Filter Encoding. Das GSW.IE verweist für zukünftige Entwicklungsarbeiten auf die Anfragesprache SPARQL. Zur Zeit des GSW.IE schien die SPARQL-Spezifizierung bereits vielversprechend, lag allerdings erst in einer Entwurfsfassung vor.

von Semantic Web-Formaten oder die Etablierung von geosemantischen Anfrageschnittstellen als GDI-Fassaden ist von untergeordneter Bedeutung und eine mögliche Linked Data-Schnittstelle (hier *μSDI* genannt) wird nur am Rande diskutiert [Janowicz et al. 2010]. Als einzelne Geo Web-Domäne tut sich das **OGC-Sensor Web** damit hervor, diverse durch SOS-Dienste zugängliche Sensordaten auch über Linked Data-Endpoints zu veröffentlichen [Page et al. 2009] [Patni et al. 2010]. Leider existieren keine SPARQL-Endpoints zur Datenrecherche, so dass anstelle von Filteranfragen nur ein *RDF-Browsen* der Sensordatenbestände möglich ist.

Zur Veröffentlichung von INSPIRE-Daten (siehe Abschnitt 2.5) empfiehlt sich ebenfalls ein SPARQL-Endpoint, und zwar in Form einer Fassade vor INSPIRE-Datenzugriffsdiensten; sogenannten *INSPIRE-Downloaddiensten*. INSPIRE lässt zwar mehrere Arten von Downloaddiensten zu, rät aber prinzipiell zum Einsatz eines OGC WFS. Tatsächlich gibt es bereits mehrere Ansätze, OGC WFS-Dienste mit semantischen Anfrageschnittstellen auszustatten, z.B. in Gomes Jr & Medeiros [2007] oder Zhao et al. [2008]. Jedoch beruhen diese Ansätze auf kaum strukturierten GML-Inhaltsmodellen mit *flacher* XML-Elementhierarchie. Heutzutage sind angesichts komplexer INSPIRE-GML-Applikationsschemata (z.B. *O&M* oder das *AAA*-Model der deutschen Vermessungsverwaltungen[60]) eher Inhaltsmodelle mit vielen verschachtelten XML-Elementen üblich. Dadurch ergeben sich neue Anforderungen für einen SPARQL-Endpoint in Verbindung mit einem OGC WFS-Dienst, wie etwa die umfangreiche Analyse von SPARQL-Anfragen, um Anfragebestandteile an den WFS weiterzuleiten, oder ein komplexes Mapping von GML auf RDF/OWL im Zuge der Ergebnistransformation. Diese und weitere Aspekte werden im Rahmen dieser Arbeit diskutiert und Lösungen erarbeitet.

2.4.3 State of the art: GeoSPARQL

Im vorangegangenen Abschnitt 2.4.2 wurden mehrere Semantic Web-Ansätze zu räumlichen Anfrageschnittstellen präsentiert. Mit dem aktuellen OGC-Entwurf namens GeoSPARQL [Perry & Herring 2011] wird ein fortschrittlicher Standardisierungsversuch für räumliche Anfrageschnittstellen unternommen. Das W3C als Urheber der Sprache SPARQL hat an GeoSPARQL bereits Interesse bekundet. GeoSPARQL wurde von der *Spatial Ontology Community of Practice* (SOCOP[61]) aus der Taufe gehoben und seit Mai 2010 von der *OGC GeoSPARQL Standard Working Group* weitergeführt. Die offizielle Kommentierungsphase fand im Juli/August 2011 statt, die Fertigstellung steht kurz bevor (Stand: Januar 2012). Es bleibt zu hoffen, dass sich GeoSPARQL dank der ausgereiften Entwurfsfassung als konsensfähig erweisen wird.

Die Spezifikation sieht fünf verschiedene Konformitätsklassen vor, die aufeinander aufbauen:

1. Core
2. Geometry Extension (*serialization, version*)
3. Topology Vocabulary Extension (*relation_family*)
4. Geometry Topology Extension (*serialization, version, relation_family*)
5. Query Rewrite Extension (*serialization, version, relation_family*)

Die Aufteilung in fünf **Konformitätsklassen** ist auf den interoperablen Charakter der GeoSPARQL-Spezifikation zurückzuführen, die sich auf diverse standardisierte Geometriekodierungsformate unterschiedlicher Versionen stützt (kursiv dargestellte Parameter *serialization*, *version*). Beispielsweise können kommende GeoSPARQL-Schnittstellen konform zum Geoformat GML der Version 3.2.x implementiert werden, um GML-Geometrien in Version 3.2.x einzulesen und zu analysieren. Ebenso unterscheidet GeoSPARQL mögliche Implementierungen darin, welche Familien

[60] **A**FIS-**A**LKIS-**A**TKIS-Modell; siehe Dokumentation unter: http://www.adv-online.de/icc/extdeu/broker.jsp?uMen=0a170f15-8e71-3c01-e1f3-351ec0023010

[61] Webseite: http://www.socop.org/

topologischer Bezeichner unterstützt werden (Parameter *relation_family*). Hier schränkt GeoSPARQL auf die drei in Kapitel 2.3.3 eingeführten Relationenfamilien ein: Egenhofer (8 Relationen), Simple Features und RCC8. Abgesehen von der durch die Parameterklassen eher künstlich bedingten Aufteilung, setzt sich GeoSPARQL aus drei wesentlichen Inhaltsbestandteilen zusammen:

- ein Geo-Vokabular mit topologischen Prädikaten
- ein Satz an topologischen und geometrischen SPARQL-Filterfunktionen
- ein Satz an Transformationsregeln zum Query Rewriting räumlicher Anfragen

Das **Geo-Vokabular** dient in erster Linie dazu, Geometriespeicherungen in Semantic Web-Formaten zu harmonisieren, damit Filterfunktionen und Transformationsregeln einheitlich auf dieser Datengrundlage operieren können. Würde im Umkehrschluss kein eigenständiges Vokabular definiert werden, so müssten die besagten Auswerteroutinen äußerst flexibel geschrieben sein, um alle in Frage kommenden Geometriespeicherungen zu berücksichtigen (siehe Abschnitt 2.4.1: *Beispiele semantischer Geo-Vokabulare*). Anhand der Abbildung 2.8 soll nun zunächst das Geo-Vokabular vorgestellt werden. Die Abbildung zeigt alle RDF/OWL-Klassen bzw. -Prädikate in einem Graphen, dessen Notation der eines UML-Klassendiagramms entspricht (siehe auch die Notationserläuterung unter *Symbolik*).

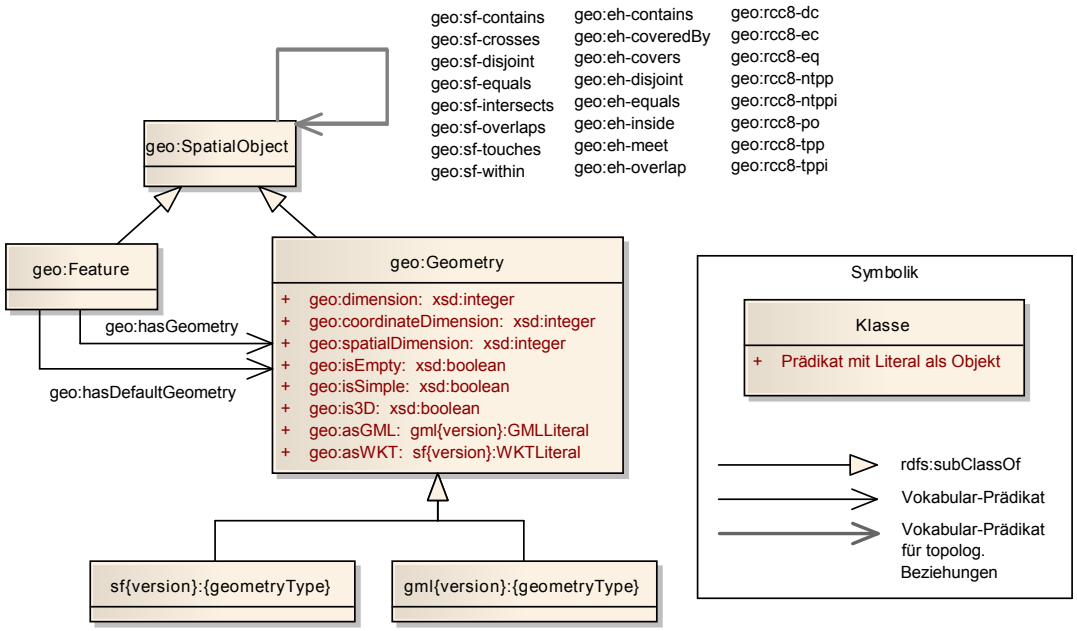


Abbildung 2.8 – GeoSPARQL - Vokabularkonzepte

Die Superklasse `geo:SpatialObject` wird **konsistent zum General Feature Model** (ISO 19109) strukturiert in die Konzepte `geo:Feature` und `geo:Geometry`. Ein `geo:Feature` ist die Repräsentation eine Realweltobjektes mit einem oder mehreren räumlichen Attributen, die über die Relation `geo:Feature geo:hasGeometry geo:Geometry` gespeichert werden. Das Prädikat `geo:hasGeometry` wird mit dem Prädikat `geo:hasDefaultGeometry` spezialisiert, um GeoSPARQL-Funktionen die primär zu prozessierende (Default-)Geometrie anzuzeigen, sollte ein Feature mit mehr als einer Geometrie gespeichert werden.[62] Domänenontologien, die von GeoSPARQL Gebrauch machen, sind nicht an die GeoSPARQL-Namensgebungen `geo:Feature` und `geo:Geometry` gebunden, sondern können lokale Subtypen

[62] aufgrund der Open World Assumption (OWA) könnten durchaus Fälle mit Mehrfachzuweisungen von `hasDefaultGeometry` auftreten. Die Spezifikation weist jedoch speziell darauf hin, dass solche OWA-Szenarien nicht ausgeschlossen werden können, daraus aber keine Prozessierfehler der GeoSPARQL-Implementierung resultieren sollten

bilden, wie z.B. `ex:Katasterparzelle rdfs:subClassOf geo:Feature`. GeoSPARQL sieht ausdrücklich vor, dass eine konforme Anfrageschnittstelle RDFS- oder OWL-Reasoning (-Entailments) beherrschen muss, um zur Laufzeit auf Subtypen von `geo:Feature` zu schließen und jene in räumlichen Operationen zu verarbeiten.

Die Klasse `geo:Geometry` enthält sechs verschiedene **geometrische Attribute** zur Geometriebeschreibung, die samt Definitionen aus *Simple Feature Access* entnommen wurden. Die geometrischen Attribute (hier als UML-Klassenattribute dargestellt) sind als Prädikate modelliert, die als Objekte einfache Literaltypen, wie z.B. `xsd:integer`, zulassen. Beispielsweise beschreibt das Prädikat `geo:dimension` die topologische Dimension (Punkt: '0', Linie: '1' etc.), `coordinateDimension` die Koordinatendimension ('2' für zweidimensionale Geometrien etc.) und `spatialDimension` die Koordinatendimension zuzüglich weiterer Dimensionen (z.B. M-Values im Kontext *Linearer Referenzierung*).

Die beiden übrigen Prädikate `geo:asGML` und `geo:asWKT` dienen der eigentlichen **Geometriekodierung**, indem sie RDF-Literale der RDF-Datentypen `GMLLiteral` bzw `WKTLiteral` als Objekte referenzieren. In diesen RDF-Literalen liegen die Geometriedaten serialisiert als Zeichenkette vor und sind konform zu standardisierten Geometriekodierungen gespeichert (z.B. GML, GeoJSON, KML). Die im GeoSPARQL Geo-Vokabular enthaltenen Prädikatnamen `geo:asGML` und `geo:asWKT` weisen darauf hin, dass das Geo-Vokabular explizite Verweise auf die zwei Geodatenformate GML und *Well-known text* (WKT) enthält. Letzteres Format ist ein Ascii-Textformat, das in *Simple Features for SQL* spezifiziert wird. WKT und sein binäres Pendant namens *Well-known binary* (WKB) lassen sich recht kompakt kodieren, was mitunter ein Grund dafür ist, warum GeoSPARQL das Geodatenformat WKT adressiert. Für die Unterstützung weiterer Geodatenformate werden GeoSPARQL-Vokabularerweiterungen hinsichtlich neuer Prädikate (z.B. `geo:asGeoJSON`) oder entsprechender RDF-Datentypen (z.B. `GeoJSONLiteral`) erforderlich. GeoSPARQL-Implementierungen sind nicht an eine bestimmte Geometriekodierung gebunden, sondern können eine oder mehrere Kodierungen unterstützen und ihre Funktionalität mit dem Hinweis auf die realisierten Konformitätsklassen (z.B. Geometry Extension(*GeoJSON*, *2.0*)) beispielsweise in den Dienstemetadaten proklamieren.

Frühere Entwurfsfassungen sahen ein eigenständiges **Geometrietypenmodell** für GeoSPARQL vor, das dem Typenmodell von *Simple Feature Access* entlehnt war (siehe Abbildung 2.4). Mittlerweile ist man von diesem Ansatz abgekommen und übernimmt kurzerhand die Geometrietypen der Geodatenstandards. Die Geometrietypen werden in GeoSPARQL als Subtypen von `geo:Geometry` eingeführt, indem jedem Geometrietyp ein URI-Bezeichner im GeoSPARQL-Namensraum gemäß dem URI-Template `http://www.opengis.net/def/dataType/{format}/{format-version}/{geometry-type}` zugeteilt wird. Demzufolge wird der Simple Features-Geometrietyp `Polygon` mit der URI `http://www.opengis.net/def/dataType/OGC-SF/1.0/Polygon` benannt, der GML-Geometrietyp `Curve` mit der URI `http://www.opengis.net/def/dataType/OGC-GML/3.2/Curve`. Letztlich lässt sich den hier besprochenen Geometrietypen als ontologische Konzepte nur geringer Informationsmehrwert beimessen, da zukünftige GeoSPARQL-Prozessoren auf den geometrischen Serialisierungen `GMLLiteral` und `WKTLiteral` operieren und den jeweiligen Geometrietyp direkt aus der Serialisierung herauslesen können.

Der praktische Umgang mit dem Geo-Vokabular wird nun im Beispiel 9 veranschaulicht. Das Beispiel zeigt ein Feature `ex:Katasterparzelle` (Zeile 7), das eine Grenzgeometrie `ex:Grenze` besitzt (Zeile 8). Die Zeilen 9 und 10 attestieren der Grenzgeometrie zweidimensionale Koordinaten und eine *simple* Geometrie gemäß Simple Features, d.h. die Grenzlinie schneidet sich nicht selbst und berührt sich auch nicht tangential. Die Grenzgeometrie ist kodiert entweder als WKT (Zeilen 11-13) oder als GML (Zeilen 14-18). Das Beispiel verdeutlicht, dass GeoSPARQL die Ausdrucksmöglichkeiten der jeweiligen Geodatenformate nachnutzt. Dabei fordert die GeoSPARQL-Spezifikation als zusätzliche Information das verwendete Koordinatenreferenzsystem (CRS) in Form einer URI, die entweder direkt in der Geometriekodierung enthalten sein kann (siehe GML-Beispiel) oder wie im Falle von WKT extern gespeichert und durch Leerstellen von der Geometrie-Zeichenkette getrennt wird. Bleibt die empfohlene CRS-Angabe aus, so geht die Anfrageschnittstelle per Definition vom System *CRS:84* aus.[63] GeoSPARQL kapselt ähnlich wie GeoRSS die

[63] der OGC-Code *CRS:84* ist zu differenzieren von *WGS:84*. Beide Koordinatenreferenzsysteme beruhen auf dem WGS84-Datum mit geographischer Koordinatenangabe (Dezimalgrad), allerdings erfolgt die Koordinatenreihenfolge bei CRS:84 in Lon/Lat (d.h. die geograph.

Geometriekodierungen in RDF-Literalen. Auf diese Weise ist ein GeoSPARQL-Prozessor in der Lage, eine Geometrie als Ganzes zu interpretieren und nicht deren einzelne Konstrukte verteilt auf mehrere Tripelinformationen wie z.B. einzelne Stützpunkte oder äußere und innere Grenzlinien. Auch ist es für nachgelagerte Anwendungen, z.B. GIS-Routinen oder Visualisierungen in Karten-Clients, leichter mit zusammenhängenden Geometriespeicherungen umzugehen.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/> .
@prefix sf10: <http://www.opengis.net/def/dataType/OGC-SF/1.0/> .
@prefix gml32: <http://www.opengis.net/def/dataType/OGC-GML/3.2/> .
@prefix ex: <http://example.org/> .

ex:Katasterparzelle rdf:type geo:Feature .
ex:Katasterparzelle geo:hasDefaultGeometry ex:Grenze .
ex:Grenze geo:isSimple "true"^^xsd:boolean .
ex:Grenze geo:coordinateDimension "2"^^xsd:integer .
ex:Grenze geo:asWKT
    "<http://www.opengis.net/def/crs/OGC/1.3/CRS84>
      Polygon((10.0 54.3, 10.2 54.3, 10.2 54.4, 10.0 54.5, 10.0 54.3))"^^sf10:WKTLiteral .
ex:Grenze geo:asGML
    "<gml:Polygon gml:id=\"geom_id1\" srsName=\"http://www.opengis.net/def/crs/OGC/1.3/CRS84\"
      xmlns:gml=\"http://www.opengis.net/gml\"><gml:exterior><gml:LinearRing>
        <gml:posList dimension=\"2\">10.0 54.3 10.2 54.3 10.2 54.4 10.0 54.5 10.0 54.3</gml:posList>
      </gml:LinearRing></gml:exterior></gml:Polygon>"^^gml32:GMLLiteral .
```

Codebeispiel 9 – Verwendungsbeispiel des Geo-Vokabulars

Um binäre topologische Beziehungen zwischen `geo:SpatialObject`-Instanzen auszudrücken, kommen in GeoSPARQL **topologische Prädikate** (`geo:sf-contains`, `geo:eh-equals`, `geo:rcc8-po` etc.) zur Anwendung, deren Präfixe `sf`, `eh` und `rcc8` auf die drei Relationenfamilien Simple Features, Egenhofer und RCC8 hindeuten. Eine Übersicht über alle topologischen Prädikate zeigt Abbildung 2.8. Die topologischen Prädikate werden als Tripel, z.B. `ex:Katasterparzelle geo:sf-within ex:Stadtgebiet`, statisch in den Instanzdaten abgelegt und vergrößern damit die Wissensbasis (vgl. mit Prädikaten aus der Datenquelle *Geonames*). Als einfache semantische Prädikate erfordern sie keinen GeoSPARQL-Prozessor. Ein regulärer SPARQL Endpoint genügt, um die topologischen Beziehungen zu detektieren. Soll hingegen die Wissensbasis unberührt bleiben und die zahlreichen topologischen Sachverhalte erst während einer Anfrage ermittelt werden, so kommt die **Query-Funktionalität** von GeoSPARQL zum Einsatz. Alle topologischen Beziehungen lassen sich mit SPARQL-Filterfunktionen, die durch GeoSPARQL neudefiniert werden (sogenannte SPARQL *extended filter functions*[64]), zur Laufzeit auswerten. Sie tragen die gleichen Namen wie die topologischen Prädikate, unterscheiden sich aber im Namensraum (z.B. anstelle von `geo:sf-contains` schreibt man `geof:sf-contains`[65]). Eine GeoSPARQL-Filterfunktion namens `geof:relate` hebt sich von den drei topologischen Relationenfamilien ab, da sie mit einem DE-9IM-Matrix Pattern parametrisiert aufgerufen wird (siehe Abschnitt 2.3.3 *Topologische Beziehungen*). Desweiteren sind unäre und binäre Funktionen für nicht-topologische geometrische Kalkulationen verfügbar, die z.B. die kürzeste Distanz zwischen zwei Geoobjekten (`geof:distance`) oder die konvexe Hülle einer Geometrie berechnen (`geof:convexHull`).

Die GeoSPARQL Query-Funktionalität kann optional durch einen **Query-Rewrite**-Vorgang ähnlich der besprochenen Vorgehensweise von Jain et al. [2009] ergänzt werden. Dabei werden Transformationsregeln aufgestellt, die hier auf SPARQL-Graphenmuster angewendet werden. Das nachstehende Beispiel 10 zeigt eine verkürzte Transformationsregel

Länge wird der geograph. Breite vorangestellt), bei WGS:84 in Lat/Lon. Beispiel: Standort Vancouver in CRS:84 x/y = -123.12/49.28 und in WGS:84 x/y = 49.28/-123.12

[64] siehe SPARQL Spezifikation [Prud'hommeaux & Seaborne 2008], Section 11.6

[65] Namensräume: `geo=http://www.opengis.net/ont/OGC-GeoSOARQL/1.0/`, `geof=http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/`

aus der GeoSPARQL Query Rewrite Extension in der standardisierten *Rule Interchange Format*-Syntax (RIF[66]). Die Regel ist definiert für zwei Features `?f1` und `?f2` (Objekte vom Typ `geo:Feature`), ihren Geometrien `?g1` und `?g2` (Objekte vom Typ `geo:Geometry`) und deren geometrischen Serialisierungen `?g1Serial` und `?g2Serial` (referenziert über das Prädikat `geo:asWKT`). Bei Anwendung der Regel wird ein SPARQL-Pattern der Form `?f1 geo:sf-intersects ?f2` (Beispiel 11; Zeile 5) in Pfadvergleichsmuster (Beispiel 12; Zeilen 10-13) und den Aufruf einer Filterfunktion `geof:sf-intersects(?g1Serial, ?g2Serial)` (Zeile 14) umgewandelt:

```
1  Forall ?f1 ?f2 ?g1 ?g2 ?g1Serial ?g2Serial
2    (?f1[geo:sf-intersects->?f2] :-
3       And
4         \# feature - feature rule
5         (?f1[geo:hasGeometry->?g1]
6          ?f2[geo:hasGeometry->?g2]
7          ?g1[geo:asWKT >?g1Serial]
8          ?g2[geo:asWKT->?g2Serial]
9          External(geof:sf-intersects(?g1Serial,?g2Serial)))
10   )
```

Codebeispiel 10 – Beispiel einer GeoSPARQL-Transformationsregel

Der Vorher/Nachher-Vergleich einer GeoSPARQL-Anfrage ergibt:

```
1  PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/> .
2  PREFIX ex: <http://example.org/> .
3
4  SELECT ?feature
5  WHERE { ?feature geo:sf-intersects ex:Katasterparzelle_A10 }
```

Codebeispiel 11 – Ursprüngliche GeoSPARQL-Anfrage

```
1  PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/> .
2  PREFIX geof: <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/> .
3  PREFIX ex: <http://example.org/> .
4
5  SELECT ?feature
6  WHERE {
7          { ?feature geo:sf-intersects ex:Katasterparzelle_A10 }
8          UNION
9          {
10           ?feature geo:hasGeometry ?g1 .
11           ex:Katasterparzelle_A10 geo:hasGeometry ?g2 .
12           ?g1 geo:asWKT ?g1Serial .
13           ?g2 geo:asWKT ?g2Serial .
14           FILTER (geof:sf-intersects(?g1Serial, ?g2Serial))
15         }
16       }
```

Codebeispiel 12 – Transformierte GeoSPARQL-Anfrage

Der Anwender ist in der Lage, eine erheblich kürzere Anfrage zu verfassen, die sich allein auf topologische Prädikate stützt. Dank des Query-Rewritings kommt er trotzdem in den Genuss einer Auswertung über Filterfunktionen. Damit endet an dieser Stelle die GeoSPARQL-Einführung, das Kapitel 5 Abschnitt 5.1.4 greift das Thema GeoSPARQL erneut auf, angefangen mit einer kritischen Betrachtung des Geo-Vokabulars.

[66]Spezifikation: http://www.w3.org/TR/rif-overview/

2.5 EU-Direktive INSPIRE

2.5.1 Projektfundamente

Die im März 2007 in Kraft getretene EU-Richtlinie *Infrastructure for Spatial Information in Europe* (INSPIRE) [Europäisches Parlament und Europäischer Rat 2007] bezweckt, Bestrebungen zum Aufbau nationaler Geodateninfrastrukturen (NGDI) koordinierend zu fördern und deren Datenprodukte und Dienste in eine übergeordnete **europäische Geodateninfrastruktur** (European Spatial Data Infrastructure: ESDI) einzugliedern. INSPIRE hat speziell vor dem Hintergrund des 6. EU-Umweltaktionsprogrammes die fachliche Aufgabe, eine integrative Umweltpolitik mitzugestalten und öffentliche Daten mit Raumbezug mittels standardisierter Web-Techniken und mit legislativen Mitteln nutzbar zu machen. Ähnliche Ziele verfolgen auch zahlreiche andere Regelungen zur IT-Infrastruktur und ihre projektbezogenen Anwendungen auf europäischer Ebene. Hier ist z.B. die 1990 erlassene EU-Umweltrichtlinie (90/313/EWG) zur Regelung des freien Zugangs zu Umweltinformationen zu nennen oder die EU-Wasserrahmenrichtlinie aus dem Jahre 2000 (2000/60/EG), die auf eine einheitliche Wasserpolitik abzielt und der als erste Richtlinie mit IT-gestützter EU-Berichterstattung die Rolle eines Testpiloten für europäische IT-Vorhaben zukommt.

Nachdem eine Expertengruppe bereits 2001 die Vorbereitungsarbeiten aufnahm, konnte die INSPIRE-Richtlinie schließlich im Jahre 2007 verabschiedet werden. Danach folgte die Überführung in nationales Recht. In Deutschland liegt seit Februar 2009 die Entsprechung mit dem *Geodatenzugangsgesetz* (GeoZG) vor. Ebenso zieht INSPIRE eine föderalistisch-geprägte Umsetzung in Landesrecht mit Wirkung auch auf die Kommunen nach sich (z.B. Bayern Juli 2008 und Nordrhein-Westfalen Februar 2009 [67]). Die seitdem begonnene **INSPIRE-Implementierungsphase** soll sich in Etappen zur vollen Operationsfähigkeit bis zum Jahre 2019 erstrecken, während die Vielfalt an Geobasis- und Geofachthemen, die sogenannten 34 *INSPIRE Annex-Themen*[68], sukzessive erweitert wird und deren Publizierung allmählich interoperabler zu geschehen hat.

Die Interoperabilität wird durch den Einsatz ISO- und OGC-standardisierter Geodienste angestrebt, die mit minimalen Modifikationen zu sogenannten *INSPIRE-Netzwerkdiensten* erweitert werden. Geodaten, ob umstrukturierte oder neu erhobene Daten, verbleiben effizienterweise beim Datenerfasser in einem für ihn zweckmäßigen Format. Lediglich zur Datenweitergabe per Netzwerkdienste erfolgt die **Harmonisierung auf dem Service-Level**, die in den meisten Fällen eine semantische Datentransformation in eine INSPIRE-konforme Speicherform erfordert. Ein wesentliches Ziel ist die grenzübergreifende Sicht auf die europäische Datenwelt. Der Datenzugriff soll mit wenigen Ausnahmen, beispielsweise bei sicherheitskritischen Daten oder jenen mit geistigen Eigentumsrechten, uneingeschränkt möglich sein. Allerdings lässt die Richtline ausdrücklich auch Bezahlstrategien zu, insbesondere bei solchen Daten, die aus kostspieligen Datenerhebungen resultieren. Aufgrund vielseitiger Interessen und unterschiedlich gelagertem Fachwissen wurde ein gemeinschaftlicher Spezifizierungsprozess angestoßen, an dem sich registrierte öffentliche Stellen als *Legally Mandated Organisation* (LMO), Wirtschaftsverbände oder sonstige Nutzergemeinschaften als *Spatial Data Interest Communities* (SDIC) und nationale Kontaktknoten und Betreiber der NGDIs als *Member State Contact Points* (MSCP) aktiv beteiligen. Das INSPIRE-Komitee, zusammengesetzt aus Vertretern der EU-Mitgliedsstaaten, assistiert bzw. berät die Europäische Kommission und stimmt über bindende Regularien ab, den *Implementing Rules* (IR). Letztere sind verpflichtende Textwerke, die aus den technischen Spezifikationen, den *Technical Guidance*-Dokumenten, hervorgehen. Das INSPIRE *Consolidation Team* (CT) koordiniert die Entwicklungsarbeit, die von *Drafting Teams* (DT) in jeweils einem von fünf Themenblöcken geleistet wird[69].

[67] Bayerisches Geodateninfrastrukturgesetz (BayGDIG); siehe http://by.juris.de/by/gesamt/GDIG_BY.htm und Geodatenzugangsgesetz - GeoZG NRW; siehe: https://recht.nrw.de/lmi/owa/br_bes_text?gld_nr=7&ugl_nr=7134&aufgehoben=N&anw_nr=2&bes_id=12584

[68] Auflistung aller INSPIRE Annex-Themen unter: http://inspire.jrc.ec.europa.eu/index.cfm/pageid/2/list/7; tragen ihren Namen aufgrund der Nennung im Anhang der INSPIRE-Richtlinie

[69] Quelle: INSPIRE-Portal; siehe: http://inspire.jrc.ec.europa.eu

Die **fünf Themenblöcke** der INSPIRE-Entwicklungsarbeiten sind:

1. **Metadaten**: definiert ein eigenes ISO-konformes Metadatenprofil (ISO 19115/19119-Anwenderprofil, *Community Profile*) zur Beschreibung von INSPIRE-Datensätzen und -Netzwerkdiensten
2. **Datenspezifikationen**: spezifiziert für fast alle 34 Annex-Themen konzeptionelle UML-Modelle und GML-Applikationsschemata. Ausnahmen bilden die ersten beiden Grundlagenthemen *Koordinatenreferenzsysteme* und *Gittersysteme*. Die Datenspezifikationen sind als Datenprodukte nach ISO 19131 angelegt. Zur Ausarbeitung der aufwändigen Datenspezifikationen wurde eine weitere organisatorische Untergliederung vorgenommen und für jede Fachdomäne eine *Thematic Working Group* (TWG) gebildet.
3. **Netzwerkdienste**: spezifiziert Schnittstellen von Geodiensten. Herkömmliche OGC-Geodienste werden zur Differenzierung *Spatial Data Services* genannt. INSPIRE-Konformität erlangen Spatial Data Services durch eine erweiterte Funktionalität und die Einhaltung von Servicequalitäten, wie z.B. Verfügbarkeits- und Performanzanforderungen. Dann nennt man sie INSPIRE-Netzwerkdienste.
4. **Daten- und Dienstebenutzung**: regelt Zugriffszeiträume auf Daten, Preisauskünfte bei kostenpflichtigen Datenabgaben und andere organisatorische und sicherheitstechnische Vereinbarungen
5. **Monitoring und Reporting**: registriert Aktivitäten der Mitgliedsstaaten und fordert ein aktives Reporting bezüglich der INSPIRE-Datenmeldungen und der allgemeinen nationalen Richtlinienumsetzung

2.5.2 Datenspezifikationen

Die Modellierung der Datenspezifikationen für die INSPIRE Annex-Themen beruht auf der Vorgabe von vier *Rahmendokumenten*:

- D2.3 Definitions of Annex Themes and Scope
- D2.5 Generic Conceptual Model
- D2.6 Methodology for the development of data specifications
- D2.7 Guidelines for the Encoding of Spatial Data

Das Dokument D2.3 **Definitions of Annex Themes and Scope** [INSPIRE Drafting Team „Data Specifications“ 2008*a*] setzt die Themendefinitionen aus der Richtlinie vertiefend fort, führt exemplarische Anwendungsfelder auf und nennt entsprechend einer Handlungsanweisung die wichtigsten Konzepte je Thema, die es auszumodellieren gilt. Wie genau die Entwicklung der Datenspezifikationen organisatorisch abzulaufen hat, wird durch D2.6 **Methodology for the development of data specifications** [INSPIRE Drafting Team „Data Specifications“ 2008*b*] umrissen. Die hier festgelegte Methodik umfasst zusammen sieben Arbeitsschritte, von der Anwendungsfallbeschreibung und Anforderungsanalyse über die Modellierung bis zur testweisen Implementierung und der finalen Kosten/ Nutzen-Abwägung.

Die eher technische Ebene wird durch die beiden anderen Dokumente abgedeckt. Das wichtigste Rahmendokument D2.5 **Generic Conceptual Model** [INSPIRE Drafting Team „Data Specifications“ 2010*a*] sieht verschiedenste Infrastrukturmaßnahmen der INSPIRE-Datenwelt vor und bringt diese in Einklang. Wesentlich ist der Einsatz der Modellierungssprache UML (Version 2.1) und der GML (Version 3.2.1) zur Erstellung der Datenmodelle. Dabei orientiert sich das Drafting Team *Data Specifications* zusätzlich am ISO 19101 Referenzmodell [ISO/TC 211 2002*a*]. Insbesondere dient das *General Feature Model* (GFM, ISO 19109) [ISO/TC 211 2005*a*] als Meta-Modell zum Entwickeln der konzeptionellen Schemata für *Spatial object types* und ihrer Eigenschaften. Auch viele weitere OGC- und ISO-Standardisierungen werden erwähnt und deren Übernahme und Abwandlungen im INSPIRE-Kontext diskutiert. So auch die zu etablierenden ISO-Registrierdienste nach ISO 19135 [ISO/TC 211 2005*c*], die als Nachschlagewerke

von INSPIRE-Begrifflichkeiten, -Definitionen und -Konzepten dienen. INSPIRE wird 6 Register umfassen. Für die Terminologie werden 1. ein Glossar und 2. ein Feature Concept Dictionary zur Harmonisierung themenübergreifender Konzepte eingerichtet. Zur Unterstützung der konzeptionellen Schemata und der Applikationsschemata werden 3. konsolidierte UML-Modelle, 4. ein Feature Catalogue Register und 5. ein Codelist-Register gepflegt. Sehr sinnvoll ist auch 6. ein CRS/UoM-Register für abgestimmte europäische Koordinatenreferenzsysteme (CRS) und häufig verwendete Maßeinheiten (*Unit of Measures*, UoM).

Während das Generic Conceptual Model nach den Worten des Drafting Teams die „*erste Interoperabilitätsstufe*“ darstellen soll, dienen als weitere Eckpfeiler der Harmonisierung a) die Datenspezifikationen inklusive der konzeptionellen UML-Modelle und b) die D2.7 **Guidelines for the encoding of spatial data** [INSPIRE Drafting Team „Data Specifications“ 2010*b*]. Letztere legen die Standard-Kodierungsregeln zur Abbildung von INSPIRE-konsolidierten UML-Modellen über zwei Umformungsprozesse in GML-Applikationsschemata fest. Auf INSPIRE-Elemente, die den ISO-Metadatenstandards 19115/19119 entstammen, werden die ISO 19139 Encoding Rules [ISO/TC 211 2007*b*] angewendet (z.B. für Kontaktadressen oder Angaben zur Datenqualität). Für die meisten anderen INSPIRE-Elemente (Geometrietypen, Datumsangaben etc.), die konform zu anderen Standards der ISO 19100 Normenreihe sind, greifen die Kodierungsregeln der GML-Spezifikation [Portele 2007]. Die relevanten Regeln für INSPIRE sind dem GML-Dokumentenanhang E zu entnehmen. Es können aber je nach Anforderungslage eventuell auch weitere Kodierungsregeln in Frage kommen, so z.B. für Rasterinformationen der Annex-II-Themen.

Die **INSPIRE-Agenda** sieht die operationelle Einführung von Annex-I-Themen für Ende 2012 vor. Danach sind sukzessive weitere Datenthemen aus Annex-II/III über Zugriffsdienste zugänglich zu machen. Diese Regelung betrifft zunächst nur neu erfasste oder stark restrukturierte Datensätze, die bis Ende 2014 bereitzustellen sind, in weiterer Folge dann auch alle übrigen INSPIRE-relevanten Datensätze bis Oktober 2019.[70]

Beispiel einer Datenspezifikation

Anhand einer beispielhaften Datenspezifikation (DS) werden nun grundlegende Charakteristiken der INSPIRE-Modellierung erläutert. Die nachfolgende Betrachtung beschränkt sich dabei auf die INSPIRE UML-Modelle, da bei der Umformung zu GML-Applikationsschemata einige INSPIRE-relevante Modellierungsdetails verloren gehen, die an die jeweiligen Elementdeklarationen der UML-Modelle (Packages, Klassen, Attribute oder Assoziationen) angeheftet sind. Dies sind entweder UML-Stereotypen zur Elementkategorisierung oder UML-Eigenschaftswerte (sogenannte *Tagged values*) zur weiterführenden Elementbeschreibung, die im INSPIRE-Kontext beispielsweise zum Speichern von GML-Umformungsregeln oder Versionsinformationen genutzt werden. Die als wichtiger einzustufenden Stereotypen bestimmen die Art oder die den UML-Elementen zugedachte Rolle.

Als beispielhafte DS soll hier das **INSPIRE Annex-Thema Protected Sites** (dt. Schutzgebiete) dienen [INSPIRE Thematic Working Group „Protected sites“ 2010*b*]. Die DS Protected Sites modelliert Schutzgebiete, die definiert sind als „*Gebiete, die im Rahmen des internationalen und des gemeinschaftlichen Rechts sowie des Rechts der Mitgliedsstaaten ausgewiesen sind oder verwaltet werden, um spezifische Erhaltungsziele zu erreichen*“ [Europäisches Parlament und Europäischer Rat 2007]. Darunter zählen z.B. Habitatgebiete gemäß der europäischen Fauna-Flora-Habitat-Richtlinie (FFH-Richtlinie; 92/43/EWG; 1992), die international ausgewiesenen UNESCO-Welterbestätten (1975) oder auch kleinere Schutzgebiete, wie z.B. Baudenkmäler oder archäologische Ausgrabungsstätten.

Jede DS kann ein oder mehrere Applikationsschemata (AS) enthalten, die gleichbedeutend sind mit UML-Packages des Stereotyps `applicationSchema` und abgebildet werden auf je ein GML-Applikationsschema. Für die DS Protected Sites wurden zwei unterschiedlich umfangreiche AS namens *Simple* und *Full* aufgestellt. Das AS Simple ist obligatorisch anzuwenden, während die erweiternden Elemente von AS Full optional bleiben. Abbildung 2.9 zeigt das UML-Modell bzw. -Package zu AS Full.

[70] siehe INSPIRE-Roadmap: http://inspire.jrc.ec.europa.eu/index.cfm/pageid/44

Die Farbgebung der Notationen im UML-Klassendiagramm (Abbildung 2.9) deutet auf die Herkunft der dargestellten INSPIRE-Objekttypen hin. Die beige hinterlegten Elemente gehören zum AS Simple, blaue stammen aus dem AS Full und die grünen Elemente sind Platzhalter bzw. vorläufige Spezifikationen der Themen aus Annex-III, wie z.B. *Bio-geographical Regions* oder *Habitats and Biotops* (dt. Lebensräume und Biotope). Sofern also Querbezüge zu anderen AS enthalten sind, werden diese farblich markiert und zusätzlich mit Namensräumen vor dem Elementnamen im Kopf eines UML-Klassenelementes angedeutet (z.B. `Protected Sites Simple::ProtectedSite` als Bezug zum AS Simple; siehe UML-Diagramm 2.9 links oben). INSPIRE- oder ISO-Basistypen werden nur zur Typisierung von UML-Klassenattributen verwendet, nicht aber als eigenständige UML-Klassenelemente innerhalb eines AS bzw. UML-Packages geführt. Beispiele hierfür sind das INSPIRE-Element `Identifier`, ISO 19103-Elemente `DateTime`, `Boolean` und `Integer`, das ISO 19115-Metadatenelement `CI_ResponsibleParty` oder GML-Elemente aus ISO 19136, wie z.B. `GM_Object` oder `GM_MultiSurface`.

Auf der Ebene der INSPIRE-Objekttypen sind sieben **Stereotypen für UML-Klassenelemente** definiert:

- **featureType** deklariert einen INSPIRE-Spatial object type, gleichbedeutend zu einem WFS-Feature type. Jeder Spatial object type ist mit einem eindeutigen Identifikator attributiert (Datentyp: `Identifier`[71]) und häufig als geographischer Objekttyp mit einem Geometrieattribut ausgestattet (Datentyp: `GM_Object` oder Spezialisierungen wie z.B. `GM_MultiSurface`)
- **dataType** Strukturierender Datentyp ohne Identität, Abgrenzung zu einem *featureType*
- **codeList** Vordefinierte offene Werteliste
- **enumeration** Vordefinierte abgeschlossene Werteliste
- **union** Strukturierender Datentyp, der die Instanzierung nur eines seiner Unterelemente gestattet
- **placeholder** Platzhalter für nachträglich zu definierende Datentypen der Annex-Themen II/III
- **type** Abstrakter Elementtyp

INSPIRE-Objekttypen bzw. UML-Klassenelemente besitzen UML-Klassenattribute und -Assoziationen, die zu GML-Properties umgeformt werden. Es sind drei sich gegebenenfalls ergänzende **Stereotypen für UML-Klassenattribute und UML-Assoziationsrollen** definiert:

- **voidable** Stereotyp für Inhalte, die in den GML-Instanzdaten fehlen können, aber in der Realität vorliegen oder messbar sind. Bei derartigen Elementen müssen fehlende Werte explizit bekannt gegeben werden, und zwar über die Attributangabe `xsi:nil="true"`. Eine zusätzliche Begründung für die fehlende Wertebelegung kann mit dem Attribut `nilReason` beigefügt werden. Mögliche Gründe sind entweder a) der Wert ist unbekannt (Attributwert: `unknown`) oder b) wird nicht vom Datenbereitsteller in den eigenen Datenbeständen gepflegt (Attributwert: `other:unpopulated`)
- **lifeCycleInfo** Stereotyp für Inhalte, die zur Objektversionierung dienen. Diese betreffen entweder ein Speicherobjekt innerhalb eines Datensatz, z.B. mittels harmonisierter UML-Attribute `beginLifespanVersion` und `endLifespanVersion`, oder sind bezogen auf das beschriebene Realweltobjekt, z.B. mittels `validFrom` und `validTo`
- **version** Stereotyp, der nur für Assoziationsrollen verwendet werden kann. Gerichtete Assoziationen verweisen an ihrem Ende auf Spatial object types, die Assoziationsrollen repräsentieren. Der Stereotyp erlaubt nun, Assoziationsrollen abhängig von der jeweiligen Version eines Spatial object types zu deklarieren.

[71] für Spatial object types der Annex-I-Themen ist das `Identifier`-Attribut obligatorisch, während die Datenthemen aus Annex-II/III das Attribut nur noch optional führen. Dieser Unterschied rührt vermutlich daher, dass die Annex-I-Themen vorwiegend Geobasisthematiken behandeln, die für die Fachthemen aus Annex-II/III gewissermaßen als Bezugsrahmen dienen sollen, weshalb ihre Identifikatorattribute verpflichtend vorgeschrieben sind

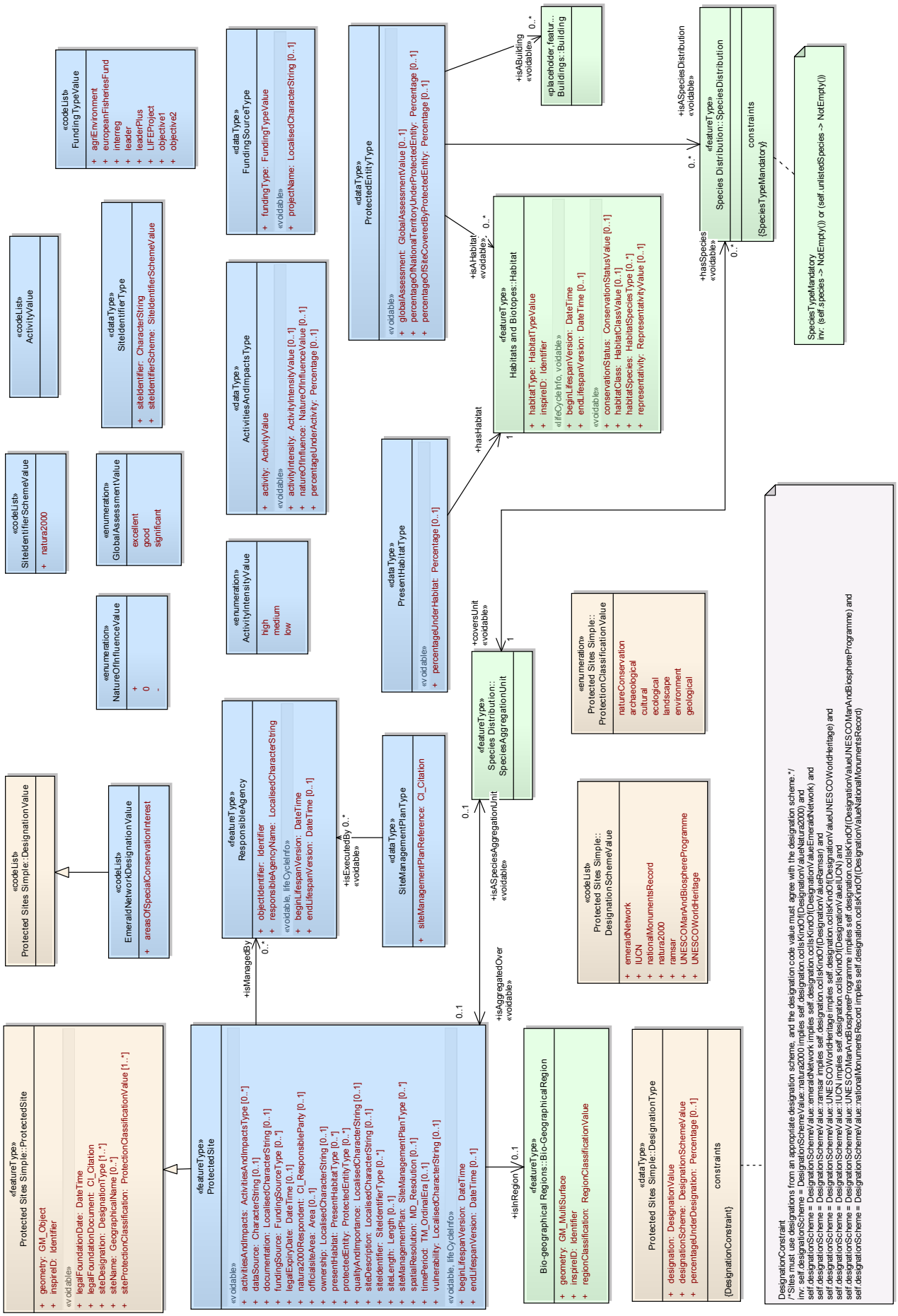


Abbildung 2.9 – UML-Modell Protected Sites - AS Full [INSPIRE Thematic Working Group „Protected sites“ 2010*b*]

Eine Besonderheit von AS Full sind die vielen Elemente und Restriktionen, die im Zusammenhang mit der Natura2000 EU-Berichterstattung stehen. Demnach sollen Natura2000- und ähnliche Reportdaten zukünftig konform zu INSPIRE erfasst werden. Spezielle Natura2000-Elemente sind z.B. die Attribute `natura2000Respondent` oder `siteLength` des Spatial object types `ProtectedSite`. Restriktionen sind in der Object Constraint Language (OCL) angegeben, einem Sprachbestandteil der UML. Im AS Full wurden mit OCL-Ausdrücken mehrere Natura2000-Wertebelegungen vorgeschrieben (siehe z.B. die Einschränkungsnotiz *DesignationConstraint* im Diagramm 2.9; links unten).

Das Drafting Team *Data Specifications* lässt den TWGs einige **Freiheitsgrade in der Ausgestaltung** der Datenspezifikationen. Beispielsweise können die TWGs die Auflösungsstufe vorgeben, in der fachspezifische Geometrien gespeichert bzw. übertragen werden sollen. Die TWG *Protected Sites* trifft dazu in der DS *Protected Sites* keine einschränkende Regelung, empfiehlt aber möglichst großmaßstäbige Geometrien, um Grenzvergleiche oder geometrische Operationen mit anderen Themendaten zu erleichtern. Die TWG legt hingegen fest, dass die Geometrien direkt in die Instanzdaten aufzunehmen sind (sogenannte *by coordinate*-Angabe) und keine entfernten Geometrieobjekte wie z.B. Katastergrenzen referenziert werden sollen (sogenannte *by reference*-Angabe). Diese und ähnliche Ausgestaltungen der Modellierung bleiben den TWGs vorbehalten, die oben besprochenen Modellierungsdetails sind davon aber ausgenommen und harmonisieren demzufolge bis zu einem gewissen Grad alle INSPIRE-Datenmodelle.

2.5.3 Netzwerkdienste

Wie eingangs erwähnt, sieht INSPIRE keine zentrale Datenhaltung vor, sondern eine Datenintegration auf dem Service-Level, so dass Originaldaten zweckmäßigerweise beim Datenerfasser verbleiben und dieser zugleich in der Rolle des Datenbereitstellers auftritt und Datenzugriffsdienste einrichtet. Da INSPIRE öffentliche Geodaten gleich welcher Aggregationsstufe adressiert, kommen nicht nur die EU-Mitgliedsstaaten als Datenbereitsteller in Frage, sondern auch die jeweiligen Bundesländer, Bezirke, Kreise oder Kommunen. Jeder EU-Mitgliedsstaat[72] steht in der Pflicht, sich über die verschiedenen administrativen Ebenen zu organisieren. Im Falle Deutschlands geschieht dies über eine Hierarchiebildung von GDI-Knoten bzw. Geodatendiensten, d.h. Kommunen binden sich in die Länder-GDIs ein. Die Länder-GDIs sind wiederum der deutschen NGDI namens *GDI-DE* untergeordnet, die den deutschen der 31 nationalen INSPIRE GDI-Knoten bildet. Hierzu werden verschiedenste Netzwerkdienste eingesetzt (siehe Abbildung 2.10; *Service-Layer*). Als kleinster gemeinsamer Nenner für INSPIRE-Netzwerkdienste wird HTTP/SOAP als Transport- und Applikationsprotokoll gemäß dem *Web Services-Interoperability* (WS-I) Basic Profile 2.0 [73] gefordert. Darüber hinaus basieren INSPIRE-Netzwerkdienste weitestgehend auf ISO- und OGC-Dienstestandardisierungen und erweitern diese um nur wenige Funktionalitäten, wie z.B. Mehrsprachigkeit oder spezielle Metadatenelemente innerhalb von Dienstbeschreibungen.

Die Diensttypen sind (gemäß Abbildung 2.10 von links nach rechts):

- **Registry Service**: ein Registrierdienst nach ISO 19135, der Register für INSPIRE-Referenzdatensätze vorhält (siehe Behandlung des Generic Conceptual Model in Abschnitt 2.5.2)
- **Discovery Service**: ein Metadatenkatalogdienst; entspricht weitestgehend dem OGC CS-W
- **View Service**: ein Kartendienst; entspricht weitestgehend dem OGC WMS
- **Download Service**: ein Rohdaten- bzw. Downloaddienst für den (gefilterten) Abruf von Geodatensätzen. Das Standardausgabeformat ist GML (GML 3.2.1 [Portele 2007]), weitere Formate sind optional. Das Drafting Team *Network Services* empfiehlt den Einsatz eines OGC WFS

[72] zuzüglich der vier EFTA-Staaten der europäischen Freihandelsassoziation: Norwegen, Island, Lichtenstein und die Schweiz
[73] OASIS-Standard für Internetkommunikation; siehe: http://ws-i.org/Profiles/BasicProfile-2.0-2010-11-09.html

- **Transformation Service**: Diensttyp, dessen Aufgabenprofile noch nicht endgültig feststehen. So könnte ein Transformation Service als Fassade vor nicht INSPIRE-konformen Karten- und Rohdatendiensten eingesetzt werden, um diese zu INSPIRE-Karten- und Downloaddiensten aufzuwerten. Ebenso kommt ein WPS für Datentransformationen in INSPIRE-konforme Datenhaltungen oder ein *OGC Web Coordinate Transformation Service* (WCTS) für ad-hoc-Koordinatentransformationen in Frage
- **Invoke Spatial Data Service**: Diensttyp, der die Erreichbarkeit von INSPIRE- und allgemeinen Geodatendiensten fördert. Darunter werden z.B. Bezahldienste, Orchestrierungsdienste für Diensteverkettungen und Geoportal-Unterstützungsdienste verstanden; gleichsam Funktionalitäten zur GDI-Administration sowie für Import- und Exportroutinen

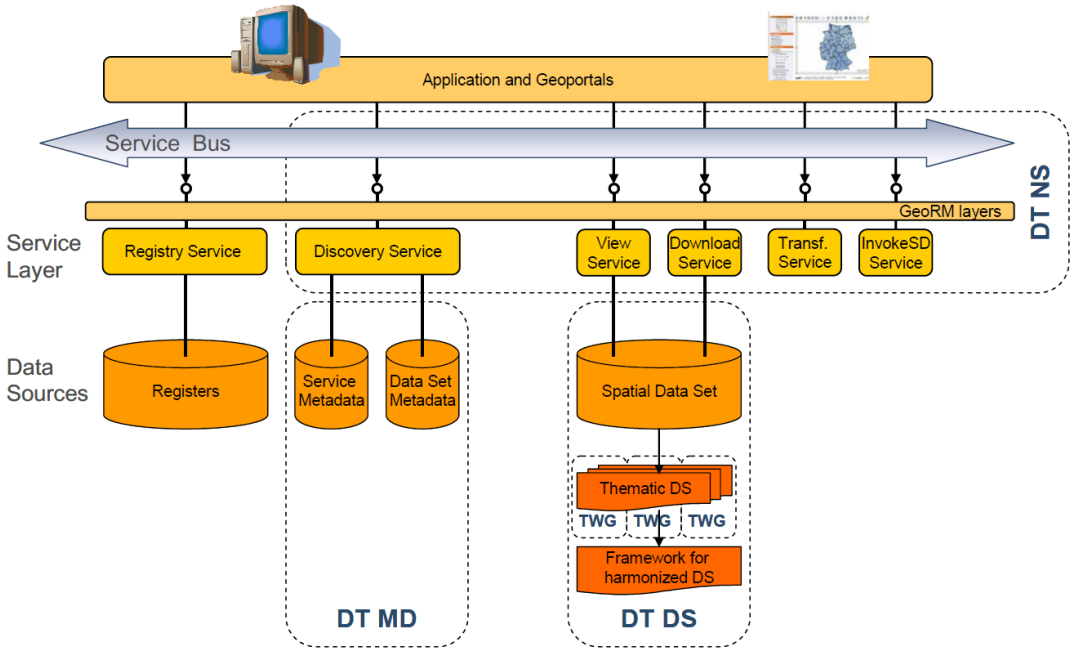


Abbildung 2.10 – INSPIRE-Architekturübersicht[74]

Der **INSPIRE-Downloaddienst** [INSPIRE „Network Services“ Drafting Team 2009] spielt für einen Semantic Web-Zugriff auf INSPIRE-Daten eine zentrale Rolle und bildet einen gewichtigen Bestandteil im Lösungsansatz dieser Arbeit (siehe Kapitel 4). Deshalb wird der Downloaddienst nun vorzugsweise näher betrachtet. Das verantwortliche Drafting Team benennt zwei mögliche Dienstarten, die von Datenbereitstellern genutzt werden können. Eine einfache Variante, ein sogenannter *Download Service for pre-defined data sets/ pre-defined parts of data sets* soll das Herunterladen vorprozessierter Datensätze nationaler oder untergeordneter Gebietsabdeckungen ermöglichen. Für diese Aufgabe eignen sich gewöhnliche FTP- oder HTTP-Webserver. Die technisch anspruchsvollere Alternative, der *Direct Access Download Service*, ist ein **OGC WFS der Version 2.0** [Vretanos 2010*b*]. Wie im Abschnitt 2.3.4 erläutert, erlaubt ein WFS-Dienst räumliche, textuelle und zeitliche Filterungen[75] und somit eine signifikante Einschränkung der Ergebnismenge im Gegensatz zum *Download Service for pre-defined data sets*.

Aufgrund der zusätzlichen Filterfunktionalität ist der INSPIRE-Direct Access Download Service bzw. WFS-Dienst die bevorzugte Dienstvariante. Dabei können Filterungen innerhalb von WFS-Datenanfragen individuell von WFS-Clients bestimmt werden (sogenannte *Ad hoc queries*). Eine zweite Möglichkeit besteht in der Verwendung vor-

[74] Quelle: INSPIRE *Network Services* Drafting Team [2008, S.8]; Abkürzungen: *DT MD*: Drafting Team *Metadata*, *DT DS*: Drafting Team *Data Specifications*, *DT NS*: Drafting Team *Network Services*

[75] explizite zeitliche Filteroperatoren stehen erst mit WFS-Version 2.0 bereit. Mit vergleichenden Filteroperatoren, wie z.B. `ogc:PropertyIsGreaterThan`, ist die Auswertung von Datumswerten auch in Vorgängerversionen möglich

definierter und geringfügig parametrisierbarer Filter, die im WFS-Dienst speziell für häufig wiederkehrende Anfragefälle registriert werden (sogenannte *Stored queries*, erst ab WFS Version 2.0 eingeführt). Die klassische Anfrageform ist die Ad hoc query, für die die Filteroperatoren der Sprache OGC Filter Encoding (FE) [Vretanos 2010*a*] zur Anwendung kommen (siehe Abschnitt 2.3.4). Laut dem Technical Guidance-Dokument [INSPIRE „Network Services" Drafting Team 2009] müssen INSPIRE-Downloaddienste vom Typ eines Direct Access Download Service alle FE-Filterausdrücke mit Ausnahme der komplexeren räumlichen Filter unterstützen. INSPIRE setzt für räumliche Filterzwecke nämlich nur den `BBox`-Operator verpflichtend voraus, der Geometrien auf Überschneidung mit einer Bounding Box prüft. *INSPIRE-optional* sind alle weiteren räumlichen FE-Operatoren (z.B. `Within` oder `Beyond`), die anstelle des `BBox`-Operators auch komplexe Vergleichsgeometrien zulassen (z.B. Polgyone und Geometriekollektionen). Allerdings ist davon auszugehen, dass INSPIRE-Downloaddienste auch die erweiterten räumlichen FE-Operatoren unterstützen, schließlich beherrschen die meisten am Markt befindlichen WFS-Implementierungen die komplette Bandbreite an räumlichen FE-Filteroperatoren. Abgesehen von den Vorgaben zur Filterfunktionalität fordert INSPIRE nur jene WFS-Dienstoperationen, die auch in der WFS-Spezifikation obligatorisch sind: die Operationen *GetCapabilities*, *DescribeFeatureType* und *GetFeature*. Wegen gewisser WFS-Profilabhängigkeiten, insbesondere zur Verwendung von *Stored queries*, wird die Unterstützung weiterer WFS-Dienstoperationen erforderlich, wie z.B. *ListStoredQueries*, *DescribeStoredQueries* usw.

Zu guter Letzt sei der Leser auf die aktuellen Entwicklungen zum OGC WFS-Dienst aufmerksam gemacht. Das INSPIRE-Drafting Team plant sehr zeitgemäß, nach Ansicht des Autors aber auch ein wenig voreilig den Einsatz von OGC WFS-Diensten der Version 2.0. Das Technical Guidance-Dokument für INSPIRE-Downloaddienste ist immernoch in einem Entwurfsstadium (Stand: Ende Januar 2012). Der Grund dafür ist vermutlich die Spezifikation des OGC WFS 2.0, die zugleich als ISO-Standard 19142 geführt wird. Die Spezifikation ist erst im November 2010 finalisiert worden und wird derzeit noch von keiner Implementierung unterstützt (Stand: Ende Januar 2012[76]). Ob bis zur ersten Datenbereitstellungsfrist Ende 2012 konforme Implementierungen verfügbar sein werden, ist noch nicht abzusehen. Nichtsdestotrotz lassen sich repräsentative Testläufe im Rahmen dieser Arbeit auch mit der Vorgängerversion OGC WFS 1.1 [Vretanos 2005*b*] durchführen, da die Funktionalität beider Versionen weitgehend identisch ist. Im Unterschied zur Version 2.0 beherrscht ein WFS 1.1 zwar keine Stored queries. Da sich der Lösungsansatz in Kapitel 4 aber ausschließlich auf Ad hoc queries konzentriert, können auch mit einem WFS 1.1 sämtliche relevanten Tests durchgeführt werden.

2.5.4 Datenbereitstellung über Netzwerkdienste

Nachdem die vorangegangenen Abschnitte in die INSPIRE-Datenspezifikationen und -Netzwerkdienste einführten, wird in diesem Abschnitt abschließend dargestellt, welche Schritte und Komponenten in Frage kommen, um existierende Datenbestände INSPIRE-konform aufzubereiten und INSPIRE-Netzwerkdienste operativ in Betrieb zu nehmen. Damit schließt der Abschnitt das Grundlagenkapitel und die Betrachtung des GDI-Projektes INSPIRE ab. Die konkrete Umsetzung und praktische Einrichtung einer INSPIRE-Plattform mitsamt Daten und Diensten ist Thema des Abschnittes 5.3 *Implementierung eines Prototypen*.

Der Abschnitt 2.5.2 hat die INSPIRE-Verbindlichkeiten bezüglich der Datenbereitstellung angesprochen. Behördliche Stellen, die INSPIRE-relevante Daten führen und augrund der Richtlinie aktiv werden müssen, pflegen ihre Datenbestände in der Regel in bereits vorhandenen Datenmodellen, die nur den ursprünglichen und nicht den INSPIRE-Anforderungen genügen. Die INSPIRE-Pflichten werden zusätzlich zu herkömmlichen Geschäftsprozessen zu erfüllen sein, so dass sich die Datenbereitsteller mit zwei Problemstellungen konfrontiert sehen. Einerseits sind ihre Datenbestände über eine Modelltransformation in das INSPIRE-Zielschema des jeweiligen Annex-Themas zu überführen. Andererseits ist der Zugriff auf die transformierten Daten über INSPIRE-Dienste einzurichten. Abbildung 2.11 zeigt

[76] siehe OGC-Register konformer Implementierungen unter: http://www.opengeospatial.org/resource/products/compliant

die zwei möglichen Lösungswege auf, die sich für beide Arten von INSPIRE-Downloaddiensten gleichermaßen eignen. Die Transformation der Instanzdaten kann entweder on-the-fly während des Datenabrufes (Option 1) oder in einer Vorabprozessierung und statischen Speicherung geschehen (Option 2).

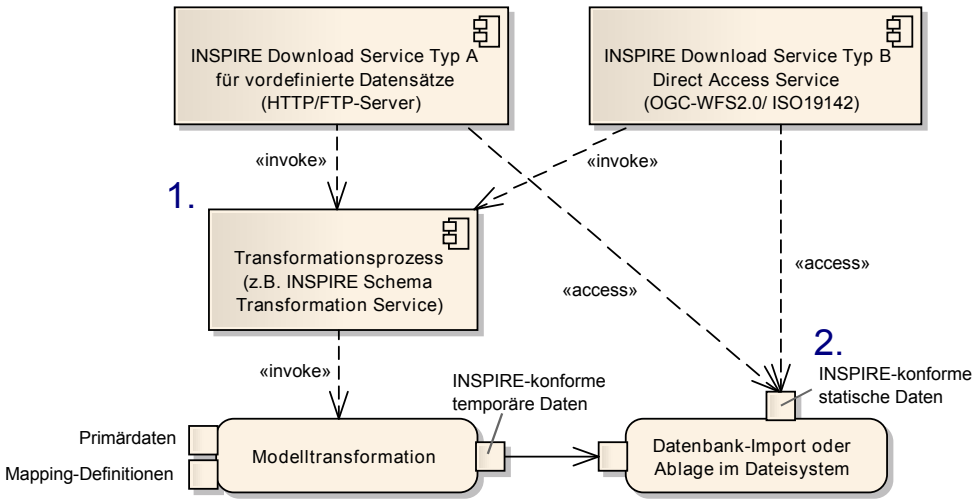


Abbildung 2.11 – Datenaufbereitung und Bereitstellung über INSPIRE-Netzwerkdienste

Der erste Lösungsweg ist im Grunde nur für kleinere Primärdatenbestände praktikabel, die sich performant zur Laufzeit transformieren lassen. Dabei startet der Benutzer die Datenanfrage an einen INSPIRE-Downloaddienst, der die Anfrage an einen Transformationsprozess weiterleitet. Der Transformationsprozess ist entweder als Dienst (z.B. ein INSPIRE *Schema Transformation Service*) oder als Desktop-Routine eingerichtet. Er wandelt die Primärdaten temporär in INSPIRE-konforme GML-Daten um und schickt sie an den Client zurück. In der Dissertation von Staub [2009] wird anstelle eines zusätzlichen Transformationsdienstes ein modellbasierter WFS-Dienst in Betracht gezogen. Dieser WFS-Dienst lässt sich sowohl als INSPIRE-Downloaddienst als auch für Modelltransformationen einsetzen. Zur Definition der Transformationsregeln entwickelt Staub die konzeptionelle und auf UML-basierende Abbildungsregelsprache *UMLT*. Obwohl hier ein großes Potential erkennbar ist, müssen sich diese und weitere Ansätze zunächst einmal für größere Datenmengen als praxistauglich erweisen. Letztlich setzen Modelltransformationen, die zur Laufzeit durchgeführt werden, eine sehr leistungsstarke IT-Plattform hinsichtlich der Prozessorleistung und des Arbeitsspeichers voraus. Hingegen kommt der alternative zweite Lösungsweg mit deutlich geringeren Systemvoraussetzungen aus. Er beinhaltet die Vorabtransformation der Instanzdaten, die mit ETL-Werkzeugen (*Extract/Transform/Load*) entweder einmalig oder aufgrund häufiger Datenaktualisierungen auch zyklisch durchgeführt werden kann. Das Transformationsergebnis lässt sich in räumliche Datenbanken einlesen bzw. als Geodatenformat im Dateisystem ablegen. Die statische Speicherung entkoppelt den Vorgang der Modelltransformation von den späteren Datenanfragen an INSPIRE-Downloaddienste, so dass während der Datenanfragen kein unnötiger Zeitverzug entsteht.

Die am Markt befindlichen Software-Lösungen verfolgen vorwiegend den zweiten Lösungsweg. Ein derartiges Beispiel ist die INSPIRE-Architektur der Firma *Conterra*. Deren Lösung integriert die ETL-Software *Feature Manipulation Engine* (FME[77]), die Benutzerhilfen mit Verweisen zu INSPIRE-Datenspezifikationen, vordefinierten graphischen Transformationsbausteinen (*FME-Transformatoren*) und INSPIRE-Wertelisten anbietet. Die FME schreibt die Transformationsergebnisse in eine proprietäre Geodatenbank (*ESRI Geodatabase*), die mit INSPIRE-Datenmodellen vorkonfiguriert ist. Auf dieser Datengrundlage setzt das ESRI-Softwarepacket *ArcGIS for INSPIRE*[78] auf und gestattet das Administrieren von INSPIRE-Katalog-, Karten- und Downloaddiensten. Der INSPIRE-Datenbereitsteller erhält

[77] Produktseite *INSPIRE Solution Pack for FME*: http://www.conterra.de/de/software/fme/desktop/desktop-inspire-solution.shtm
[78] Produktseite: http://www.conterra.de/de/software/esri-software/afi.shtm

mit den ESRI/ Conterra-Produkten also sowohl eine Transformationssoftware für Geodaten als auch INSPIRE-konforme Diensteimplementierungen. Eine lohnenswerte Alternative hierzu ist das Open Source-Framework *Deegree* Version 3.0 der Firma *latlon*, das in der Ausführung *Deegree3 inspireNode*[79] INSPIRE-Karten- und Downloaddienste enthält und konzeptionelle Stärken hinsichtlich der Datenanbindung und -speicherung aufweist. Näheres dazu ist dem Abschnitt 5.3 zu entnehmen, der die Software im Kontext eines Prototypen vorstellt.

[79] Produktseite: http://wiki.deegree.org/deegreeWiki/InspireNode?action=show&redirect=deegree3%2FInspireNode

3 Anforderungsanalyse

In diesem Kapitel werden anhand von fiktiven Anwendungsszenarien potentielle Nutzersichten erörtert und daraus Anforderungen an eine technische Unterstützung dieser Anwendungsszenarien abgeleitet. Die Anforderungen können zunächst nur aus der Perspektive des Anwenders formuliert werden. Alle weiteren für eine Systementwicklung relevanten, technischen Anforderungen setzen einen groben Entwurf der Systemarchitektur voraus und sind deshalb nicht Gegenstand des Kapitels. Sie werden in den folgenden Kapiteln 4 *Lösungsansatz* und 5 *Konzeption und Implementierung eines Prototypen* besprochen.

Wie sich im Abschnitt 3.2 zeigen wird, sind die resultierenden Nutzeranforderungen recht allgemeiner Natur, d.h. sie sind nicht INSPIRE-projektspezifisch, denn sie betreffen im Wesentlichen den Zugriff auf Online-Datenquellen, die über Internetdienste verfügbar gemacht eine nutzerselektierte Abfrage erlauben. Dennoch lohnt sich deren Betrachtung, um sich einen realistischen Anwendungskontext vor Augen zu führen und zu verifizieren, inwieweit sich die angestrebte Software-Lösung für einen konkreten Einsatz eignet, d.h. die Anforderungen aus Anwendersicht unterstützt. Eine abschließende Diskussion der hier genannten Inhalte findet in Kapitel 6.1 statt.

3.1 Anwendungsszenarien

Die in Tabelle 3.1 beschriebenen Anwendungsszenarien sind klassifiziert nach Anwenderprofilen (Spalte *Nutzertyp*), die vom Gelegenheitsnutzer bis hin zum Fachmann reichen. Mit dieser Klassifizierung wird weniger die Abgrenzung zwischen einer Freizeitbeschäftigung und einer professionellen Tätigkeit bezweckt, vielmehr soll sie ein grobes Maß für die Erfahrung des Anwenders im Umgang mit Internet- bzw. Semantic Web-Daten und deren Verarbeitung vorgeben. Für jeden Nutzertyp werden exemplarische Anwendungsszenarien beschrieben, die konkrete Anwender und ihre Motivation zur Datenrecherche benennen (Spalten *Akteur* und *Motivation*).

Tabelle 3.1 – Übersicht der Nutzertypen und Anwendungsszenarien

Nr.	Nutzertyp	Akteur	Motivation	Externe Quellen
I	Gelegenheitsnutzer (Suche über eine Datenkategorie)	Naturfreund	Erhalt der Natur	keine
II	Fortgeschrittener Nutzer (Suche über diverse Datenkategorien)	Lehrer	Naturerfahrung bzw. Erholung	Geonames LinkedGeoData
III	”	WWF-Aktivist	Aktiver Naturschutz	GeoSpecies Eurostat/Riese
IV	Erfahrener Nutzer (Suche mit zusätzlicher Daten-integration/-umformung)	Energieversorger	Naturerkundung bzw. Ressourcenexploration	Datenintegration versch. Quellen

Für die Verwendung sind primär INSPIRE-Datensätze - hier INSPIRE-Schutzgebietedaten - vorgesehen. In der Spalte *Externe Quellen* werden verfügbare Linked Data-Datentöpfe aufgeführt, die zwecks Informationsverschneidung mit INSPIRE-Daten für Fragestellungen der Anwendungsszenarien II bis IV von Interesse sein können. Hierbei handelt es sich um eine Auswahl bedeutender Datenquellen aus dem Fundus des Semantic Web, je nach Bedarf können aber auch alternative Quellen gewählt werden. Insbesondere bietet INSPIRE nach einer vollständigen inhaltlichen Implementierung einen, den genannten externen Quellen vergleichbaren Datenbestand. Den nun anschließenden Szenariobeschreibungen folgt jeweils eine kurze Analyse der erforderlichen Dateninhalte und Verarbeitungsprozesse.

Szenario I

Ein Ausflug führt eine Gruppe von Naturfreunden in ein entferntes und für die Gruppe noch unbekanntes Wandergebiet, das durch seine Kulturlandschaft mit malerischen Ortschaften und ausgedehnten Wäldern lockt. Während der Tagestour macht die Wandergruppe zwei auffällige Beobachtungen, bei denen es jedoch fraglich ist, ob sie sich in oder außerhalb eines Schutzgebietes abspielen. Erstens wird eine Gruppe von Wildcampern beobachtet, die sich in gemütlicher Stimmung ein Lagerfeuer anzünden. Trotz geäußerter Bedenken der Naturfreunde lassen sich die Wildcamper nicht zum Abbrechen der Zelte bewegen. Ein Naturfreund erfasst für alle Fälle die Standortkoordinaten. Zweitens haben forstwirtschaftliche Arbeiten ersichtlichen Schaden entlang eines Waldweges hinterlassen. Zwar liegen die Abholzungsstellen weiter entfernt, doch der Transport von Gehölz und Stämmen hat zu deutlichem Schaden im Buschwerk am Rande des Weges geführt. Die per GPS-Empfänger getrackte Wanderroute wird mit Markern versehen, um die betroffene Wegesstrecke erneut auffinden zu können. Die Naturfreunde verabreden bei der Rückfahrt, dass eine Person aus der Gruppe im Internet recherchiert, ob die Standorte der Naturgefährdung/ Waldschäden in einem Schutzgebiet liegen (die weniger-reglementierten Landschaftsschutzgebiete (LSG) bleiben bei der Auswertung außen vor). Falls dies zutrifft, soll die zuständige Behörde umgehend ausfindig und Meldung gemacht werden.

Das Szenario bedingt eine Datenlage, die drei wesentliche Informationen zu Schutzgebieten bereithält. Erstens die geometrische Ausdehnung zum Vergleich mit den Koordinaten der Fundstellen. Zweitens die Einordnung in einzelne Schutzgebieteklassen (Naturschutzgebiet etc.), so dass wie gefordert großflächige LSGs aus der Analyse ausgeklammert werden können. Drittens die Angabe bzw. Kontaktadresse einer zuständigen Behörde, die im Notfall informiert werden kann. Alle drei Informationen beinhaltet das INSPIRE-Datenmodell Protected Sites, weshalb die Hinzunahme weiterer Datenquellen entfallen kann.

Szenario II

Ein Lehrer plant mit der Umwelt-AG eine Wochenendfahrt hinaus in die Natur. Die Anfahrt soll nicht allzu lange dauern, der Zielort dennoch weit genug von der Schule entfernt liegen, damit in der für die Schüler unbekannten Umgebung entsprechende Klassenfahrtstimmung aufkommen kann (mind. ca. 50 km). Der Lehrer stellt sich als Rahmenprogramm zum Naturerlebnis mehrere Wanderungen vor, die in ein Naturschutzgebiet oder ein Schutzgebiet vergleichbar hochwertiger Schutzkategorie führen. Das Schutzgebiet soll älteren Bestandes sein, so dass sich die Natur vor Ort schon seit langem regenerieren konnte (z.B. Schutzstatus zumindest seit 1990). Zusätzlich soll eine Jugendherberge in erreichbarer Nähe zum Schutzgebiet aufgefunden werden (weniger als 3 km Fußweg; Luftlinie). Der Lehrer führt eine Standortanalyse zu den genannten Kriterien über Online-Datenquellen durch und wählt aus den in Frage kommenden Örtlichkeiten ein Reiseziel aus. Nun interessieren ihn obendrein wissenschaftlich fundierte Informationsquellen zum Schutzgebiet seiner Wahl und weitere Freizeitangebote in der näheren Umgebung für ein lehr- wie abwechslungsreiches Wochenende.

Erneut werden Schutzgebietedaten angefordert, und zwar mit der Attributierung der geometrischen Ausdehnung, der Klassifizierung in Widmungsklassen, des Gründungsdatums und zusätzlich verlinkter Informationsmaterialien (z.B. Gesetzestext, Bewirtschaftungsplan, Touristeninformation etc.). Außer den INSPIRE-Schutzgebietedaten benötigt der

Lehrer zwei weitere Informationsquellen. Erstens ist der Standort der Schule über eine Geokodierung des Ortsnamens zu bestimmen, wofür sich das LOD-Projekt *Geonames* anbietet. Zweitens sind Standorte von Jugendherbergen und Freizeitangeboten einzukalkulieren. Als Bezugsquelle kommt das LOD-Projekt *LinkedGeoData* in Betracht. Daraus ergibt sich eine verteilte Suche über mindestens drei Informationsquellen und eine aufwendige geometrische Analyse, z.B. eine (temporäre) konzentrische Bufferbildung um die Standorte der Schule, der Jugendherbergen und Freizeitangebote sowie die geometrische Überschneidung berechneter Buffer mit Schutzgebietegrenzen. Vermutlich sind mehrere Anfrageschritte gemäß eines GIS-Workflows in Reihe zu schalten.

Szenario III

Ein engagierter WWF-Mitarbeiter hat in einem nahe gelegenen Wiesenstück zufällig eine Brutstätte der Tierart X (z.B. die nach der Roten Liste stark gefährdete Vogelart *Brachpieper*) ausgemacht. Bei mehrtägiger Erkundung des Geländes findet der ornithologisch Bewanderte weitere drei Nistplätze der gleichen Art vor. An die ihm bekannte Datenbank Y (z.B. GeoSpecies) stellt er Anfragen nach weiteren Vorkommnissen von Nist- bzw. Rückzugsplätzen der Vogelart. Er weis nun, dass die Vogelart hierzulande stark verbreitet ist. Ebenso, dass mehrere nah gelegene Schutzgebiete eingerichtet sind. Jedoch ist für ihn fraglich, ob die besagte Vogelart im Speziellen genügend Schutz geniesst. Falls dies nicht zutrifft, will er seinem Bundesland die Ausweisung eines eigenständigen FFH-Gebietes gemäß der EU-Vogelschutzrichtlinie vorschlagen; natürlich in Absprache mit seiner WWF-Ortsgruppe/ WWF Deutschland. Damit einhergend versucht er proaktiv, den deutschen Nachholbedarf bei der Umsetzung der EU-Vogelschutzrichtlinie zu begegnen.

Seine Diagnose stützt sich auf die Beantwortung folgender Fragestellungen:

- liegt das Beobachtungsgebiet der Nistplätze innerhalb von Grenzen existierender Schutzgebiete? (Kriterium: z.B. mind. 2/3 der Fläche)
- sind existierende Schutzgebiete mit ihren Schutzzielen geeignet? (Kriterium: z.B. kein LSG mit Agrartätigkeit)
- ist die Art X namentlich in den Schutzzielen genannt (findet eine regelmäßige öffentliche Registrierung statt)?

Zur Absicherung, dass der Vogelart nicht doch bereits genügend Schutz in anderen ausgewiesenen Gebieten zuteil wird, sucht der WWF-Mitarbeiter vorab nach allen in der Nähe gelegenen Schutzgebieten, die in Verbindung mit der Vogelart X stehen. Mit diesem gesammelten Wissen kann er nun in Aktion treten.

Der wesentliche Unterschied zum Szenario II besteht darin, dass die genannten Informationsquellen, z.B. INSPIRE-Schutzgebiete und GeoSpecies, nicht nur geometrisch, sondern auch attributiv über die Bezeichnung einer Tierart miteinander verschnitten und einer gemeinsamen Analyse unterzogen werden. Das Szenario setzt u.a. eine fortgeschrittene inhaltliche INSPIRE-Realisierung voraus, die eine Verlinkung mehrerer INSPIRE-Thematiken herbeiführt, wie in diesem Fall die logische Verknüpfung von Schutzgebieten mit den darin vorkommenden Arten und ihren Verteilungen (*Species distribution*).

Szenario IV

Ein Privatunternehmen gewinnt eine Ausschreibung zum Aufbau einer Informationsplattform der Energiewirtschaft (eines oder mehrerer Energieversorger). Vorrangig ist die Suche nach neuen Energieträgern (z.B. Erdgasvorkommen). Neben geologischen und infrastrukturellen Daten spielt das Thema Schutzgebiete selbstverständlich eine wichtige Rolle, um jene Landschaftsteile aus der erwähnten Analyse auszusparen. Dem beauftragten Dienstleister liegen trotz der thematisch umfangreichen Datenhaltungen der Energieversorger keine für Deutschland flächendeckenden und gleichzeitig formellen Schutzgebietsinformationen vor, um eine automatische Lokalisierung von günstigen Versuchs- und Erprobungsstandorten vorzunehmen. Die Daten müssen für diesen Zweck nicht *belastbar* sein, jedoch zumindest aus

öffentlicher Hand stammen, um das Risiko fehlerhafter oder veralteter Daten im ersten Schritt der iterativen Auswertung möglichst gering zu halten. Der Auftragnehmer richtet sich an Umweltbehörden mit der Bitte um Datenabgabe gemäß der *Public Sector Information*-Richtlinie (PSI) zur Weitergabe von öffentlichen Informationen an die Privatwirtschaft und bekommt diese von der zuständigen Umweltbehörde ausgehändigt. Kernaufgabe des Dienstleisters ist nun eine intelligente Datenintegration aller Datenthemen und umfasst folgende Gesichtspunkte:

- Prüfung der Themendaten und Schemata (z.B. auf Elementabhängigkeiten, Abgleich mit Referenzdatensätzen)
- Verknüpfung der Themenfelder untereinander (z.B. über fachliche oder geographische Attribute)
- Datenaufbereitung für die automatische Auswertung als auch Sichtkontrolle durch einen Sachbearbeiter

Der hier skizzierte professionelle Einsatz beinhaltet verschiedenste Abläufe der Semantic Web-Datenverarbeitung. Einerseits werden verschiedene Informationsquellen (zumindest Schutzgebiete-, Geologie- und Geobasisdaten) abgerufen, d.h. Daten gesucht, gefiltert und die Ergebnismenge in einem gängigen Semantic Web-Format heruntergeladen. Den INSPIRE-Datensätzen ist bei der Datenakquise des Anwendungsszenarios besondere Bedeutung beizumessen, weil sie aufgrund der behördlichen Datenhoheit eine hohe Datenkonsistenz, -aktualität und -vollständigkeit vermuten lassen. Eine Weiterverarbeitung könnte aus verketteten Umformungs- und Integrationsprozessen, Inferenzbildungs- und Aktualisierungsroutinen bestehen und womöglich die Anzeige von Objektgeometrien in einem Kartenbetrachtungswerkzeug einschliessen. Viele der Szenario-unterstützenden Software-Bausteine sind außerhalb des Einflussbereiches der Datenbereitstellung und gehen über die Bedienung einer Rechercheoberfläche für eine verteilte Suche wie in den Szenarien II und III hinaus. Deshalb entlasten den Anwender am ehesten Vorkehrungen, die auf einen zweckmäßigen Datenexport abzielen, wie beispielsweise gängige Transferformate oder harmonisierte und leicht zu parsende Inhaltskodierungen von Geometrien und häufigen Informationstypen, z.B. Datumsfelder und numerische Werte.

3.2 Nutzeranforderungen

Die unten aufgeführten Anforderungen lassen sich aus den vorangegangenen Anwendungsszenarien ableiten. Sie werden hier in funktionale (*FA*) und nicht-funktionale Anforderungen (*NFA*) differenziert. Bis auf FA7 beziehen sich alle funktionalen Anforderungen auf eine Server-seitige Funktionalität und sind gemäß der Zielsetzung der Arbeit zu berücksichtigen (siehe hierzu die thematische Abgrenzung, Kapitel 1 Abschnitt 1.3). Dabei ist darauf hinzuweisen, dass die *weichen* Kriterien der nicht-funktionalen Anforderungen nur bis zu einem gewissen Grad beeinflusst werden können und sehr stark mit übergeordneten Strukturen zusammenhängen. Beispielsweise sind sie abhängig von Übertragungsgeschwindigkeiten im Internet, der inhaltlichen Qualität von externen Datenhaltungen und ihrer Verfügbarkeit über Diensteschnittstellen.

Funktionale Anforderungen

FA1 Recherche/ Zugriff auf relevante Datensätze, deren Stellenwert sich u.a. ermessen läßt durch:

- Fachliche Merkmale: differenzierte und klar umgrenzte Themengebiete
- Zeitliche Merkmale: aktuelle und historische Datenbestände, hohe zeitliche Auflösung
- Räumliche Merkmale: große räumliche Abdeckung, dem Thema angemessene räumliche Auflösung
- Zuständigkeit und Datenqualität: *siehe nicht-funktionale Anforderung NFA1*

FA2 Vielseitige Filterungsmöglichkeiten während der Datenrecherche mit selektivem Durchgriff auf alle charakteristischen Datenfelder. Typische Anfragefälle gilt es zu unterstützen:

- Filtern über klassifizierende Attribute, z.B. *Welches Schutzgebiet gehört dem UNESCO-Welterbe an?*
- Filtern über geographische Attribute, z.B. *Welche Habitate liegen innerhalb von Schutzgebieten?*
- Filtern über temporale Attribute, z.B. *Welches Schutzgebiet hat sein Gründungsdatum in den 80ern?*
- Filtern über Messwerte, z.B. *Welches Schutzgebiet hat eine Flächengröße, die zum Durchwandern einlädt?*

FA3 Einbindung externer Datenquellen/ Dienste

FA4 Verschneidung und Integration verschiedener Datenquellen für eine umfassende Datenanalyse, idealerweise *on-the-fly* oder zumindest ohne viele Zwischenspeichervorgänge

FA5 Hohe Informationsvernetzung durch Anreicherung der Datensätze mit Referenzen auf entfernte Datenquellen und Register, um den semantischen Informationsgehalt zu steigern

FA6 Etablierung von Download- und Reproduktionsmechanismen (Übertragungsprotokolle, Transferformate etc.) für die Datenweitergabe bzw. Integrationsaufgaben.

FA7* Benutzerunterstützung bei Suchanfragen und der Datensichtung (GUI-Elemente einer *facettierten Suche*, Kartenbetrachtungswerkzeuge etc.)

Nicht-funktionale Anforderungen

NFA1 Datenkonsistenz und allgemeine Richtigkeit, Vollständigkeit, Aktualität

NFA2 Stabilität und stete Verfügbarkeit von Online-Datenquellen

NFA3 Performanz von Suchanfragen, kurze Zugriffszeiten und schnelle Auswerteprozeduren von Webdiensten

NFA4 Reproduzierbarkeit von Rechercheergebnissen

NFA5 Einfacher Umgang mit Konzepten (Objekttypen) und Instanziierungen

NFA6 Harmonisierte Datenwelten ohne Redundanz

NFA7 Weitreichender Einsatz von Standards bezüglich Zugriffsschnittstellen, Transferformaten und Referenzdaten

4 Lösungsansatz

Das vorrangige Ziel dieser Arbeit besteht darin, INSPIRE-Datenbestände unter Beachtung von Linked Data-Prinzipien im Semantic Web zu veröffentlichen und damit eine verteilte Suche über verschiedene INSPIRE-Datenquellen zu ermöglichen. Für ein solches *Semantic Enablement* für INSPIRE-Datenbestände bedarf es zweier Schlüsselfaktoren:

1. eine Strategie zur Modellierung von INSPIRE-Ontologien
2. ein Architekturkonzept für SPARQL-Anfragen auf INSPIRE-Ontologien

Die beiden Kapitelabschnitte 4.1 und 4.2 gehen nun getrennt auf die genannten Schlüsselfaktoren ein und erläutern die Lösungen dieser Arbeit, die in Kapitel 5 vertieft werden.

4.1 Modellierung von INSPIRE-Themenontologien

Die Bildung von INSPIRE-Ontologien, d.h. die Überführung von INSPIRE-Konzepten und -Instanzdaten in Semantic Web-Speicherformen, ist eine wesentliche Voraussetzung dafür, dass reguläre Semantic Web-Werkzeuge INSPIRE-Informationen interpretieren können. Als Grundlage für INSPIRE-Ontologien kommen sowohl die INSPIRE UML-Datenmodelle (konzeptionelle Modelle) in Betracht als auch die daraus abgeleiteten Transferformate (logische Modelle), die in INSPIRE als GML-Applikationsschemata entworfen sind. Die **Wahl des Quellmodells**, ob UML oder GML, wird dadurch erleichtert, dass die UML-Modelle eine umfangreichere Semantik bieten, die insbesondere in den INSPIRE-spezifischen UML-Eigenschaftswerten und -Stereotypen enthalten ist (siehe Abschnitt 2.5.2). Diese und andere semantischen Details, wie z.B. Mehrfachableitungen von UML-Klassen, können nicht verlustfrei in GML-Applikationsschemata abgebildet werden, weshalb hier die Festlegung auf die INSPIRE UML-Modelle als Quelle für INSPIRE-Ontologien erfolgt.

Aufgrund der themenorientierten Abstimmungsprozesse bilden die INSPIRE UML-Modelle sogenannte **Domänenmodelle**. Hierbei ist zu beachten, dass je INSPIRE Annex-Thema nicht nur ein, sondern in den meisten Fällen mehrere UML-Modelle spezifiziert werden. Im Semantic Web werden Domänenmodelle in Form von Domänenontologien definiert (siehe Abschnitt 2.2.4). Der Lösungsansatz sieht vor, aus den UML-Modellen eines INSPIRE Annex-Themas je eine Domänenontologie zu bilden. In der weiteren Arbeit wird aufgrund dieser 1:1 Beziehung eines INSPIRE Annex-Themas zu einer INSPIRE-Domänenontologie der Begriff *INSPIRE-Themenontologie* gebraucht.

Anstelle der UML wird im Semantic Web-Umfeld bevorzugt die **konzeptionelle Schemasprache** OWL basierend auf RDF angewendet.[80]. Die beiden Sprachen unterscheiden sich prinzipiell dadurch, dass sich die OWL im Gegensatz zur UML auch als logische Schemasprache eignet, indem OWL-Vokabulare die Struktur von OWL-Instanzdokumenten vorgeben.[81] Das bedeutet, dass die OWL sowohl auf konzeptioneller als auch logischer Ebene fungiert. Die Abbildung 4.1 veranschaulicht diesen ambivalenten Charakter der OWL und zeigt zugleich die gewählte Strategie auf, um mit dieser Ambivalenz umzugehen. Dazu werden einerseits INSPIRE-Konzepte aus den UML-Modellen in die Themenontologien überführt (konzeptionelle Modellebene; siehe *Modellüberführung* in Abbildung 4.1). Andererseits sind Regeln

[80] für einfache Anwendungsfälle lässt sich zur Modellierung auch RDF-Schema (RDFS) einsetzen

[81] das konkrete Transferformat spielt dabei eine untergeordnete Rolle, weil sich Semantic Web-Formate, wie z.B. RDF/XML oder Turtle, verlustfrei aufeinander abbilden lassen; siehe Abschnitt 2.2.2

aus den Spezifika der GML-Applikationsschemata abzuleiten, um die Datentransformation zwischen GML- und OWL-Instanzdaten durchführen zu können. Diese werden ebenfalls in die Themenontologien überführt (logische Modellebene; siehe *Ableitung von GML/OWL-Abbildungsregeln* in Abbildung 4.1).

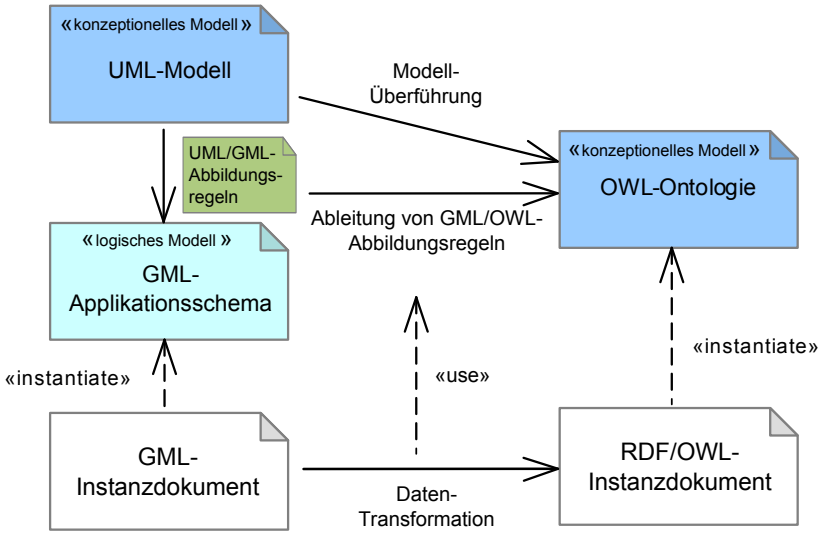


Abbildung 4.1 – Modellierungsunterschiede und Transformationswege zwischen UML+GML und OWL

Die resultierenden INSPIRE-Themenontologien sollen diverse Zwecke erfüllen. Die untere Auflistung liefert einen Überblick über die primären Verwendungsziele.

INSPIRE-Themenontologien dienen als:

1. Zielmodell und Speichergrundlage für GML/OWL-Abbildungsregeln
2. Konfiguration einer SPARQL-Anfrageschnittstelle
3. Harmonisierung von INSPIRE-Referenzinformationen
4. Basis für das Ontologie-Alignment mit Upper-Ontologien

Die Themenontologien werden als **Zielmodelle** für die Transformation von GML-Instanzdaten des jeweiligen INSPIRE Annex-Themas in RDF/OWL-Instanzdaten verwendet. Um zu klären, welche Art der Datentransformation am ehesten den Zweck erfüllt, ist zunächst auf die Fachliteratur zu verweisen, die zwischen zwei grundlegenden Arten der Datentransformation unterscheidet. Staub [Staub 2009] bezeichnet sie als a) *Modelltransformation* (semantische Transformation) und b) *Strukturumbau auf Formatebene* (strukturelle Transformation). Eine Modelltransformation bewerkstelligt üblicherweise verschiedene Abwandlungen und Neubildungen von Objekten und Objektattributen und führt in der Regel zu erheblichen semantischen Veränderungen und sehr wahrscheinlich auch zu Informationsverlusten. Hingegen beschränken sich strukturelle Transformationen auf Umformungen der Elementstruktur unter weitestgehender Beibehaltung der Inhalte und möglichst analoger Transferformate. Obwohl GML und RDF/OWL einige grundlegende Ähnlichkeiten aufzuweisen haben (siehe Abschnitt 2.3.2) und GML wie auch das gebräuchliche Transferformat RDF/XML auf XML basieren, ist die hier angestrebte Transformation von GML nach RDF/OWL mehr als nur ein einfacher Strukturumbau. Ausgewählte und häufig wiederkehrende GML-Elementtypen bzw. -Elementinstanzen werden speziell behandelt; u.a. betreffen die teils semantischen Umformungen:

- die eindeutige Identifikation von Referenzinformationen (z.B. Codelist-Werte, CRS) über auflösbare URIs

- die Modellierung von Geometrien und Messwerten konform zu den Vokabularen GeoSPARQL bzw. MUO [82]
- eine Verflachung der Elementstruktur von stark verschachtelten INSPIRE- und ISO-Datentypen

Letztlich aber soll die hier vorgeschlagene Datentransformation ins Semantic Web zu keiner inhaltlichen Verfremdung führen und die jeweilige INSPIRE-Themenontologie erkennbar den zugehörigen INSPIRE UML-Modellen gleichen. Erreicht wird dies mit einer größtmöglichen Übernahme der INSPIRE-Elementnamen und -Elementdefinitionen in die Themenontologien. Dieses Vorgehen hilft andererseits dabei, einfache Abbildungsregeln zwischen den GML-Applikationsschemata und den INSPIRE-Themenontologien zu formulieren. Als Teil des Lösungsansatzes werden OWL-Annotationen neudefiniert, mit deren Hilfe die Abbildungsregeln direkt innerhalb der Themenontologien gespeichert werden können und somit keine externen Transformationsskripte entstehen. Die OWL-Annotationen dienen dazu, OWL-Klassen und -Prädikate der Themenontologien mit Informationen zu ihren korrespondierenden GML-Objekttypen bzw. GML-Properties zu verknüpfen. Die Syntax der Annotationen beschränkt sich dabei auf qualifizierende XML-Elementnamen zur Referenzierung von GML-Objekttypen bzw. auf XPath-Ausdrücke für GML-Properties (siehe Abschnitt 5.1.5 *Abbildungsregeln auf INSPIRE GML-Applikationsschemata*).

Die wenigen, oben angedeuteten Ausnahmen der Modellüberführung sind eigens dazu gedacht, eine **einfache Suche** und Formulierung von SPARQL-Anfragen zuzulassen und dabei alle Anfragefälle aus Anforderung FA2 zu unterstützen (siehe Abschnitt 3.2). Die Modellierungsaspekte der INSPIRE-Themenontologien (siehe Abschnitt 5.1) sind auf ebendieses Ziel der einfachen Benutzeranfragen und Publizierung von INSPIRE-Datensätzen ausgerichtet und weniger auf erkenntnistheoretische Betrachtungen von Konzeptzusammenhängen. Deshalb eignen sich die Themenontologien mit den enthaltenen GML/OWL-Abbildungsregeln auch gleichzeitg zur **Konfiguration einer SPARQL-Anfrageschnittstelle**, mit der sich der nächste Abschnitt 4.2 befasst.

Ein weiteres Verwendungsziel ist die **Harmonisierung von Referenzinformationen**, wie z.B. Koordinatenreferenzsysteme und Maßeinheiten, deren INSPIRE-Bezeichner zwischen URIs, URNs und einfachen Abkürzungen variieren können. Anstelle dieser für die Interopabiliät unzweckmäßigen Variabilität werden einheitliche, auflösbare URIs zur Benennung von Referenzinformationen vorgeschlagen und deren praktikable Speicherung als Semantic Web-Ressourcen untersucht. Ein letzter, nicht zu unterschätzender Aspekt ist das **Ontologie-Alignment**, d.h. das Verknüpfen mit externen Ontologiekonzepten. Ähnlichkeiten zwischen Ontologien und ihren konkreten Instanzdaten werden hierdurch transparent und können z.B. zur Rechercheoptimierung oder Inferenzbildung herangezogen werden.

4.2 Architektur für geosemantische Anfragen

Wie die Einleitung in Abschnitt 1.3 erläutert, konzentriert sich die vorliegende Arbeit auf fachliche INSPIRE-Daten und schließt die Behandlung von INSPIRE-Metadaten aus. Während die semantische Interoperabilität mit Hilfe der Datenspezifikationen gewährleistet wird, stützt sich INSPIRE gemäß dem OGC-Ansatz auf eine syntaktische Interoperabilität über Diensteschnittstellen. INSPIRE adressiert den OGC WFS in der Version 2.0 als INSPIRE-Downloaddienst bzw. genauer als *Direct Access Download Service* (siehe Kapitel 2.5.3). Ein alternativer *Download Service for predefined data sets/ pre-defined parts of data sets* ist von untergeordneter Bedeutung und wird in dieser Arbeit nicht berücksichtigt, da dieser Diensttyp nur dafür vorgesehen ist, fertig zusammengestellte Geodatensätze zum Download anzubieten, nicht jedoch die höherwertige Eigenschaft der Datenfilterung während des Downloads gestattet. Das Lösungskonzept geht deshalb vom OGC WFS bzw. einem INSPIRE Direct Access Download Service als **Quellsystem einer geosemantischen Anfragearchitektur** aus. Betrachtet man die verschiedenen WFS-Operationen hinsichtlich ihrer Tauglichkeit im INSPIRE-Umfeld, so fällt auf, dass die *DescribeFeatureType*-Operation im Grunde überflüssig ist, weil die INSPIRE-Datenmodelle bekannt sind und nicht nötigerweise zur Laufzeit erfragt und analysiert werden müssen. Alle anderen WFS-Operationen erfüllen ihren Zweck. Eine zentrale Rolle kommt der *GetFeature*-Operation für alle

[82] Measurement Units Ontology; siehe: http://forge.morfeo-project.org/wiki_en/index.php/Units_of_measurement_ontology

Suchanfragen und Exportvorgänge zu. Um die exportierte Datenmenge der GetFeature-Operation zu komprimieren, können Elementinformationen ausgelagert, d.h. nur per Verlinkung (Stichwort *by reference*), eingebettet werden. Die Verlinkungen in Form von *GetGmlObject*-Requests (WFS 2.0: *GetPropertyValue*) dienen einem nachträglichen Hinzuladen jener Detailinformationen.

Es sollen nun drei **Systemkonstellationen** verglichen werden, die für eine Semantic Web-Architektur mit INSPIRE-Downloaddiensten in Betracht kommen. Alle drei Systemkonstellationen erfüllen die beiden Grundvoraussetzungen, indem sie erstens SPARQL-Anfragen auf INSPIRE-Datensätze auswerten und zweitens die originäre INSPIRE-Datenhaltung (GML) in eine Semantic Web-Form (RDF/OWL, RDF-Triplestore) überführen. Die drei möglichen Systemkonstellationen sind:

- Option A: ein **separates Semantic Web-Repository**, in das zyklisch aktuelle Daten überführt werden
- Option B: ein **Semantic Web WFS-Profil** für SPARQL-Anfragen und RDF/OWL-Ausgaben
- Option C: ein **Semantic Web-Proxy** als Fassade vor WFS-Diensten

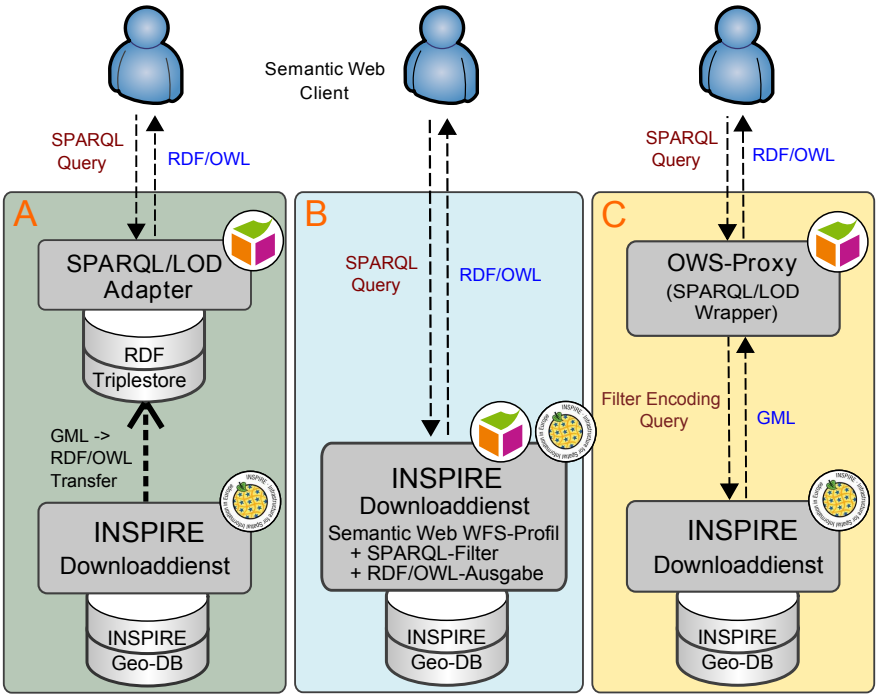


Abbildung 4.2 – Vergleich potentieller Systemarchitekturen

Option A

Die Systemarchitektur A (siehe Diagramm 4.2) sieht den vollständigen WFS-Datenexport in ein **separates Semantic Web-Repository** (z.B. RDF-Triplestore) vor, um gegen Letzteres semantische Anfragen richten zu können. Damit lassen sich einerseits Latenzzeiten für Transformationsprozesse während der Suchanfragen (GML → RDF/OWL) vermeiden. Andererseits befinden sich die INSPIRE-Daten nach der Überführung in einer Semantic Web-nativen Umgebung, so dass ein Administrator die Informationsfülle zwecks Nachnutzung im Semantic Web optimal erweitern kann, z.B. durch Hinzufügen von Verlinkungen und Annotierungen. Nachteilig ist jedoch die redundante Datenhaltung in der Geodatenbank und im RDF-Triplestore, die zur Vermeidung von Inkonsistenzen Synchronisierungsvorgänge des RDF-Triplestore erfordert. Desweiteren wird für den Betrieb des RDF-Triplestore ein leistungsstarkes System benötigt,

weil die Datenmenge im INSPIRE-Regelbetrieb aller Voraussicht nach enorme Dimensionen annehmen wird.[83] Es sind also die Prozessorleistung, die Speicherausstattung und andere Leistungskennwerte darauf auszulegen.

Option B

Systemarchitektur B erweitert den WFS-Dienst im Sinne einer **WFS-Profilierung** um spezielle Semantic Web-Schnittstellen für SPARQL-Anfragen bzw. RDF/OWL-Formatausgaben. Hierdurch lässt sich eine vollständige Dezentralisierung der Rechenleistung erzielen, da die Anfrageverarbeitung direkt bei der Informationsquelle stattfindet. Allerdings ist es in dieser Systemkonstellation nicht möglich, Themen- und Datensatz-übergreifende Ergebnisaufbereitungen und Datenverschneidungen durchzuführen. Ebensowenig lässt sich darauf bauen, dass nach der OGC-Spezifizierung eines derartigen WFS-Profils auch die nötigen Implementierungen folgen werden und die Datenbereitsteller gewillt sind, den freiwilligen Mehraufwand zu leisten, um die eventuell verfügbaren Implementierungen einzusetzen und zu warten.

Option C

Systemarchitektur C stützt sich auf eine Proxy-Komponente - im Weiteren auch **OWS-Proxy** genannt, die die WFS-Dienste zum Semantic Web kapselt und die INSPIRE GML-Daten mittels eines *virtuellen Repository* ins Semantic Web überführt. Das Repository ist virtuell, da zu jeder Suchanfrage INSPIRE-Daten in RDF/OWL-Ressourcen umgeformt werden, die somit stets einen temporären, flüchtigen Charakter haben. Dieser Ansatz bietet den Vorteil, fortwährend auf aktuellen Daten zu operieren, indem stets die originären Datenquellen abgerufen werden. Außerdem kann die Rechenleistung auf viele WFS-Dienste verteilt werden, indem der Proxy parallele Filteranfragen in die Quelldienste delegiert. Je nach Bedarf sind verschiedene Einsatzszenarien denkbar, die darüber entscheiden, wo und wieviele Proxy-Komponenten sinnvollerweise zu betreiben sind, ob thematisch zusammengehörige und wieviele WFS-Dienste jeweils angebunden werden. Von Nachteil ist die rechenintensive Transformation (GML $\rightarrow$ RDF/OWL) zur Laufzeit. Es gilt auch zu bedenken, dass Benutzeranfragen zu allgemein formuliert sein könnten. Werden beispielsweise Anfragen ohne Filtereinschränkung gestellt, könnten die Anfragen zu viele WFS-Dienste adressieren und in einer nicht zu bewältigenden Ergebnismenge münden, so dass grundsätzlich auch Abbruchbedingungen für Suchvorgänge definiert werden sollten.

Abwägung und Entscheidungsfindung

Um die beste Systemarchitektur zu identifizieren, sind die Kernaspekte der vorgestellten Lösungen nun in einem direkten Vergleich zu diskutieren. In den Lösungen A und C ist eine Konfiguration nur an zentraler Stelle vorzunehmen. Im Gegensatz zu Option B treten auch keine Mehraufwände seitens der INSPIRE-Datenbereitsteller auf. Obendrein ist fraglich, ob eine flächendeckende Einführung eines noch zu spezifizierenden WFS Semantic Web-Profils (Option B) überhaupt realisierbar ist. Aufgrund der vielen Fragezeichen und Umsetzungsschwierigkeiten kommt Option B nicht in die engere Auswahl.

Die beiden verbleibenden Lösungen A und C unterscheiden sich primär in der Art der zugrunde liegenden Datenhaltung, entweder als statisches (A) oder virtuelles Semantic Web-Repository (C). Ein virtuelles Repository geht einher mit Performanzeinbußen aufgrund der anfallenden aufwendigen Datenübertragungs- und Transformationsprozesse. Ebenso ist von Nachteil, Benutzeranfragen zwingend mit Filtereinschränkungen stellen zu müssen, um dank einer Vorauswahl die Zahl der anzufragenden Downloaddienste sowie die resultierende Ergebnisdatenmenge gering zu halten. Mit Hilfe eines statischen Semantic Web-Repository, das Speicheroptimierungen und das Einrichten von Datenbank-Indizes erlaubt, lassen sich Suchvorgänge erheblich beschleunigen.

[83] berücksichtigt man die 32 datenbezogenen INSPIRE-Themen und je hunderte Datenprodukte der Mitgliedsstaaten, kann die Gesamtdatenmenge eine Größenordnung von mehreren Gigabyte im zwei- oder dreistelligen Bereich annehmen.

Bei all den genannten positiven Aspekten hat das statischen Repository jedoch das große Manko, dass eine redundante Datenhaltung aufgebaut und aktualisiert werden muss. Im Gegensatz zur Lösung C (OWS-Proxy) ergeben sich daraus höhere Serveranforderungen. Zusätzlich sind regelmäßige Aktualisierungszyklen durchzuführen, um die replizierten Daten konsistent zu den Primärdaten zu halten. Die nötigen Wartungs- und Aktualisierungsvorgänge verhindern einen leichtgewichtigen Ansatz, der sich mit einem OWS-Proxy ohne Datenhaltungsschicht und flexibleren Einsatzmöglichkeiten bietet. **Der Lösungsansatz dieser Arbeit ist** deshalb die **Systemarchitektur C**. Einen detaillierten Einblick in die gewählte Architektur liefert Abschnitt 5.2, dort werden die einzelnen Funktionalitäten und Prozessabläufe vorgestellt.

An dieser Stelle soll darauf hingewiesen werden, dass die gewählte Systemarchitektur zwar die Aufgabe hat, eine Fassade zu mehreren INSPIRE-Datenbeständen zu bilden, um diese innerhalb einer verteilten Suche (Federation) virtuell zu einer großen zusammenhängenden Linked Data-Datenquelle zu vereinen. Die Arbeit strebt jedoch keine Systemlösung an, die eine **Federation über diverse Linked Data-Quellen** in der Linked Data-Cloud durchführt, so z.B. auch Geonames und LinkedGeoData einbindet. Letztere Aufgabenstellung ist nicht INSPIRE-spezifisch, sondern betrifft ganz allgemein den Datentransfer im Semantic Web. Es handelt sich dabei um ein sehr aktives Forschungsthema, was sich anhand diverser noch unfertiger Software-Lösungen (sogenannte *Federator*-Applikationen) und insbesondere der jüngst ausgearbeiteten Spracherweiterung von SPARQL verdeutlicht, die sogenannte *Federated Query-Extension*[84]. Um den Entwicklungen nicht vorwegzugreifen und den Fokus auf das Semantic Enablement von INSPIRE zu lenken, soll die Thematik nur am Rande thematisiert werden, so z.B. in den Testszenarien in Abschnitt 5.3.4 *Testergebnisse*.

[84] Spezifikation bzw. *Working Draft* vom November 2011: http://www.w3.org/TR/sparql11-federated-query/

5 Konzeption und Implementierung eines Prototypen

Die Abschnitte des Hauptkapitels folgen logisch der Gliederung des Lösungsansatzes und vertiefen die dort skizzierten Konzepte. Der erste Abschnitt 5.1 beschäftigt sich mit der Modellierung von INSPIRE-Themenontologien, während der zweite Abschnitt 5.2 die Programmlogik der gewählten Proxy-Architektur konkretisiert. Zuletzt folgt die Beschreibung zur prototypischen Umsetzung der Konzepte und ein Testbericht zu durchgeführten Testfällen mit repräsentativen INSPIRE-Datensätzen (Abschnitt 5.3).

5.1 Ontologie-Modellierung

Der Themenkomplex Ontologie-Modellierung beschäftigt sich nun eingehend mit der Definition der INSPIRE-Themenontologien, welche selbstverständlich geprägt sind durch die Quellinformationen aus den INSPIRE-Modellen. Dabei liefern die INSPIRE-Modellarten UML und GML komplementäre Informationen für verschiedene Verwendungen, die auch die Einteilung dieses Abschnitts vorgeben sollen. Die UML-Modelle eignen sich zur Ableitung von Ontologiekonzepten. Sie sind demnach Ausgangspunkt für die Vokabularentwicklung (siehe Abschnitt 5.1.1) und die Vokabularverfeinerung in Bezug auf a) die Identifizierung von Konzepten (Abschnitt 5.1.2), b) deren Bezüge zu Referenzinformationen (Abschnitt 5.1.3) und c) der Speicherung mit geographischer Information (Abschnitt 5.1.4). Die GML-Modelle bzw. die dafür vorgesehenen INSPIRE UML/GML-Abbildungsregeln geben Auskunft über die genaue Speicherstruktur der GML-Instanzdaten, die bekannt sein muss, um GML in RDF/OWL zu transformieren. Aus den Strukturinformationen werden deshalb GML/OWL-Abbildungsregeln hergeleitet, deren Gesetzmäßigkeiten und Speicherformen in Abschnitt 5.1.5 dargestellt werden. Der Abschnitt 5.1.6 verschafft zu guter Letzt einen Überblick über die in der Arbeit angestrebte architektonische Vernetzung von INSPIRE-Themenontologien und ihre Bezüge zu externen Ontologien.

5.1.1 Vokabularentwicklung aus INSPIRE-Datenmodellen

Gemäß dem Lösungsansatz (Abschnitt 4.1) bilden die INSPIRE UML-Modelle die Quellinformation, um daraus OWL-Konzepte der Themenontologien abzuleiten. Die Sprachen UML und OWL entstammen unterschiedlichen Systemwelten. Die UML ihrerseits ist die primäre Sprache für den Ansatz der *Model-getriebenen Architektur* in der Software-Entwicklung (*Model driven architecture*, MDA). Mit Hilfe der UML werden Software-Komponenten und -Schnittstellen strukturell entworfen und Programmabläufe, aufgeteilt in Workflows, Aktivitäten und Aktionen, modelliert und dokumentiert. Einen Teilbereich der UML machen die omnipräsenten UML-Klassendiagramme aus. Sie eignen sich auch zur Datenmodellierung, die in der Software-Entwicklung zumeist aber auf die Erstellung von Datenbankschemata bezogen ist. Hingegen wird die OWL vornehmlich zur Datenmodellierung eingesetzt und bettet sich ein in die Forschungsbereiche um semantische Techniken und künstliche Intelligenz. Eine Umformung von UML-Modellen nach OWL-Ontologien und umgekehrt bewegt sich demzufolge zwangsläufig auf dem Feld der Datenmodellierung, sei es um OWL-Ontologien mit UML-Klassendiagrammen leichter zu veranschaulichen oder Software-Entwicklern in ihrer gewohnten UML-Umgebung eine teilautomatisierte Generierung von OWL-Ontologien zu gestattet, um Semantic Web-Techniken auf ihre Datenhaltung anwenden zu können.

Ein umfassender Vergleich beider Sprachen, der die prinzipiellen Sprachpotentiale und -ähnlichkeiten sowie die konkreten Übersetzungsmöglichkeiten herausarbeitet, liefern Kiko & Atkinson [2008]. Kiko & Atkinson betrachten zusätzlich zu UML-Standardelementen die UML ergänzende Sprache *Object Constraint Language* (OCL[85]) zur Einschränkung von UML-Notationselementen, so dass die vielen OWL-Restriktionen mit OCL-Entsprechungen verglichen werden können. Die wertvolle Gegenüberstellung dient im Weiteren als eine Grundlage zur Bildung der Themenontologien. Abgesehen davon beschäftigen sich diverse Projektarbeiten und Veröffentlichungen mit der technischen Überbrückung der Kluft beider System- und Sprachwelten. Beispielsweise definieren Djuric et al. [2004] das *Ontology Definition Metamodel* (ODM) in Verbindung mit einem UML-Profil, dem *Ontology UML-Profil*. Das UML-Profil beinhaltet neue UML-Stereotypen und -Tagged values, die das Markieren und Gleichsetzen von UML-Elementen mit OWL-Sprachkonstrukten in üblichen UML-Modellierungswerkzeugen erlauben. Eine Wiederverwendung dieser Profil-Stereotypen und -Tagged values ist im Kontext von INSPIRE jedoch nur bedingt möglich, weil INSPIRE eigene Stereotypen und Tagged values anwendet, die mit jenen des Ontology UML-Profils konkurrieren würden. Ebenso sollten dem Verständnis des Autors nach die UML-Modelle zur sauberen Trennung zwischen Quell- (UML) und Zielmodell (OWL) nicht mit Abbildungsanweisungen angereichert werden, um ein Auseinanderdriften von Entwicklungspfaden im Falle zukünftiger Aktualisierungen der INSPIRE-Datenmodelle zu vermeiden. Desweiteren stellen Djuric et al. [2004] einige Randbedingungen an die UML-Modellierung, z.B. die Ausgestaltung von Assoziationen durch explizite Assoziationsklassen (mit Stereotyp `Property`) anstelle der üblichen Verlinkungsnotation. Die Randbedingungen haben zur Folge, dass die INSPIRE UML-Modelle zur Anwendung des Ontology UML-Profils erheblich umstrukturiert werden müssten, wovon - wie bereits geschildert - abgesehen wird.

Allgemeine Abbildungsregeln

Anstelle der bloßen Adaption eines existierenden technischen Lösungsweges, der die Besonderheiten der INSPIRE-Datenmodelle unberücksichtigt lässt, werden auf den nächsten Seiten einfache und INSPIRE-spezifische Abbildungsregeln vorgeschlagen. Zum einen orientieren sie sich an den INSPIRE-Stereotypen, die wertvolle Indizien zur Semantik liefern. Zusätzlich ist auf die syntaktische Ähnlichkeit zwischen der GML und RDF bzw. OWL hinzuweisen, die sich in der GML mit dem *Object-Property-Muster* und Objektverlinkungen *by reference* äußert (siehe Abschnitt 2.3.2, Absatz *Geography Markup Language - Kurzporträt*). Das Objekt-Property-Muster ist wegen der direkten Verwandtschaft der INSPIRE UML- und GML-Modelle implizit auch in den UML-Modellen enthalten, wodurch kongruente Sprachkonstrukte in allen drei hier relevanten Sprachfamilien UML, GML und OWL vorkommen und in Tabelle 5.1 gegenübergestellt werden.

Tabelle 5.1 – Kongruente Sprachkonstrukte der Sprachfamilien UML, GML und OWL

Elementtyp	**UML**	**GML**	**OWL**
Objektklasse	Klasse	Objekttyp	Klasse
Relation bzw. Objektverlinkung	Assoziation/ Klassenattribut	Property	Prädikat

Ausgehend von den UML/OWL-Entsprechungen, die in der Tabelle 5.1 aufgezeigt werden, gibt die Tabelle 5.2 den Rahmen für die UML/OWL-Modellabbildung vor. Es sind hier zunächst nur die wichtigsten Abbildungsregeln aufgeführt, die nachfolgend in einen Gesamtzusammenhang gestellt werden (siehe Absatz *Design-Entscheidungen* im gleichen Abschnitt) und weiter verfeinert werden (siehe Absatz *Abbildung von INSPIRE-gebräuchlichen ISO-Datentypen*).

[85] Spezifikation: http://www.omg.org/spec/OCL/2.3.1

Tabelle 5.2 – Modellabbildung von UML nach OWL
(siehe Absatz *Design-Entscheidungen* hinsichtlich der Spezifika: *a*: Namensgebung, *b*: Elementeindeutigkeit und *c*: Stereotypen `voidable`, `lifeCycleInfo` und `version`)

INSPIRE-Datenmodell UML-Notation	**INSPIRE-Themenontologie** Abbildung auf OWL-Konstrukte
UML-Klassen und -Datentypen	
UML-Klassen mit Stereotyp `featureType`, `dataType` oder `type`	• je eine neue OWL-Klasse (owl:Class) [a] • alle zugehörigen UML-Klassenattribute/-Assoziationen werden auf je ein Prädikat abgebildet (siehe nachfolgende Regeln) • für jedes Prädikat wird eine qualifizierende OWL-Klassenrestriktion eingeführt, die die von der OWL-Klasse über das jeweilige Prädikat referenzierten Datentypen und Objekttypen einschränkt (owl:allValuesFrom) [b] • Optionalität und Multiplizität von UML-Klassenattributen/-Assoziationen bleiben unberücksichtigt [b]
UML-Klassen mit Stereotyp `union`	• je eine neue OWL-Klasse (owl:Class) [a] • jede Subklasse eines UML-Union wird auf je eine OWL-Klasse abgebildet • die neu gebildete OWL-Klasse referenziert auf ihre Subklassen (owl:unionOf), wodurch die OWL-Instanzen obligativ eine der Subklassen angehören müssen
UML-Klassen mit Stereotyp `placeholder`	• keine Berücksichtigung, Modellierung erfolgt im Kontext anderer INSPIRE-Datenmodelle
UML-Datentyp mit Stereotyp `codelist` oder `enumeration`	• je eine neue OWL-Klasse (owl:Class) [a] • die Einträge der jeweiligen Werteliste werden als OWL-Instanzen (owl:NamedIndividual) der neu gebildeten OWL-Klasse typisiert (rdf:type) • die OWL-Klasse wird auf die Menge dieser OWL-Instanzen eingeschränkt (owl:oneOf)
UML-Klassenattribute	
UML-Klassenattribute mit einfachen Datentypen	• je ein neues Prädikat zur Attributierung einer OWL-Instanz mit einem rdf:Literal (owl:DatatypeProperty) [a] • das Prädikat wird nicht als *funktional* deklariert (owl:FunctionalProperty) [b] • keine Deklaration von Definitions-/ Wertebereich (rdfs:domain, rdfs:range) [b]
UML-Klassenattribute, typisiert mit UML-Klassen	• je ein neues Prädikat zur Verknüpfung zweier OWL-Instanzen (owl:objectProperty) [a] • das Prädikat wird nicht als *funktional* deklariert (owl:FunctionalProperty) [b] • keine Deklaration von Definitions-/ Wertebereich (rdfs:domain, rdfs:range) [b]
UML-Klassenattribute mit Stereotyp `voidable`	• zu behandeln wie gewöhnliche UML-Klassenattribute [c]
UML-Klassenattribute mit Stereotyp `lifeCycleInfo`	• zu behandeln wie gewöhnliche UML-Klassenattribute [c]
UML-Assoziationen	
Vererbungsassoziationen zwischen UML-Klassen	• je eine neue Vererbungsbeziehung zwischen korrespondierenden OWL-Klassen (rdfs:subClassOf) [a]
unidirektionale UML-Assoziationen	• je ein neues Prädikat zur Verknüpfung zweier OWL-Instanzen [a] (owl:objectProperty) • das Prädikat wird nicht als *funktional* deklariert (owl:FunctionalProperty) [b] • keine Deklaration von Definitions-/ Wertebereich (rdfs:domain, rdfs:range) [b]
bidirektionale UML-Assoziationen	• je zwei neue Prädikate für Relationen zwischen OWL-Instanzen [a] (owl:objectProperty) • die Prädikate werden nicht als *invers* oder *funktional* deklariert (owl:inverseOf, owl:FunctionalProperty) [b] • keine Deklaration von Definitions-/ Wertebereich (rdfs:domain, rdfs:range) [b]
UML-Assoziationsrollen mit Stereotyp `voidable`	• zu behandeln wie gewöhnliche UML-Assoziationen [c]
UML-Assoziationsrollen mit Stereotyp `version`	• zu behandeln wie gewöhnliche UML-Assoziationen [c]

Design-Entscheidungen

Von besonderer Bedeutung ist die **Namensgebung** der abgeleiteten OWL-Konzepte, da mit den Namen bereits Semantik verbunden und der Umgang mit Konzepten, die selbsterklärende Namen tragen, erheblich erleichtert wird. Die INSPIRE-Elementbezeichner sind leicht verständlich und ihrer Definition nach sinngemäß benannt. Eine direkte Übernahme und Benennung von OWL-Konzepten nach INSPIRE-Elementen ist zu empfehlen. Erstens aufgrund der einfacheren Modellabbildung von UML auf OWL, indem manuelle Modifikationsschritte zur Vergabe von Konzeptnamen unterbleiben. Zweitens aber auch wegen der Aussicht, die UML-Modelle als Orientierungsshilfe und Dokumentationsmittel der Themenontologien zu verwenden. Eine sinnvolle Ausnahme, die eine namentliche Abweichung von den INSPIRE-Bezeichnern rechtfertigt, wird weiter unten erläutert.

Zunächst aber soll auf eine weitreichende Modellierungsproblematik hingewiesen werden, die bei der Überführung von **UML-Klassenattributen und -Assoziationen** in OWL-Prädikate auftritt. Innerhalb eines UML-Diagramms bzw. -Packages besitzen UML-Klassenattribute und -Assoziationen keine universell eindeutigen Bezeichner, sondern sind abhängig von der UML-Klasse, in dessen Kontext sie definiert wurden. Im Gegensatz dazu sind OWL-Prädikate stets eindeutig in ihrem ontologischen Namensraum benannt. Zwei oder mehr gleichnamige UML-Klassenattribute bzw. -Assoziationen werden demzufolge unter Beibehaltung von INSPIRE-Namen auf ein einziges OWL-Prädikat abgebildet, d.h. es liegt eine n:1 Abbildung vor. Ein Beispiel dazu: beide UML-Klassen `ps-f:ProtectedSite` und `ps-f:SiteIdentifierType` enthalten je ein Klassenattribut namens `ps-f:siteIdentifier` (siehe Abbildung 2.9). Daraus resultiert ein OWL-Prädikat namens `ont_ps:siteIdentifier`. Die ungleichmäßige Abbildung von UML auf OWL hat mehrere Konsequenzen, die folgenreichsten sind:

1. der Verlust der für die Datentransformation relevanten UML/GML-Struktur
2. die Übertragung unterschiedlicher Semantiken auf ein OWL-Prädikat
3. die namentliche Differenzierung zwischen OWL-ObjectProperty und -DatatypeProperty

Erstens geht die UML-Struktur (implizit auch die GML-Struktur) bei der Übertragung nach OWL verloren, so dass die Bildung von Regeln zur Datentransformation von GML- nach OWL-Instanzdaten erschwert wird. Einen Lösungsvorschlag zu dieser Problematik liefert Abschnitt 5.1.5. Zweitens kann es passieren, dass gleichnamige und aus unterschiedlichen Klassen abstammende Klassenattribute bzw. -Assoziationen unterschiedliche Semantiken aufweisen und das abgeleitete OWL-Prädikat verschiedene oder sogar widersprüchliche Bedeutungen erhält. Zwar kann eher ausgeschlossen werden, dass gleichnamige INSPIRE UML-Klassenattribute bzw. -Assoziationen gänzlich widersprüchliche Bedeutungen tragen, es sei denn im Falle von Homonymen. Schwerwiegender lastet aber das Problem, dass ein resultierendes OWL-Prädikat nicht mit ontologischen Restriktionen belegt werden kann, weil es unter Umständen verschiedene UML-Klassenattribute bzw. -Assoziationen repräsentiert, die in unterschiedlichen Klassenzusammenhängen und Objektbeziehungen stehen. Die mangelnde **Elementeindeutigkeit** in der UML-Welt überträgt sich somit auf die OWL-Ontologie, so dass nur vereinzelt Restriktionen zur Anwendung kommen können, u.a. betrifft dies die Festlegung des Definitions- und Wertebereichs von OWL-Prädikaten (per Definitionsprädikate `rdfs:domain` und `rdfs:range`), die Deklaration von inversen Prädikaten (entgegengesetzte Prädikate einer bidirektionalen Objektverlinkung; deklariert mit `owl:inversOf`) oder das Attestieren der Funktionalität von Prädikaten (beziehen sich auf Objekte derselben Identität; typisiert als `owl:FunctionalProperty`). Die Themenontologien sind deshalb offener gestaltet und unbestimmter in der Verwendung der OWL-Prädikate, wodurch ontologische Reasoner weniger Vorgaben und Hilfsanweisungen erhalten, um Instanzdaten zu validieren und semantische Schlussfolgerungen anzustellen. Die offene Definition der OWL-Prädikate kann geringfügig eingeschränkt und kompensiert werden, indem die neugebildeten OWL-Klassen **qualifizierende Klassenrestriktionen** erhalten, die den Gebrauch und den Objekttyp von OWL-Prädikaten im Kontext der OWL-Klasse vorschreiben (Definitionsprädikat `owl:allValuesFrom`).

Der dritte Problemfall tritt dann zutage, wenn sich die Datentypen gleichnamiger INSPIRE UML-Klassenattribute bzw. -Assoziationen grundlegend unterscheiden und der Wertebereich eines resultierenden OWL-Prädikats sowohl

a) OWL-Klassen als auch b) einfache Datentypen einbezieht. In der OWL-Teilsprache DL (siehe Abschnitt 2.2.2) ist ein OWL-Prädikat in `owl:ObjectProperty` (Fall a) oder `owl:DatatypeProperty` (Fall b) zu differenzieren, die Definition eines neutralen Prädikats, das beide Fälle abdeckt, ist aus Gründen der **Entscheidbarkeit in OWL-DL** nicht erlaubt. Die Entscheidbarkeit einer Ontologie ist allerdings ein wichtiges Kriterium zum Einsatz von Reasonern und anderen semantischen Werkzeugen, die größtenteils OWL-DL konform entwickelt werden. Es ist deshalb ratsam, die INSPIRE-Themenontologien generell konform zu OWL-DL zu gestalten und im geschilderten Problemfall gleichzeitig eine `owl:ObjectProperty` und eine `owl:DatatypeProperty` einzuführen, deren Namen voneinander abweichen müssen. In der Arbeit von Bohring & Auer [2005], die die Transformation von XML-Schema nach OWL behandelt, wird die gleiche Problemstellung auf eine systematische Weise gelöst. Und zwar erhält jede `owl:ObjectProperty` im Namen den Präfix `has` und jede `owl:DatatypeProperty` den Präfix `dtp`, z.B. `ont_ps:hasSiteIdentifier` und `ont_ps:dtpSiteIdentifier`. Anstelle der von Bohring & Auer angedachten Präfixe `has` und `dtp` wird hier zu einer alternativen Lösung tendiert, die weniger konsequent, jedoch sicherlich praktikabler im Gebrauch ist. In den wenigen Ausnahmefällen, in denen gemäß OWL-DL zwei Prädikate zu unterscheiden sind, soll nur dem Namen der jeweiligen `owl:DatatypeProperty` das Suffix `_DataValue` angefügt werden (z.B. `ont_ps:siteIdentifier_DataValue`). Die Vorgehensweise vermeidet große namentliche Abweichungen der OWL-Konzeptnamen zu den INSPIRE-Bezeichnern aus den UML-Modellen, wodurch die genannten Vorteile bei der Übernahme der INSPIRE-Bezeichner weitestgehend erhalten bleiben.

Die Abbildung von **UML-Klassen** auf OWL-Klassen ist hingegen mühelos zu realisieren, denn jede UML-Klasse ist eindeutig innerhalb eines UML-Packages. Ebenso differenziert die OWL nicht in verschiedene Klassenkategorien, weshalb aus jeder UML- eine OWL-Klasse folgt. Namenskonflikte mit OWL-Prädikaten können nicht entstehen, denn Konzept-Namen bzw. -URIs sind prinzipiell case-sensitiv, so dass spätestens die Groß-/Kleinschreibweise der INSPIRE UML-KLassen (stets mit großem Anfangsbuchstaben geschrieben) bzw. UML-Attribute und -Assoziationen (stets mit kleinem Anfangsbuchstaben) zu einer namentlichen Differenzierung zwischen OWL-Klassen und OLW-Prädikaten führt. Möglicherweise existieren gleichnamige UML-Klassen in verschiedenen UML-Packages eines einzigen INSPIRE Annex-Themas. In solchen Fällen resultiert wiederum nur eine OWL-Klasse, der gegebenenfalls unterschiedliche Bedeutungen beigemessen werden. Das diese Vorgehensweise aber auch vorteilhaft sein kann, beweist das Beispiel der namensgleichen UML-Klasse `ProtectedSite` aus den UML-Packages bzw. Applikationsschemata `Protected Sites Simple` und `Protected Sites Full`. Die verantwortliche INSPIRE-TWG teilt ein und dasselbe Konzept `ProtectedSite` in eine *abgespeckte* und verpflichtende Simple-Variante und eine üppiger attributierte, dafür aber optionale Full-Variante auf. Die datenstrukturierende Maßnahme ist dem Konzeptdenken des Semantic Web fremd. Mit der Abbildung auf nur eine OWL-Klasse wird also eine Konsolidierung herbeigeführt.

Die **INSPIRE-Stereotypen voidable, lifeCycleInfo und version** (siehe Abschnitt 2.5.2) haben keine besondere Relevanz für INSPIRE Semantic Web-Ressourcen. Zum einen markiert der Stereotyp `voidable` Elemente, zu denen ein Datenbereitsteller nicht obligatorisch Informationen liefern muss, die fehlende Information aber explizit und gegebenenfalls unter Angabe von Verhinderungsgründen bekanntzugeben hat. Das Semantic Web geht generell anders mit einem Informationsmangel um, indem erst gar keine Aussagen gebildet werden, sondern gemäß der Open World Assumption prinzipiell davon ausgegangen wird, dass externe Fakten die eigene Wissensbasis um fehlende Informationen vervollständigen könnten. Zum anderen beziehen sich INSPIRE-Datumsfelder, die mit dem Stereotyp `lifeCycleInfo` deklariert sind, auf die Gültigkeitsdauer von INSPIRE-Speicherobjekten. Es ist davon auszugehen, dass ein Anwender sich eher für Objektinformationen und weniger für Einzelheiten der Speicherung interessiert. Die Speicherdetails müssen deshalb nicht gesondert hervorgehoben werden, sondern sind bestenfalls wie gewöhnliche UML-Klassenattribute in OWL-Prädikate umzuwandeln. Desweiteren scheint der Stereotyp `version` vor dem Hintergrund einer Versionierung der INSPIRE-Datenmodelle eingeführt zu sein, jedoch wird der Stereotyp trotz bisheriger Versionsschritte nicht angewendet. Sollten sich Assoziationsrollen tatsächlich in späteren UML-Modellversionen ändern, so manifestiert sich eine solche Änderung üblicherweise auf höherer Ebene in der UML-Modellversion bzw. dem XML-Namensraum des GML-Applikationsschemas, weshalb die generelle Praktikabilität des Stereotyps `version` in Frage gestellt werden kann.

Folglich sollen die drei INSPIRE-Stereotypen `voidable`, `lifeCycleInfo` und `version` keine spezielle Berücksichtigung im Zuge der UML/OWL-Modellabbildung finden.

Abbildung von INSPIRE-gebräuchlichen ISO-Datentypen

Eine einfache Handhabe von INSPIRE-Themenontologien setzt voraus, dass trotz einer ausdrucksstarken Modellierung die Inhaltsspeicherung nicht zu komplex ausfällt. Dies gilt ebenso für die ontologischen RDF/OWL-Datentypen, weil sie als inhaltliche Hüllen korrekt interpretiert und in Suchfiltern eingesetzt werden müssen. Deshalb ist darauf zu achten, bei der Abbildung von INSPIRE UML-Datentypen umsichtig vorzugehen, gleichartige und verwandte Typisierungen zu erkennen und in wenige, aber aussagekräftige RDF/OWL-Datentypen umzuwandeln. Die UML-Datentypen geben den Wertebereich von UML-Klassenattributen vor und enthalten als primitive Datentypen entweder eine einzige Objektinformation, z.B. `CharacterString` oder `Boolean`, oder setzen sich als komplexe Datentypen aus mehreren Unterelementen zusammen, beispielsweise `CI_Citation` oder `GeographicalName`. Sie lassen sich darüber hinaus einteilen in Kategorien der numerischen Datentypen, der textuellen Datentypen, Datentypen zur Speicherung geometrischer Merkmale usw.

Gegenstand der folgenden Betrachtung sind jene UML-Datentypen, die nicht im Rahmen von INSPIRE neudefiniert, sondern aus ISO-Standards und -Modellen importiert wurden. INSPIRE macht regen Gebrauch von ISO-Datentypen. Darunter zählen vor allem die GML-Geometrietypen aus ISO 19136 (z.B. `GM_MultiSurface`), primitive Datentypen aus ISO 19103 (z.B. `DateTime`) und Metadatenelemente aus ISO 19115 (z.B. `CI_Citation`). Die Tabelle 5.3 gibt einen Überblick über die importierten ISO-Datentypen (linke Spalte), ihre Eingruppierung in Kategorien (siehe Absätze) und ihre relative Häufigkeit (mittlere Spalte). Die zugrunde liegende Analyse beschränkt sich auf die fertigen Annex-I-Themen, lässt aber bereits weitreichende Rückschlüsse auf den allgemeinen Gebrauch von Datentypen auch für Themen von Annex-II und -III zu. Die rechte Spalte gibt geeignete Abbildungen auf RDF/OWL-Datentypen vor, die im Folgenden erläutert werden.

Auffällig ist die häufige Verwendung von **Geometrietypen**. So trägt fast jedes zehnte UML-Klassenattribut dazu bei, dass INSPIRE-Modelle und -Feature Types ausgesprochen raumbezogen sind. Wie schon im Grundlagenkapitel in Abschnitt 2.4.3 anklang, ist GeoSPARQL prädestiniert für die Repräsentation geographischer Informationen. Demzufolge wird das GeoSPARQL Geo-Vokabular nachgenutzt, um ISO-Geometrietypen in eine ontologische Form zu überführen und unter der OWL-Klasse `geo:Geometry` zusammenzufassen. Alle OWL-Klassen der Themenontologien, die einer INSPIRE UML-Klasse des Stereotyps `featureType` entsprechen und mit einer räumlichen Ausdehnung attributiert sind, werden per `rdfs:subClassOf`-Prädikat von der GeoSPARQL OWL-Klasse `geo:Feature` abgeleitet. Eine weiterführende Diskussion zum Thema Geometriespeicherung mit GeoSPARQL findet in Abschnitt 5.1.4 statt.

Bezüglich der **Datentypen für Sachattributinformationen** haben Datumstypen und textuelle Datentypen eine ähnliche Häufigkeit, während numerische Datentypen und vordefinierte Wertelisten (Codelist-/ Enumerationstypen) seltener sind. Die Abbildung von ISO-Datentypen auf OWL-Konzepte und -Datentypen lässt sich in vier Abbildungskategorien aufteilen:

1. Primitive UML-Datentypen, zu denen vergleichbare OWL-Datentypen existieren
2. Komplexe UML-Datentypen, für die eine eigenständige OWL-Klasse gebildet wird
3. Sonderfall: vordefinierte Wertelisten (Codelist-/ Enumerationstypen)
4. Sonderfall: Messwerte (UML-Typ `Measure` und `Measure`-Spezialisierungen: z.B. `Length` und `Area`)

Die meisten **primitiven UML-Datentypen** der 1. Kategorie (z.B. `DateTime` und `Integer`) haben direkte Entsprechungen in der XML-Welt (z.B. `xsd:dateTime` und `xsd:integer`). XML-Schematypen werden sowohl von GML als auch von RDF/OWL genutzt, so dass aufgrund identischer lexikalischer Bereiche der primitiven Datentypen eine

direkte Inhaltstransformation von GML nach RDF/OWL stattfinden kann. Deshalb ist es ratsam, primitive UML-Datentypen auf XML-Schematypen abzubilden und dadurch für möglichst viele Gemeinsamkeiten zwischen GML-Applikationsschemata und RDF/OWL-Ontologien zu sorgen. Hingegen erfordern **komplexe UML-Datentypen** der 2. Kategorie (z.B. `CI_Citation`, `CI_ResponsibleParty` etc.) neue eigenständige OWL-Klassen. Die diesbezüglichen Modellierungsaspekte werden in Abschnitt 5.1.6 *Informationsvernetzung, Bezüge zu Basiskonzepten* genannt.

Tabelle 5.3 – Abbildung von INSPIRE ISO-Datentypen (* siehe besondere Erläuterungen im Begleittext)

INSPIRE ISO-Datentypen der Annex-I-Themen	%-Vorkommen	Mapping auf OWL-Klassen und -Datentypen
Geometrietypen		
GM_Primitive (abstrakte Root-Klasse)	< 5%	OWL-Klasse *geo:Geometry**
GM_Object (Root-Klasse)	< 1%	”
GM_Point	< 5%	”
GM_Curve	< 1%	”
GM_Surface	< 1%	”
GM_MultiSurface	< 5%	”
∑	**∼ 9%**	
Datumstypen		
DateTime	∼ 29%	xsd:dateTime
TM_OrdinalEra	< 1%	OWL-Klasse *base:TM_OrdinalEra**
TM_Period	< 1%	OWL-Klasse *base:TM_Period**
∑	**∼ 30%**	
Identifikatoren und textuelle Datentypen		
anyURI	< 1%	OWL-Ressource
Boolean	< 5%	xsd:boolean
CI_Citation	< 5%	OWL-Klasse *base:CI_Citation**
CI_ResponsibleParty	< 1%	OWL-Klasse *base:CI_ResponsibleParty**
CharacterString	∼ 15%	rdf:PlainLiteral
LocalisedCharacterString	< 5%	rdf:PlainLiteral (mit optionalem *language tag*)
∑	**∼ 27%**	
Numerische Datentypen		
Area	< 5%	OWL-Klasse *base:Measure**
Distance	< 1%	”
Integer	< 5%	xsd:integer
Length	< 5%	OWL-Klasse *base:Measure**
MD_Resolution	< 5%	OWL-Klasse *base:MD_Resolution**
Measure (Root-Klasse)	< 5%	OWL-Klasse *base:Measure**
Percentage	< 5%	xsd:float
Velocity	< 1%	OWL-Klasse *base:Measure**
∑	**∼ 16%**	
Vordefinierte Wertelisten		
Codelist-/ Enumerationstypen zusammengefasst	**∼ 18%**	jeweils eine individuelle OWL-Klasse*

Vordefinierte Wertelisten (Codelist-/ Enumerationstyp) eignen sich hervorragend zur Objektklassifizierung und -filterung. Die INSPIRE-Datenmodelle behergen viele und teils sehr reichhaltige Wertelisten, deren Optionswerte genauen Definitionen unterliegen, die in Definitionstexten und Beschreibungen wiedergegeben sind. Damit derartige Definitionen in die Themenontologien hineinwandern, werden Optionswerte nicht in RDF-Literalen, sondern in eigenständigen OWL-Individuen gespeichert (siehe auch den UML/OWL-Vergleich von Kiko & Atkinson [2008]). Dazu wird jeder Codelist-/ Enumerationstyp als OWL-Klasse abgebildet. Zugehörige Optionswerte werden als OWL-Individuen der jeweiligen OWL-Klasse typisiert und mit Definitionen und Beschreibungen aus den INSPIRE UML-Modellen annotiert. Da Enumerationen abgeschlossene Wertelisten darstellen, sind deren OWL-Klassen auf die zugehörigen Optionswerte über das Prädikat `owl:oneOf` einzuschränken. Die Optionswerte weisen über ihren Namen indirekt auf die Zugehörigkeit zu einer Werteliste hin, z.B. enthält die Werteliste `<http://ex.org/ProtectionClassificationValue>` einen Optionswert `<http://ex.org/ProtectionClassificationValue/archaeological>`. Aufgrund der gewählten Namensgebung können keine Namenskonflikte auftreten, sofern zwei Wertelisten gleichnamige Optionswerte enthalten sollten. Abbildung 5.1 zeigt die exemplarische OWL-Modellierung der Codelist `ProtectionClassificationValue`.

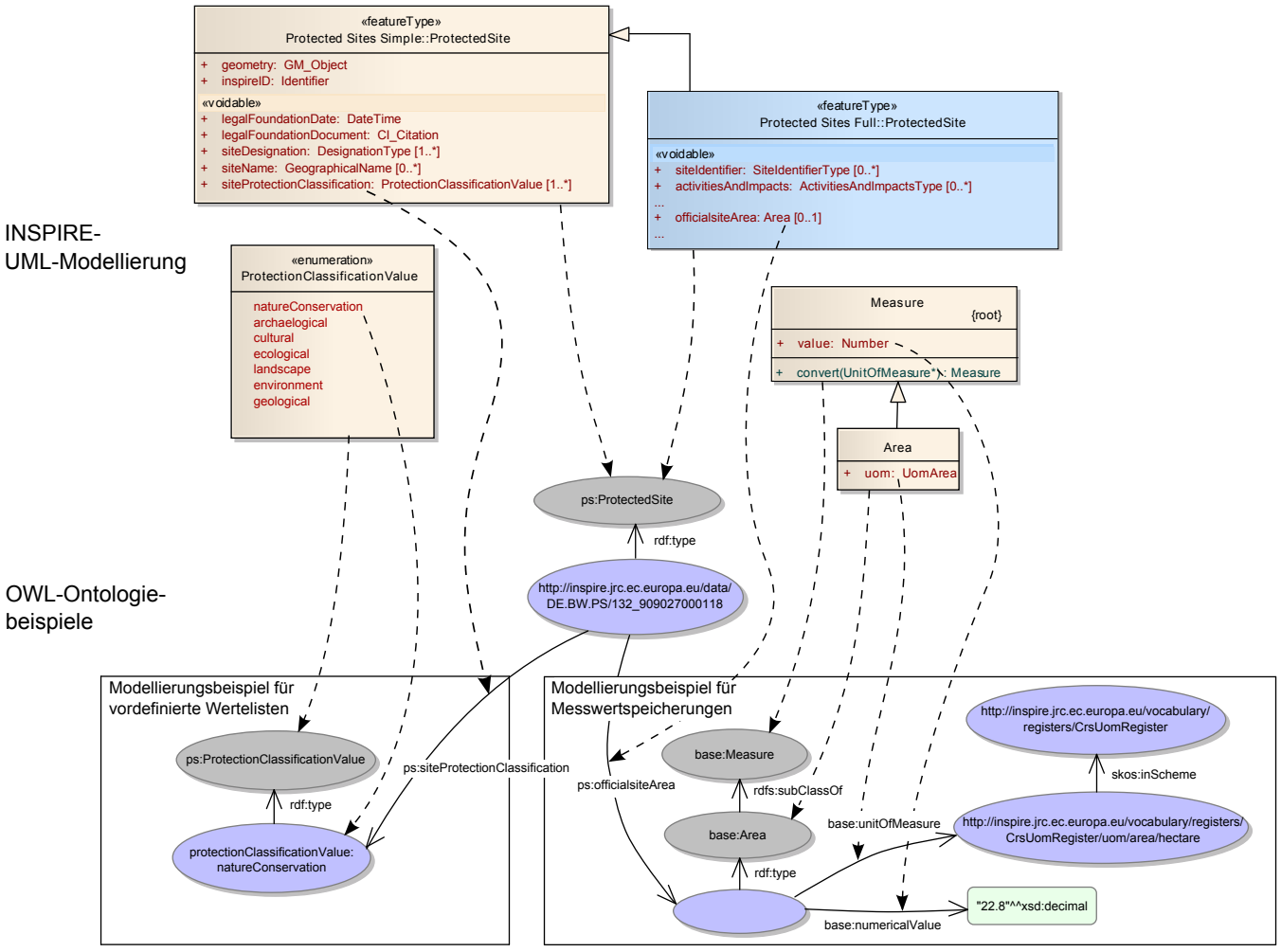


Abbildung 5.1 – OWL-Modellierung von vordefinierten Wertelisten und Messwerten

Messwertelemente, wie z.B. die UML-Datentypen `Area`, `Length` oder `Velocity`, bestehen üblicherweise aus zwei Informationen: dem numerischen Messwert und der Maßeinheit, in der gemessen wurde (engl. *unit of measure*; *uom* abgekürzt). Das Informationspaar kann beispielsweise als zusammenhängender Text in einem RDF-Literal übertragen werden, z.B. `"200 km2"`. Die Speichermethode ist aber hinderlich, denn beide Informationen müssen zu jeder Datenanalyse und -suche zunächst wieder voneinander getrennt werden. Eine bessere Alternative isoliert den

Messwert und transferiert die Maßeinheit als RDF-Datentyp, z.B. `"200"^^http://ex.org/units/km2`. Die zweite Variante führt zwar zu einer klaren Informationstrennung, jedoch muss für jede Maßeinheit ein RDF-Datentyp im Sinne einer RDF/OWL-Syntaxerweiterung definiert werden. Das Vorgehen kann leicht zu Interoperabilitätsproblemen mit Semantic Web-Werkzeugen führen, die nur auf konventionellen RDF-Datentypen operieren und darüber Typenvergleiche anstellen. Aus diesem Grund wird zu einer weiteren Alternative tendiert, die die Maßeinheit als OWL-Individuum speichert. Das Diagramm 5.1 zeigt die hier vorgeschlagene Lösung, die sich am Ansatz der *Measurement Units Ontology* (MUO[86]) orientiert. Jeder Messwerttyp wird auf eine OWL-Klasse gemappt (z.B. `base:Area`) und als Subtyp der OWL-Klasse `base:Measure` deklariert. Der numerische Wert ist über das OWL-Prädikat `base:numericalValue` in einem RDF-Literal des Datentyps `xsd:decimal` gespeichert und die Maßeinheit in Form eines OWL-Individuums über das OWL-Prädikat `base:unitOfMeasure` referenziert. Dank dieser Modellierungsmethodik kann ein Register für Maßeinheiten angelegt werden (siehe hierzu Abschnitt 5.1.3 *Harmonisierung von Referenzinformationen*). Desweiteren werden SPARQL-Anfragen erleichtert, indem Maßeinheiten nicht nur in SPARQL-Filterfunktionen, sondern auch innerhalb von SPARQL-Graphenmustern ausgewählt und Messwerte darüber eingeschränkt werden können. Die Abbildung 5.1 beinhaltet ebenfalls ein Modellierungsbeispiel für Messwerte; dargestellt sind ein Flächentyp (`Area`) und seine Speicherung.

Damit sind nun alle Abbildungsregeln angesprochen. Sie dienen dem Ziel, zur Überführung des INSPIRE-Fachvokabulars in eine ontologische Entsprechung eine größtmögliche Systematik anwenden und dennoch handhabbare und semantisch wertvolle Konzepte bilden zu können. Die Systematik erstreckt sich dabei auf die Namensgebung, die Analyse der häufigsten und wichtigsten INSPIRE ISO-Datentypen und die Abbildungsmöglichkeiten zwischen UML und RDF/OWL.

5.1.2 Identifikationsmanagement

Ein wesentlicher Grundsatz der Veröffentlichung von Linked Data- und anderen Semantic Web-Datenbeständen ist die eindeutige und persistente Identifizierung von Ressourcen. Im Sinne der Linked Data-Prinzipien sind Ressourcen vorzugsweise mit auflösbaren HTTP-URIs zu benennen, um eine Linkverfolgung durchführen und an der jeweiligen Internet-Adresse weiterführende Ressourceninformationen vorfinden zu können. Die Gesichtspunkte 1. Eindeutigkeit, 2. Persistenz und 3. HTTP-Auflösung gilt es auch bei der Identifizierung von INSPIRE-bezogenen Semantic Web-Ressourcen zu berücksichtigen. Dieser Abschnitt behandelt die Namensgebung bzw. Identifizierung dreier verschiedener INSPIRE-Ressourcenkategorien:

1. Konzepte der Themenontologien
2. Instanzen derjenigen Konzepte, die sich aus INSPIRE-Spatial object types (Stereotyp: `featureType`) herleiten und im Weiteren als *unabhängige Instanzen* bezeichnet werden
3. Instanzen derjenigen Konzepte, die sich aus INSPIRE-Data types (Stereotyp: `dataType`) herleiten und im Weiteren als *abhängige Instanzen* bezeichnet werden

Identifikatoren für Konzepte

Im Allgemeinen leitet sich die URI eines OWL-Konzeptes von der URI der zugehörigen Ontologie ab. Damit alle INSPIRE-Themenontologien und -Konzepte einem einheitlichen Namensraum angehören, wird die Basis-URI mit `http://inspire.jrc.ec.europa.eu` festgelegt und ein Muster zur Benennung der Ontologie-URIs vorgegeben (Variablen sind als kursive Segmente wiedergegeben):

`http://inspire.jrc.ec.europa.eu/vocabulary/themes/`*`<INSPIRE-Themenontologie>`*`/`

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

Demzufolge lautet die Ontologie-URI des Annex-Themas *Schutzgebiete*:

`http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/`

Für zugehörige Konzepte wird das Muster erweitert zu:

`http://inspire.jrc.ec.europa.eu/vocabulary/themes/<INSPIRE-Themenontologie>/<Ontologiekonzept>`

Ein Konzept `ResponsibleAgency` bekommt demnach die URI:

`http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/ResponsibleAgency`

Die beispielhafte Basis-URI ist natürlich nur als Vorschlag zu verstehen, das Einrichten einer Internetdomäne zur langfristigen Bereitstellung von INSPIRE-Konzepten kann nicht Gegenstand dieser Arbeit sein. Für einen offenen Umgang mit INSPIRE-Konzepten im Semantic Web ist dieser organisatorische Schritt jedoch unerlässlich.

Die hier getroffenen Namenskonventionen lassen sich auf alle INSPIRE-Themen und -Konzepte anwenden. Von dieser Regelung ausgenommen seien INSPIRE-Basiskonzepte, die in eine separate Domänenontologie mit der Ontologie-URI `http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes/` ausgelagert werden (siehe hierzu Abschnitt 5.1.6). Eingebundene Domänenontologien, die wie z.B. GeoSPARQL extern gepflegt werden, besitzen vorgegebene Ontologie-URIs, z.B. lautet die URI des GeoSPARQL-Geovokabulars: `http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/`.

Identifikatoren für unabhängige Instanzen

In den INSPIRE GML-Applikationsschemata sind GML-Features - respektive INSPIRE Spatial objects - mit bis zu drei Identifizierungsmerkmalen ausgestattet, die sich unterschiedlich gut für OWL-Instanzen eignen. Erstens verfügt jedes Feature obligatorisch über das XML-Attribut `gml:id`. Dessen Eindeutigkeit ist jedoch per Definition auf ein GML-Instanzdokument bzw. eine GML-FeatureCollection, einer Sammlung von GML-Features, beschränkt und würde demnach keine eindeutige Benennung von INSPIRE OWL-Instanzen herbeiführen. Die zweite Möglichkeit, die sich mit dem GML-Element `gml:identifier` bietet, scheidet ebenfalls aus, weil `gml:identifier` als optional definiert und die lückenlose Verwendung des Elementes seitens der Datenbereitstellern nicht sichergestellt ist. Das Projekt INSPIRE führt aufgrund der genannten Widrigkeiten ein neues GML-Element namens `base:Identifier` ein. Ein solcher INSPIRE-Identifikator, auch *INSPIRE-Id* oder *external object identifier* genannt, besteht aus drei sich ergänzenden und global eindeutigen Teilinformationen 1. `namespace`, 2. `localId` und 3. `versionId`. Die `namespace`-Angabe benennt die datenführende Stelle und/oder das Datenprodukt. Innerhalb des `namespace`-Kontextes wird eine Ressource per `localId`, zuzüglich einer optionalen `versionId` für Versionierungszwecke, eindeutig bestimmt.

Das Element `Identifier` ist für alle INSPIRE-Features der fertig spezifizierten Annex-I-Themen verpflichtend vorgeschrieben. Bedauerlicherweise ist man in den Entwürfen der Datenmodelle zu Annex-II/III dazu übergegangen und definiert das `Identifier`-Element nur noch optional. Sollten die Datenmodelle für Annex-II/III in dieser Form verabschiedet werden, ließen sich die jeweiligen Instanzdaten nicht global unterscheiden, was ein übergeordnetes INSPIRE-Datenmanagement erheblich erschweren würde. Es bleibt also zu hoffen, dass das verantwortliche INSPIRE-Drafting Team eine Änderung an den neueren Datenmodellen vornimmt. Der ungewissen *Zukunftsmusik* kann an dieser Stelle keine Beachtung geschenkt werden. Deshalb gehen die folgenden Abhandlungen von der Annahme aus, dass jegliche `Identifier`-Elemente verpflichtend sind und dadurch die Grundvoraussetzung für eindeutig identifizierbare Semantic Web-Ressourcen schaffen.

Wie sich INSPIRE-Identifikatoren mit ihren drei Teilinformationen zweckmäßig ausgestalten lassen, wird im INSPIRE-Infrastrukturdokument *Guidelines for the encoding of spatial data* [INSPIRE Drafting Team „Data Specifications“

2010*b*] in Kapitel B.2 *Use of URIs* beschrieben. Interessanterweise finden sich dort auch Beispiele für INSPIRE-Identifikatoren in Form von HTTP-URIs. Die *data.gov.uk*-Initiative der britischen Verwaltung, die sich u.a. nennenswert am Aufbau des Semantic Web beteiligt, zeigt darin auf, nach welcher URI-Syntax INSPIRE-Ressourcen aus britischer Hand benannt sein könnten:

`http://<section>.data.gov.uk/doc/<namespace>/<localId>[/<versionId>][<rendition>]`

Ein gegebenes Beispiel lautet:

`http://location.data.gov.uk/doc/osgb/1234567890123456`

Diese URI-Syntax gibt einige hilfreiche Anregungen. Zum einen sind die drei Identifikatorattribute `namespace`, `localId` und `versionId` als einzelne URI-Teilpfade abgebildet, so dass auch ihre hierarchische Ordnung in der Reihenfolge der URI-Teilpfade beachtet wird.[87]. Die im URI-Muster enthaltene Variable `rendition` (dt. Wiedergabe) deutet auf die Differenzierung in der Ausgabeform der Ressource hin, die im *Content Negotiation* im Rahmen von Linked Open Data eine gewichtige Rolle spielt: z.B. als HTML-Repräsentation (Wert: `page`), als RDF/XML-Ressource (Wert: `data`) (siehe Abschnitt 2.2.4). Die Information über die Ausgabeform soll auch im URI-Muster eingeführt werden, das in diesem Konzept vorgeschlagen wird. Zur besseren Verständlichkeit sei der Variablenname von `rendition` in `format` abgewandelt. Desweiteren wird die Variable `format` als eigenständiger URI-Teilpfad in der URI-Syntax weiter links eingeordnet, um die leichte Lesbarkeit der INSPIRE-Identifikatorattribute u.a. auch für automatische Ausleseprozesse zu steigern. Daraus ergibt sich das URI-Muster für unabhängige INSPIRE OWL-Instanzen wie folgt:

`http://inspire.jrc.ec.europa.eu/<format>/<namespace>/<localId>[/<versionId>]`

Eine beispielhafte Ressourcen-URI als RDF/XML-Repräsentation:

`http://inspire.jrc.ec.europa.eu/data/NL.KAD.AU.GEM/4507/V1.0`

Die Forderung nach der Auflösbarkeit von HTTP-Adressen ist für INSPIRE OWL-Instanzen weitaus schwieriger zu erfüllen als für INSPIRE-Konzepte. Während Themenontologien an einer zentralen Stelle registriert und darüber ihre Konzepte zugänglich gemacht werden können, sind INSPIRE-Features über diverse INSPIRE-Downloaddienste verteilt. Die INSPIRE-Architektur ist nicht darauf ausgelegt, einzelnen Features individuelle Zugriffsadressen einzuräumen, so wie dies im Semantic Web der Fall ist, wo z.B. *Linked Data-Endpoints* den Direktzugriff auf Ressourcen gestatten. Zwar lassen sich INSPIRE-Features leichthin von GML nach RDF/OWL umformen und die abgeleiteten OWL-Instanzen gemäß der obigen URI-Syntax benennen. Der umgekehrte Weg, eine OWL-Instanz mit bereits kodierter URI (siehe obiges Beispiel) erneut vom zuständigen Downloaddienst abzurufen, weitere Attribute zu filtern und auszugeben, ist nicht ohne zusätzliche technische Vorkehrungen möglich. Das Internet setzt für solche Belange sogenannte *Resolver*-Systeme ein. Es handelt sich dabei um Dienste, die dauerhafte Adressen pflegen und jene dauerhaften Adressen anhand von Mapping-Tabellen auf konkrete und zeitlich variable Speicheradressen von Ressourcen auflösen. Ändert sich die konkrete Speicheradresse, so muss nur die Mapping-Tabelle daraufhin angepasst werden, die dauerhafte Ressourcenadresse bleibt jedoch erhalten. Bekannte und auf Resolver-basierende Identifizierungsdienste heißen *Digital Object Identifier* (DOI[88]) und *Persistent Uniform Resource Locator* (PURL[89]). Für INSPIRE OWL-Instanzen könnte ein ähnliches System zentral eingerichtet und gewartet werden, das die Adressenauflösung vorzugsweise über den INSPIRE-Namensraum abwickelt (Attribut `namespace`). Das Sequenzdiagramm in Abbildung 5.2 zeigt die beispielhafte Bearbeitung einer Benutzeranfrage zum Auffinden und Ausgeben einer INSPIRE OWL-Instanz mitsamt der zugehörigen Prädikatinformationen.

[87] die Hierarchie der INSPIRE-Identifikatorattribute ergibt sich wie folgt: Namensräume (`namespace`) gruppieren lokale Identifikatoren (`localId`), die wiederum die zugehörigen Versionierungsidentifikatoren (`versionId`) gruppieren. Hinsichtlich einer URI gilt folgende Regel: je weiter links der URI-Teilpfad vorkommt, umso höher ist dessen Stellenwert innerhalb der URI-Adresse

[88] Webseite der International DOI Foundation: http://www.doi.org

[89] Webseite: http://purl.oclc.org

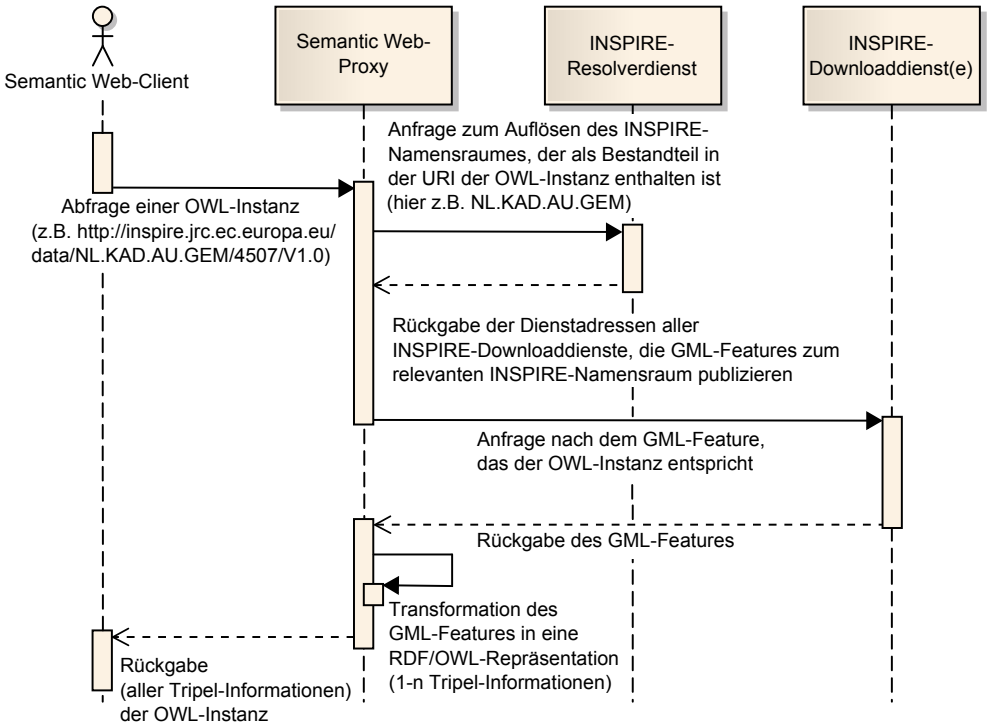


Abbildung 5.2 – Ablauf einer OWL-Instanzanfrage mit URI-Resolving

Identifikatoren für abhängige Instanzen

Komplexe INSPIRE-Data Types (Elemente des Stereotyps `dataType`) gruppieren zwei oder mehr Unterelemente, sind jedoch als *schwache* Elemente selbst abhängig von übergeordneten INSPIRE-Features. Denn im Gegensatz zu Features (Stereotyp `featureType`) besitzen sie keine eigenen Identifizierungsmerkmale (Element `base:Identifier`), weswegen sich keine eindeutige URI für vergleichbare OWL-Instanzen bilden lässt. Das vereinfachte UML-Diagramm 5.3 soll diesen Sachverhalt verdeutlichen. Der abgebildete Feature Type `ProtectedSite` enthält einen INSPIRE-Identifikator (UML-Attribut `inspireID`), wohingegen die Data Types `DesignationType` und `PresentHabitatType` ohne Identifikator definiert sind.

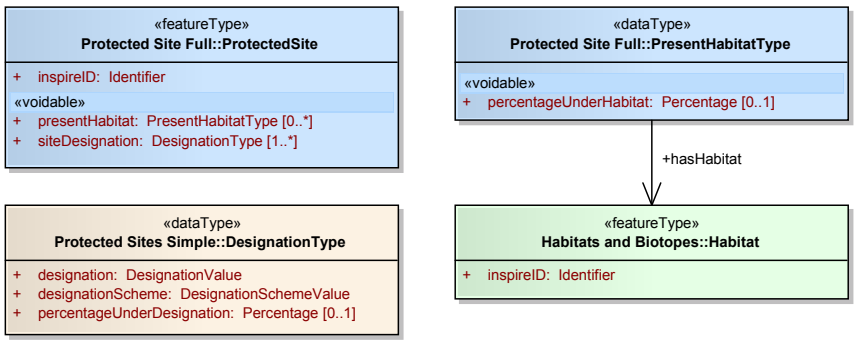


Abbildung 5.3 – Identifizierung von INSPIRE-Feature Types und -Data Types [90]

[90] Vereinfachung des UML-Diagramms 2.9 zum INSPIRE-Thema *Protected Sites*. Farblich unterschieden sind die jeweiligen INSPIRE-Namensräume, deren Package-Namen im Kopf der UML-Klassennotationen festgehalten sind

Es soll nun aufgezeigt werden, wie sich solch abhängige OWL-Instanzen trotz fehlenden INSPIRE-Identifikators mit einer URI auszeichnen lassen. Dabei stehen angesichts eines virtuellen Repositorys, das dem OWS-Proxy zugrunde liegt, nicht die Möglichkeiten statischer RDF-Triplestores zur Verfügung. Es können also keine nachträglichen Identifizierungsprozesse dazu dienen, Blank Nodes in persistente, synthetische URIs umzuwandeln. Die Lösungsstrategie beruht auf der Serialisierung mit Hash-Werten. In der Informationstechnik werden zu verschiedenen Zwecken, z.B. zur Verschlüsselung oder für Prüfsummen, beliebig lange Textnachrichten in eine kryptische Form konstanter Länge (128 Bit) transformiert. Dafür stehen verschiedene Hash-Algorithmen zur Auswahl, u.a. die bekannten *MD5*-Summen. Das Beispiel unten zeigt die Serialisierung einer Data Type-Instanz, ausgegeben als MD5-Summe in Hexadezimalschreibweise:

Aus der Data Type-Instanz einer *Schutzgebietewidmung*:

```
<ps:siteDesignation>
 <ps:DesignationType>
  <ps:designationScheme>Natura2000DesignationValue</ps:designationScheme>
  <ps:designation>siteOfCommunityImportance</ps:designation>
 </ps:DesignationType>
</ps:siteDesignation>
```

folgt die MD5-Summe: fbb94c1be34cfb79c3c18940b0293be1

Natürlich kann ein Hash-Wert nicht allein die konkrete Data Type-Instanz identifizieren. Zur eindeutigen Identifikation ist auch die URI der übergeordneten und unabhängigen OWL-Instanz (entsprechend einem Feature) sowie das OWL-Prädikat als Pfadangabe zur abhängigen OWL-Instanz (entsprechend einem Data Type) vonnöten. Daraus ergibt sich eine zusammengesetzte dreigeteilte URI-Syntax, bestehend aus:

1. URI-Syntax für unabhängige OWL-Instanzen:
`http://inspire.jrc.ec.europa.eu/`*`<format>`*`/`*`<namespace>`*`/`*`<localId>`*`[/`*`<versionId>`*`]`

2. Das verbindende OWL-Prädikat: *<predicate>*

3. Der Hashwert als Serialisierung der abhängigen OWL-Instanz: *<hashContent>*

Mittels Trennzeichen (hier `+`-Delimiter) zusammengesetzt, lautet die finale URI-Syntax:

`http://inspire.jrc.ec.europa.eu/`
`<format>``/`*`<namespace>`*`/`*`<localId>`*`[/`*`<versionId>`*`]`+*<predicate>*+*<hashContent>*

Ein konkretes Beispiel sieht dann wie folgt aus:

http://inspire.jrc.ec.europa.eu/data/SK_PS_8/SKUEV0151
+ps:siteDesignation+fbb94c1be34cfb79c3c18940b0293be1

Erschwerend kommt hinzu, dass INSPIRE-Data Types ihrerseits wiederum weitere Data Types, häufiger jedoch externe INSPIRE-Feature Types referenzieren. Letztgenannter Fall ist im vereinfachten UML-Diagramm 5.3 dargestellt. Der Data Type `PresentHabitatType` verweist über die UML-Assoziation `hasHabitat` auf einen Feature Type namens `Habitat`. Auch der Feature Type `Habitat` kann wiederum Referenzen bzw. Querverweise auf weitere Objekttypen enthalten, so dass die konkreten Elementinhalte ausgehend von einer Data Type-Instanz *ins Bodenlose führen könnten* und der Serialisierungsvorgang abgebrochen werden muss. Die Problematik lässt sich lösen, indem innerhalb einer Data Type-Instanz die nachgelagerten Features (hier vom Feature Type `Habitat`) mit der vorgeschlagenen URI-Syntax für unabhängige OWL-Instanzen kodiert und damit die Referenzen auf jene Features substituiert werden. Erst im Folgeschritt wird dann die Serialisierung der Data Type-Instanz mittels Hash-Algorithmus angestoßen.

Als Fazit ist festzuhalten, dass die eindeutige Referenzierung von abhängigen OWL-Instanzen ein schwieriges Unterfangen ist. Darüber hinaus ist die Frage zu stellen, ob überhaupt jede OWL-Instanz eine Identifizierung benötigt

oder eine Lockerung der strikten Linked Data-Prinzipien einer gewissen Praxis-Nähe gleich käme, die in anderen Semantic Web-Projekten anzutreffen ist. Diese und weitere Gesichtspunkte werden im Abschnitt 5.3.4 *Testergebnisse* einer Prüfung unterzogen.

5.1.3 Harmonisierung von Referenzinformationen

Die INSPIRE-Datenmodelle stellen viele themenübergreifende **Bezüge zu zentralen Informationseinheiten** her, die hier als *Referenzinformationen* bezeichnet werden. Dieser Abschnitt gibt einen Überblick über die in INSPIRE gebräuchlichen Referenzinformationen, wie z.B. Sprachen- und Länderkodierungen, Maßeinheiten und Koordinatenreferenzsysteme, und analysiert deren Anwendung und Identifizierung im Vergleich zum gegenwärtigen Umgang derartiger Referenzinformationen im Semantic Web. Die Gegenüberstellung dient dazu, Möglichkeiten zur Überführung von INSPIRE-Referenzinformationen in Semantic Web-Speicherformen zu erörtern und diesbezügliche Empfehlungen für eine organisatorische wie technische Umsetzung auszusprechen.

Referenzinformationen besitzen häufig sprachenunabhängige und eindeutige Identifikationsschlüssel und weisen präzise Definitionen in gegebenenfalls mehrsprachiger Ausführung auf. Aufgrund ihrer häufigen Anwendung sind die sensiblen Referenzinformationen günstigerweise durch eine zentrale Stelle zu pflegen, die inhaltliche Modifikationen kontrolliert und für eine hohe Beständigkeit sorgt. Zur Organisation und Wartung von Referenzinformationen kommen in INSPIRE die nach ISO 19126 [ISO/TC 211 2009] und ISO 19135 [ISO/TC 211 2005*c*] standardisierten **Registrierdienste** zum Einsatz, die sogenannten *Consolidated Registers*.[91]. Ein Registrierdienst beherbergt Register, wie z.B. das INSPIRE-Register zu Koordinatenreferenzsystemen und Maßeinheiten (*CRS/UoM-Register*). Die Register sind ihrerseits unterteilt in Registereinträge, die konkrete Definitionen beispielsweise der Maßeinheiten *Meter* oder *Fuß* enthalten. Das W3C-geprägte Semantic Web strukturiert das Wissen über Referenzinformationen mit Hilfe von **Ontologien**. Die Ontologiesprache OWL bietet dafür vielfältige syntaktische Mittel, um vergleichbare Register und Registereinträge in ontologischer Form mitsamt Definitionsinformationen zu bestücken und zugleich beliebige semantische Relationen zur Verknüpfung von Einträgen einzuführen.

Das erst jüngst fertigspezifierte **W3C-Vokabular SKOS** (siehe Abschnitt 2.2.4) soll dabei helfen, die Publizierung der Inhalte von *Wissensorganisationssystemen* (engl. *Knowledge Organisation Systems*, abgekürzt: KOS) im Semantic Web zu vereinheitlichen. Als KOS kommen z.B. Taxonomiesysteme und Thesauri in Frage. Auch die in INSPIRE betriebenen ISO-Registrierdienste zählen darunter. Ihre ISO-Register und -Registereinträge eignen sich hervorragend zur Beschreibung und Veröffentlichung von SKOS-Vokabularkonzepten. Ein praktisches Beispiel hierzu kommt vom *Joint Research Center* (JRC[92]), dem technischen INSPIRE-Koordinator und u.a. der verantwortlichen Stelle zum Aufbau des INSPIRE-Geoportals. In dieser Funktion hat das JRC das Fachvokabular des Umweltthesaurus *General Multilingual Environmental Thesaurus* (GEMET[93]) und die INSPIRE-Register des Feature Catalogue und des Feature Concept Dictionary in SKOS-Konzepte abgebildet und unter dem Namen *Semantic Lab*[94] zur Recherche freigegeben. An dieser Stelle sei auch angemerkt, dass das OGC eine ähnliche Strategie für verschiedene OGC-Spezifikationselemente[95] verfolgt. Hierfür nutzt das OGC SKOS-Prädikate, ohne jedoch die RDF-Ressourcen explizit als SKOS-Klassenkonzepte zu typisieren.

[91] siehe Generic Conceptual Model [INSPIRE Drafting Team „Data Specifications“ 2010*a*], Kapitel 17 *Registers and registries*
[92] Webauftritt: http://ec.europa.eu/dgs/jrc
[93] Projektseite: http://www.eionet.europa.eu/gemet
[94] Projektseite mit integrierter Suchoberfläche: https://semanticlab.jrc.ec.europa.eu/
[95] GML/RDF/HTML-Definitionen, die vom OGC gepflegt werden, sind zu finden unter: http://www.opengis.net/def; die Einteilung in Kategorien wird beschrieben unter: http://www.opengis.net/def/def-type/

Genereller Ansatz zur Behandlung von Referenzinformationen

Dem Beispiel des JRC folgend wird vorgeschlagen, alle INSPIRE-Register auf gleiche Weise in SKOS-Konzepte zu überführen und zu veröffentlichen. Und zwar derart, dass für jedes ISO-Register ein SKOS-Schema und für jeden Registereintrag ein SKOS-Konzept angelegt wird. Zur weiteren Untergliederung innerhalb der Register sollen auch übergeordnete SKOS-Konzepte eingeführt werden, die als sogenannte *broader term* deklariert die spezialisierteren Konzepte (*narrower term*) gruppieren. Übertragen auf die vielfach verwendeten INSPIRE-Codelists/-Enumerations wäre eine denkbare SKOS-Hierarchie:

1. **Register** für Codelists/Enumerations
   URI: http://inspire.jrc.ec.europa.eu/vocabulary/base/CodelistRegister
2. **Übergeordneter Registereintrag** (*broader term*) für einzelne Codelists/ Enumerations,
   z.B. `ActivityValue` aus der Datenspezifikation für *Schutzgebiete* (*ProtectedSites*)
   URI: http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/ActivityValue
3. **Untergeordneter Registereintrag** (*narrower term*) für betreffende Codelist-/ Enumeration-Werte,
   z.B. `agriculturalStructures` der Codelist `ActivityValue`
   URI: http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/ActivityValue/agriculturalStructures

Der automatischen Generierung von SKOS-Konzepten aus den INSPIRE ISO-Registern steht konzeptionell nichts im Weg. Daran anschließen ließe sich ein semantisches Alignment bekannter Semantic Web-Vokabularien, damit die Verknüpfung identischer Semantic Web-Konzepte der Wissensvernetzung und Inferenzbildung Vorschub leisten kann. Zur Veranschaulichung soll das Beispiel eines SKOS-Konzeptes dienen. Das Codebeispiel 13 zeigt die Definition eines SKOS-Konzeptes für die Maßeinheit *Meter* auf.[96] Die dargestellte OWL-Instanz (`rdf:type owl:NamedIndividual`; Zeile 2) wird verschiedentlich benannt (Zeilen 3-5), definiert (Zeile 6), einem übergeordneten Register (`skos:Scheme`; Zeile 8) und einem allgemeineren Begriff zugeordnet (`skos:broader`; Zeile 9). Darüber hinaus wird das Konzept mit einem OWL-Prädikat des MUO-Vokabulars als Maßeinheit der Länge klassifiziert (Zeile 10) und mit bestehenden und semantisch identischen Semantic Web-Konzepten verlinkt (Zeilen 11-15).

```
1  <skos:Concept rdf:about="http://inspire.jrc.ec.europa.eu/vocabulary/base/CrsUomRegister/uom/length/meter">
2   <rdf:type rdf:resource="&owl;NamedIndividual"/>
3   <rdfs:label>meter</rdfs:label>
4   <skos:prefLabel>meter</skos:prefLabel>
5   <skos:altLabel>metre</skos:altLabel>
6   <skos:definition>The metre is the length of the path travelled by light in vacuum
7           during a time interval of 1/299 792 458 of a second.</skos:definition>
8   <skos:inScheme rdf:resource="http://inspire.jrc.ec.europa.eu/vocabulary/base/CrsUomRegister"/>
9   <skos:broader rdf:resource="http://inspire.jrc.ec.europa.eu/vocabulary/base/CrsUomRegister/uom/length"/>
10  <muo:measuresQuality rdf:resource="http://purl.oclc.org/NET/muo/ucum/physical-quality/length"/>
11  <owl:sameAs rdf:resource="http://purl.oclc.org/NET/muo/ucum/unit/length/meter"/>
12  <owl:sameAs rdf:resource="http://www.opengis.net/def/uom/OGC/1.0/metre"/>
13  <owl:sameAs rdf:resource="urn:ogc:def:uom:OGC:1.0:metre"/>
14  <owl:sameAs rdf:resource="urn:ogc:def:uom:UCUM::metre"/>
15  <owl:sameAs rdf:resource="urn:ogc:def:uom:UNECE::MTR"/>
16 </skos:Concept>
```

Codebeispiel 13 – Definitionsbeispiel eines SKOS-Konzeptes

Ein wesentliches Semantic Web-Prinzip ist die Wiederverwendung externer Vokabulare. Die oben vorgeschlagene Neudefinition bzw. Ableitung von SKOS-Konzepten aus den INSPIRE-Registern ist aber insofern unumgänglich, da INSPIRE viele anwendungsspezifische Besonderheiten, wie z.B. die INSPIRE-Codelists und -Enumerations, mitbringt.

[96] die Inhalte sind größtenteils der OGC-Informationsressource http://www.opengis.net/def/uom/OGC/1.0/metre entnommen

Desweiteren sind viele Semantic Web-Projekte zur Katalogisierung von Referenzinformationen zwar terminologisch durchdacht, es mangelt ihnen aber häufig an vollständigen und harmonisierten assertionalem Faktenwissens. Dies betrifft insbesondere die eher seltenen Referenzeinheiten, wie z.B. die Maßeinheit `Hektar`. Deshalb müssen je nach Informationskategorie INSPIRE-spezifische Lösungen gefunden werden, weshalb nun eine genauere Betrachtung ausgewählter Kategorien folgt.

Sprach- und Länderkodierungen

Die Ausweisung einer Textsprache trägt grundsätzlich zur Interoperabilität in der EDV bei. Sprachangaben sind zumeist eine Art Beiwerk bzw. Attributierung (*Label*) eines Textes. In den seltesten Fällen wird die Sprache als eigenständige Ressource aufgefasst, nach der semantisch gefragt werden kann, ob z.B. *eine bestimmte Softwaredokumentation in chinesischer Sprache vorhanden ist?* [97]. Aus diesem Grund hat das W3C, der Urheber der RDF-Spezifizierungsserie, für RDF-Literale die Option vorgesehen, sie mit Sprachkürzeln (*language tag*) zu markieren. Die Sprachkürzel sollen dabei den Vorgaben aus dem Request for Comments (RfC) 3066 der *Internet Ingineering Task Force* (IETF[98]) entsprechen.[99] Nach dem RfC 3066 sind vornehmlich Sprachenkodierungen (z.B. `de`) oder Kombinationen aus Sprachen- und Länderkodierungen (z.B. `en-US`: Englisch/USA) anzuwenden. Dabei stützt sich der RfC 3066 auf die ISO-Standards ISO 639-1/2 (zwei- bzw. dreistellige Alpha-Codes) für Sprach- und ISO 3166 für Länderkodierungen. Der RfC 3066 lässt aber ausdrücklich auch weitere sogenannte *Subtags* zu, wie z.B. IANA-Bezeichner[100] für Zeichensätze. Trotz dieser vielfältigen Optionen, sind RDF-Literale vorwiegend mit zweistelligen ISO 639-1 Codes beschrieben. Diese gängige Praxis vereinfacht die Konvertierung von INSPIRE-Sprachangaben in Semantic Web-Entsprechungen erheblich.

In INSPIRE existieren drei Elementtypen mit sprachlicher bzw. geographischer Einordnung:

- **gmd:LocalisedCharacterString**: Elementtyp, der den XSD-Datentyp `xsd:string` um ein XML-Attribut namens `locale` vom Typ `xsd:anyURI` erweitert (z.B. `nl-NL`). Ein *Locale* ist in der Informationstechnik für gewöhnlich ein Satz an Einstellungen für Computerprogramme. Das INSPIRE Generic Conceptual Model [INSPIRE Drafting Team „Data Specifications“ 2010*a*] beschreibt ein Locale als „*eine Kombination aus einer Sprache, eventuell eines Landes und/oder eines Zeichensatzes*“. Diese Definition ähnelt den Ansprüchen des RfC 3066. Leider ist das `locale`-Attribut damit nur unzureichend spezifiert und einem Datenadministrator wird freie Wahl gelassen, ob er die Locale-Information standardkonform kodiert (z.B. zu RfC 3066) oder eine andere Repräsentationsform wählt.
- **gn:GeographicalName**: Komplexer Elementtyp, der die Sprache ohne weitere Länder- oder Zeichensatzangaben im Unterelement `gn:language` speichert. Hierfür schreibt INSPIRE dreistellige ISO 639-3 Sprachen-Codes bzw. die ebenfalls dreistelligen ISO 639-5 Codes zur Strukturierung in Sprachfamilien vor. Eine Sonderstellung nehmen mehrsprachige geographische Bezeichner ein: in einem solchen Fall wird `mul` für *multilingual* eingetragen. Zwar ist das Unterelement `gn:language` nur optional anzugeben (Stereotyp `voidable`), die Angabe wird jedoch vom zuständigen INSPIRE-Drafting Team dringend empfohlen.
- **CountryCode**: Länderkodierungen sind in einer eigenständigen Codelist namens `CountryCode` im INSPIRE UML-Package `Base Types` der konsolidierten UML-Modelle aufgenommen. Hier erfährt man auch, dass die vorgegebenen Codelist-Werte dem *interinstitutionellen Style Guide* des Publikationsbüros der Europäischen Union

[97] das Fragebeispiel entstammt der Webseite http://www.lingvoj.org/, auf der semantische Sprachressourcen (konform zu ISO 639-3) gehostet wurden. Mittlerweile ist dieser Datenbestand verschmolzen mit dem Linked Data-Projekt *Lexvo.org* (Projektseite: http://lexvo.org). Lexvo.org strukturiert standardkonforme Bezeichner und Kodierungen für Sprachen, Länder bzw. geographische Regionen, Zeichensätze und Zeichenkodierungen

[98] Webauftritt: www.ietf.org/

[99] der Verweis auf den RfC 3066 erfolgt in der Recommendation *RDF Semantics* (RDFS) [Hayes 2004]

[100] die Internet Assigned Number Authority (IANA) registriert abgestimmte Listen von Nummerierungen bzw. Kodierungen zwecks einer interoperablen Kommunikation in Internetprotokollen, siehe: http://www.iana.org/protocols/

entstammen [101]. Das EU-Publikationsbüro bezieht sich auf die zweistelligen Alpha-Codes von ISO 3166, legt jedoch zwei Ausnahmen fest: `UK` anstelle von `GB` für Großbritannien und `EL` anstelle von `GR` für Griechenland.

Nach dieser Bestandsaufnahme sind pragmatische Mapping-Anweisungen vonnöten. Es wird zwecks GML → RDF/OWL-Transformation deshalb empfohlen, aus den textuellen Elemententypen `gmd:LocalisedCharacterString` und `gn:GeographicalName` zusätzlich zum Textinhalt nur die Sprachangaben herauszufiltern[102] und als *language tag* eines RDF-Literals zu verwenden. Die seltener angewendeten Länderkodierungen oder linguistischen Angaben werden ignoriert, um die primären Informationen (Text + Sprachkürzel) hervorzuheben und die Semantic Web-Repräsentationen so kompakt wie möglich zu halten. Sollten INSPIRE-Sprachkodierungen dreistelligen ISO-Codes nach 639-2 bzw. 639-3 und -5 gleichen, so werden sie - sofern möglich - in zweistellige ISO 639-1 Codes übersetzt, z.B. `ger` bzw. `deu` nach `de` umgewandelt. Die Codelist `CountryCode` sollte getrennt und nach dem Muster anderer INSPIRE-Codelists/-Enumerations behandelt werden. Siehe hierzu die Ausführungen zur Ontologiemodellierung in Abschnitt 4.1 und die Bildung von SKOS-Konzepten in diesem Abschnitt.

Maßeinheiten

Jeder Messwert - mit Ausnahme von absoluten Zahlenwerten - erhält seine Bedeutung erst in Verbindung mit einer Maßeinheit. Im Bereich des Semantic Web existieren einige Ontologien, die Maßeinheiten führen. Die beiden ausgereiftesten Ontologien sind vermutlich die der NASA (Teilontologie der *Semantic Web for Earth and Environmental Terminology*: SWEET[103]) sowie die *Ontologie Measurement Units Ontology* (MUO[104]) des freien Softwareprojektes *Morfeo*. Letztere ist angelehnt an die Informationsstruktur des allgemeinen, nicht im Semantic Web verankerten Kodierungssystems *Unified Code for Units of Measure* (UCUM[105]) mitsamt seinen Ableitungen von SI-Einheiten, Kategorien und Kombinationen von Basisgrößen. Das Kodierungssystem UCUM orientiert sich zwar an internationalen Standards von ISO und ANSI, geht aber darüber hinaus und erlaubt moderne Maßeinheiten u.a. mit der Unterstützung aktueller Zeichensätze. Eine vergleichbare Bedeutung kommt der *United Nations Economic Commission for Europe* (UNECE) bzw. ihrer Empfehlung eines Kodierungssystems für Maßeinheiten zu, bezeichnet als: *UNECE Recommendation No.20, Codes for Units of Measure (UoM) used in international trade*[106].

Gegenüber der UNECE Rec. No.20 hat das UCUM-System Vorteile in doppelter Hinsicht. Erstens ist durch die MUO-Ontologie eine semantische Repräsentation der UCUM vorhanden, und das sogar sowohl auf TBox- wie auch auf ABox-Level. Zweitens ergibt sich eine gewisse Abhängigkeit von INSPIRE in Richtung UCUM. Insofern, dass alle Messwert-Elementtypen in INSPIRE auf den Elementtypen der GML-Schemata `measures.xsd` und `basicTypes.xsd` beruhen, z.B. `LengthType` und `AreaType`. Diese Elementtypen sind abgeleitet vom XML-ComplexType `MeasureType`, der abgesehen vom Messwert des Datentyps `xsd:double` auch ein XML-Attribut namens `uom` besitzt. Hierin sollen Maßeinheiten festgehalten werden, entweder als Symbol (z.B. `m` für Meter) oder als URI. In der beigefügten Beschreibung der XSD-Elementdefinition steht ein expliziter Hinweis auf das Kodierungssystem UCUM, deren Symbolik im OGC-Kontext empfohlen wird. Auch die URI-Variante kann sich an den Symbol-Bezeichnern von UCUM orientieren. Mit der Verwendung von UCUM-Kodierungen würden Semantic Web und Geo Web *auf den gleichen Nenner gebracht*. Es wäre sehr hilfreich, wenn die genannten Gesichtspunkte auch bei der Erstellung des INSPIRE CRS/UOM-Registers Berücksichtigung finden würden, so dass aus dieser abgestimmten Quelle leichthin SKOS-Konzepte für Maßeinheiten gewonnen werden könnten.

[101] siehe Länder-Codes unter http://publications.europa.eu/code/en/en-370100.htm
[102] für das Element `gmd:LocalisedCharacterString` wird die Sprachangabe aus dem Attribut `locale` gewonnen, für das Element `gn:GeographicalName` aus dem Unterelement `gn:language`
[103] Projektseite: http://sweet.jpl.nasa.gov/; eine Ontologiebeschreibung für wissenschaftliche Maßeinheiten ist zu finden unter: http://sweet.jpl.nasa.gov/2.1/reprSciUnits.owl
[104] Dokumentation: http://forge.morfeo-project.org/wiki_en/index.php/Units_of_measurement_ontology
[105] Dokumentation: http://unitsofmeasure.org/
[106] Download: http://www.unece.org/cefact/recommendations/rec_index.html

Koordinatenreferenzsysteme

Dem nachfolgenden Abschnitt 5.1.4 *Transfer geographischer Information* wird der Aspekt der Georeferenzierung vorweggenommen, da es sich bei Identifikatoren von Koordinatensystemen (engl. Akronym: CRS) um klassische Referenzinformationen handelt, die entsprechend zu organisieren ist. Semantic Web-Projekte verwenden für die Angabe von Koordinaten zumeist die Georeferenzierung WGS84 mit geographischen Koordinaten. Hierbei erfolgt häufig keine explizite CRS-Referenzierung, sondern lediglich ein impliziter Hinweis über die Definition oder Benennung von Konzepten, wie z.B. den Basic Geo-Prädikaten `http://www.w3.org/2003/01/geo/wgs84_pos#latitude` für die geographische Breite und `http://www.w3.org/2003/01/geo/wgs84_pos#longitude` für die geographische Länge (siehe Abschnitt 2.4.1). Stattdessen erlaubt eine fortschrittliche und flexible Geodatenverarbeitung Koordinatenangaben in beliebiger Georeferenzierung, wofür jeder Geometrie respektive jedem Koordinatenwert die explizite Information über das verwendete Koordinatenreferenzsystem angeheftet werden muss. Zu diesem Zweck sind im Geo Web drei verschiedene Varianten von CRS-Bezeichnern gebräuchlich:

- **EPSG-Codes**[107]: weitverbreitete, proprietäre Codes im Sinne einer Industrienorm, die von der *International Association of Oil and Gas Producers* (OGP[108]) gepflegt werden (z.B. WGS84-Code: `EPSG:4326`)
- **OGC-URNs**: das OGC führt seit einigen Jahren eigene URI-Schemata für persistente URN-Namen[109], u.a. existieren URN-Bezeichner für CRS. Dabei spielt es keine Rolle, ob die Systeme vom OGC definiert werden, wie z.B. `urn:ogc:def:crs:ogc:1.3:crs84`, oder von einer externen Stelle stammen, z.B. von der OGP: `urn:ogc:def:crs:epsg::4326`
- **OGC-HTTP-URIs**: seit dem Frühjahr 2010 ist das OGC bemüht, zusätzlich zu den URN-Bezeichnern auch auflösbare HTTP-URIs[110] anzubieten (z.B. `http://www.opengis.net/def/crs/OGC/1.3/CRS84` oder die HTTP-Weiterleitung auf die *EPSG*-Datenbasis: `http://www.opengis.net/def/crs/EPSG/0/4326`)

Nur die zuletztgenannte Variante der OGC-HTTP URIs als auflösbare URIs sind im Umfeld des Semantic Web praktikabel einzusetzen. Hinter den OGC-HTTP URI-Adressen verbergen sich CRS-Definitionen in den Formaten GML, RDF und HTML. Gut vorstellbar, dass das OGC - als die Fachkompetenz bzw. Standardisierungsgremium in der Geo Web-Domäne - den eingeschlagenen Weg fortsetzt und zukünftig ein abgestimmtes und allumfassendes Register für Koordiantenreferenzsysteme führt, welches sowohl ISO-konforme als auch Semantic Web-kompatible Inhalte bereithält. Eine Festlegung des OGC auf OGC-HTTP URIs und deren Empfehlung als CRS-Identifikatoren wäre auch für eine interoperable Kommunikation innerhalb des Geo Web oder hinsichtlich der Entwicklung GeoSPARQL-fähiger SPARQL-Endpoints wünschenswert. Ein erweitertes OGC CRS-Register, basierend auf den jetzigen Ansätzen und unter Verwendung von SKOS-Konzepten, ließe sich zudem vortrefflich mit dem Geo-Vokabular von GeoSPARQL verschmelzen. Für diesen Typus von Referenzinformation ist es also nicht ratsam, das INSPIRE CRS/UoM-Register als primäre Quelle für RDF- bzw. SKOS-Konzepte heranzuziehen. Denn die wenigen INSPIRE-relevanten europäischen CRS[111] fallen nicht ins Gewicht gemessen an den vielen globalen und nationalen CRS, die ein erweitertes OGC CRS-Register harmonisieren und ein damit verknüpftes Geo-Vokabular beinhalten könnte.

5.1.4 Transfer geographischer Information

Der in Abschnitt 2.4.3 vorgestellte *GeoSPARQL*-Entwurf besitzt großes Potential, den Austausch geographischer Informationen im Semantic Web zu standardisieren. Die räumliche Filterfunktionalität ist ein gewichtiger Bestandteil

[107] EPSG-Registrierdatenbank: http://www.epsg-registry.org/
[108] Webauftritt: http://www.ogp.org.uk/
[109] einen Überblick über die Richtliniendokumente der sogenannten *OGC Naming Authority* liefert die Portalseite: http://www.opengeospatial.org/ogc/policies/directives
[110] siehe hierzu das OGC-Whitepaper *OGC Identifiers - the case for http URIs* [Cox 2010]
[111] festgelegt und dokumentiert in der INSPIRE Datenspezifikation *Coordinate Reference Systems* [INSPIRE Thematic Working Group „Coordinate Reference Systems and Geographical Grid Systems“ 2010]

von GeoSPARQL und wird in Abschnitt 5.2.5 diskutiert. Der andere Aspekt ist die Spezifizierung eines *Geo-Vokabular*, d.h. die harmonisierte Speicherung geographischer Merkmale von Semantic Web-Ressourcen. Dieser Abschnitt setzt sich kritisch mit dem in GeoSPARQL definierten Geo-Vokabular auseinander. Zu diesem Zweck wurde untersucht, welche allgemeingültigen, praxisnahen Anforderungen sich hinsichtlich eines Geo-Vokabulars formulieren lassen. Die aus der Untersuchung resultierenden Anforderungen sind unten aufgeführt. Daran schließt sich ein Vergleich an, inwieweit der GeoSPARQL-Entwurf diesen Anforderungen genügt. Zu guter Letzt werden geringfügige Verbesserungen und Erweiterungen für das GeoSPARQL Geo-Vokabular vorgeschlagen.

Alle Anforderungen an das Geo-Vokabular erfolgen vor dem Hintergrund des ontologischen Kompromisses, der im Geospatial Semantic Web Interoperability Experiment [Lieberman 2006] getroffen wird. Lieberman erläutert die Schwierigkeit in der ontologischen Modellierung, ob eine geographische Entität frei von jeglichen geometrischen Charakteristiken definiert wird (z.B. Stadt als politische Größe) oder sogleich mit einen geometrischen Bezug definiert wird (z.B. Stadtgebiet als geometrisches Gebilde Punkt bzw. Polygon). Letztlich führt Lieberman eine logische Trennung zwischen geographischen Entitäten (Stadt im Sinne des ersten Beispiels) und geometrischen Entitäten durch (Umriss der geographischen Entität Stadt) und verknüpft beide Konzepte mit der Assoziation *hasLocation*. Eine vergleichbare Differenzierung in Features (geographische Entitäten) und Geometrien (geometrische Entitäten) ist ein wesentliches Modellierungsmuster aller OGC-Standards, das in den *OGC-Abstract Specifications* spezifiziert wurde und somit auch für den OGC-Entwurf GeoSPARQL bindend ist. An diesem maßgeblichen Modellierungsmuster orientieren sich auch die nun folgenden Anforderungen.

Anforderungen an die geographische Speicherung im Semantic Web

1. **Unterschiedliche Geometrietypen**: die Speicherung von Geometrien in verschiedenen Geometrietypen wird unterstützt, zumindest die von Punktkoordinaten zur schnellen Lokalisierung (insbesondere für das Labeln im Rahmen des Social Tagging) und Bounding Box-Koordinaten, um einfachste rechteckige Objektausdehnungen zu speichern. Die Geometrietypen sind kenntlich zu machen, um die geometrische Komplexität vor dem Lesen der Geometrie abschätzen zu können.

2. **Georeferenzierung der Koordinaten**: jedes Koordinatentupel muss eindeutig einem Koordinatenreferenzsystem bzw. dessen Identifikator zugeordnet werden. Dabei ist auf eine definitionsgemäße Koordinatenreihenfolge zu achten (zur Problematik WGS84 vs. CRS84 siehe Abschnitt 2.4.3) oder explizite semantische Assoziationen einzuführen (z.B. per Prädikate *latitude* und *longitude*).

3. **Fokusierung auf WGS84**: das Semantic Web/ Web2.0 ist mit seiner Fixierung auf geographische WGS84-Koordinaten zu berücksichtigen. Jedes Feature sollte demzufolge Originalgeometrien in WGS84-Koordinaten mitführen oder zumindest generalisierte Geometrien im CRS WGS84 enthalten (z.B. als Zentroidpunkt oder als Bounding Box).

4. **Unterstützung zur Geometrieverarbeitung**: die geometrische Speicherform ist derart zu wählen, dass sowohl dedizierte GIS-Funktionen unterstützt werden (zur Prüfung topologischer Beziehungen, Kalkulationen der konvexen Hülle, geometrischer Schnittmengen etc.) als auch herkömmliche semantische Auswertungen zur Rückwärtskompabilität mit gebräuchlichen Geometriespeicherformen möglich sind (z.B. die SPARQL-Filterung mit numerischen Vergleichen von Koordinatenwerten, die wie im Geo Basic-Vokabular einzeln über Prädikate gespeichert sind).

5. **Verwendung topologischer und geometrischer Assoziationen**: ermöglicht werden sollte das explizite Abspeichern topologischer Beziehungen zwischen Features (z.B. Enthaltensein, Überschneiden) und wünschenswerterweise auch von Beziehungen zwischen Geometrien (z.B. Generalisierung, Verweis auf Originalgeometrie).

6. **Kompakte Geometriekodierung**: die Geometriekodierung sollte kompakt und ohne inhaltliche Redundanzen sein, um einen performanten Zugriff und Transfer im Internet sicherzustellen.
7. **Standardkonforme Geometriespeicherung**: für die Geometrieformatierung sind nach Möglichkeit etablierte Standards zu verwenden, so dass gängige GIS-Programme bzw. Visualisierungskomponenten die transferierten Geometrien direkt nachnutzen können.
8. **Genauigkeitsangabe**: die geometrische Genauigkeit sollte zumindest auf einfachste Weise dokumentiert werden können, z.B. per Erfassungsmaßstab oder mit der Angabe der geometrischen Genauigkeit. Dadurch lässt sich u.a. ein optimaler Visualisierungbereich vorgeben. Mit Hilfe der Angaben könnten auch GIS-Funktionen ihre Berechnungen in Abhängigkeit zur geometrischen Genauigkeit durchführen und eine zweckmäßige numerische Präzision anwenden.

Vergleich von GeoSPARQL mit den aufgestellten Anforderungen für Geo-Vokabulare

zu 1. **Unterschiedliche Geometrietypen**: ***erfüllt***
GeoSPARQL unterstützt verschiedenste Geometrietypen. Dabei beinhaltet GeoSPARQL kein eigenständiges Geometrietypenmodell, sondern bezieht die Geometrietypen aus den adressierten Standards zur Geometrieformatierung, im Speziellen also jene aus den Standards GML und Simple Features (WKT). Das verhindert eine Überspezifizierung in GeoSPARQL und verlagert die Aufgabe, Geometrieformate zu harmonisieren bzw. ineinander umzuwandeln, auf zweckmäßige Weise in die Programmlogik von GeoSPARQL-Implementierungen. Punkt- und Bounding Box-Koordinaten lassen sich mit Hilfe der gängigen Geometrietypen von GML (`Point` bzw. `Envelope`) und WKT ausdrücken (`Point` bzw. `Polygon`). Ebenso ist die Forderung nach Kenntlichmachung der Geomtrietypen erfüllt: sie sind aus den Geometriekodierungen von GML und Simple Features (WKT) herauszulesen.

zu 2. **Georeferenzierung der Koordinaten**: ***erfüllt***
GeoSPARQL schreibt vor, dass jegliche Geometriekodierungen entsprechende Geoereferenzierungshinweise (CRS-Angabe in Form einer URI) enthalten, entweder bereits unterstützt vom originären Geometrieformat (vgl. GML-Geometrien) oder ergänzt durch einen, der Geometriekodierung vorangestellten CRS-Identifikator (vgl. WKT-Geometrien, siehe auch Abschnitt 2.4.3). Die Behandlung der Georeferenzierung ist in GeoSPARQL also konsequent durchdacht.

zu 3. **Fokusierung auf WGS84**: ***partiell erfüllt***
GeoSPARQL erklärt das CRS84 zum Standardkoordinatensystem. Darüber hinaus sind auch Geometrien anderer Georeferenzierung gestattet, die kenntlich gemacht werden müssen (siehe Punkt 2). Die Anfoderung, dass Geometrien auch stets im CRS WGS84 und in geographischen Koordinaten vorliegen, wird dadurch nicht erfüllt. Deshalb wird am Ende des Abschnittes ein Erweiterungsvorschlag gemacht.

zu 4. **Unterstützung zur Geometrieverarbeitung**: ***partiell erfüllt***
GeoSPARQL erlaubt eine umfangreiche GIS-Filterfunktionalität. Aufgrund der definierten Speicherformen werden GeoSPARQL-fähige SPARQL-Endpoints vorausgesetzt. Eine Rückwärtskompabilität zu herkömmlichen SPARQL-Endpoints ist leider nicht bezweckt. Ein Verbesserungsvorschlag folgt hierzu.

zu 5. **Verwendung topologischer und geometrischer Assoziationen**: ***partiell erfüllt***
GeoSPARQL unterstützt topologische Beziehungen, lässt aber geometrische außer Acht. Ein Erweiterungsvorschlag folgt hierzu.

zu 6. **Kompakte Geometriekodierung**: ***erfüllt***
GeoSPARQL erreicht durch die Speicherung in RDF-Literalen eine optimale Komprimierung.

zu 7. **Standard-konforme Geometriespeicherung**: ***erfüllt***
GeoSPARQL orientiert sich an den gängigen Geodatenstandards (z.B. GML und Simple Features).

zu 8. **Genauigkeitsangabe**: ***nicht erfüllt***
GeoSPARQL führt keine derartigen Prädikate mit. Ein Vorschlag zur Erweiterung folgt.

Verbesserungs-/ Erweiterungsvorschläge

Die hier aufgeführten Vorschläge sind als konstruktive Kritik gedacht und wurden innerhalb des GeoSPARQL-Reviews mitsamt den Vorüberlegungen als Kommentare beim OGC eingebracht (siehe Anhang F). Für eine bessere Übersichtlichkeit und zum Nachvollziehen der weiteren Ausführungen dient das Diagramm 5.4. Gegenüber dem Diagramm 2.8 aus dem Grundlagenabschnitt über GeoSPARQL sind in Diagramm 5.4 alle Anpassungen der folgenden Verbesserungsvorschläge eingegangen und gekennzeichnet (siehe Stereotypen *Review: added*, *Review: reduced*, *Review: replaced*).

- zu 3. und 4. Die beiden Anforderungen 3 und 4 sind von der GeoSPARQL-Spezifizierung nur bedingt erfüllt. Danach bleibt zu beanstanden, dass GeoSPARQL-Geometrien nicht immer auch in WGS84-Koordinaten gehalten sind und herkömmliche Auswerteroutinen keine Chance erhalten, GeoSPARQL-Geometriespeicherungen zu interpretieren. Letztere Forderung ist im Hinblick auf Rückwärtskompabilität zu SPARQL-Prozessoren ohne GeoSPARQL-Filtererweiterung formuliert. Einfache SPARQL-Prozessoren werden üblicherweise dazu eingesetzt, über Prädikate referenzierte und getrennte Koordinatenwerte auszulesen und zu vergleichen, nicht jedoch die neuen in GeoSPARQL geforderten komplexen Geometriekodierungen, eingelagert in Literalen. Eine Lösung ist die Einführung verpflichtender **Prädikate nach dem Modell des Basic Geo-Vokabulars**, die semantisch eindeutig definiert sind und die Georeferenzierung auf das WGS84 einschränken. Ausgehend von der OWL-Klasse `geo:Geometry` sind dies Prädikate zur Speicherung punktueller Referenzlokalisierungen (`geo:refLocation_lat` und `geo:refLocation_long`) und jene zur Speicherung von Bounding Box-Koordinaten (`geo:bbox_minLat`, `geo:bbox_minLong`, `geo:bbox_maxLat` und `geo:bbox_maxLong`). Aus komplexen Geometrien der GeoSPARQL-Literale (z.B. vom Typ `Polygon` oder `TriangulatedSurface`), lassen sich die benötigten einfachen Koordinatenwerte ableiten, womit der Rückwärtskompabilität und WGS84-Konformität Genüge getan ist.
- zu 5. Konzeptionell bestehen keine Hürden für die Aufnahme binärer Assoziationen der Klasse *geo:Geometry*, so dass auch fehlende **geometrische Assoziationen** aufgenommen werden können. Letztlich helfen sie die Ausdrucksfähigkeit des Geo-Vokabulars zu steigern, das ohnehin vor einem Reifeprozess steht und Potential für Verfeinerungen aufweist. Vorgeschlagen werden optionale Assoziationen zum Analysieren von geometrischen Generalisierungsstufen: `geo:generalizes` und `geo:generalizedBy`. Ein weiterer Vorschlag besteht darin, das Prädikat `geo:hasGeometry` mit dem Prädikat `geo:hasOriginalGeometry` zu spezialisieren, um damit den Bezug zum Primärdatenbestand bzw. zur detailreichsten, originären Geometriespeicherung auszudrücken. Abgesehen von diesen Erweiterungen beinhaltet GeoSPARQL eine Menge an binären Assoziationen in Form von topologischen Prädikaten, deren Bedeutungen den topologischen Relationen von Simple Features, Egenhofer und RCC8 entnommen sind (siehe Abbildung 2.8). Wie der Abschnitt 2.3.3 offenlegt, gleichen sich viele dieser Relationen. Ebenso ist es ratsam, zwar einen ausdrucksstarken aber begrenzten Relationensatz im Semantic Web einzuführen, damit keine unnötige Prädikatenvielfalt hervorgeht und Prädikate häufiger (nach-) genutzt werden. Deshalb wird empfohlen, nur die Simple Features-Relationen beizubehalten.
- zu 8. GeoSPARQL definiert keine Prädikate zur Angabe der geometrischen Genauigkeit. Deshalb wird ein Prädikat `geo:captureResolution` als räumliches Attribut der OWL-Klasse `geo:Geometry` vorgeschlagen, das Auskunft über den jeweiligen geometrischen Erfassungsmaßstab geben soll. In diesem Zusammenhang ist festzustellen, dass manche der in GeoSPARQL definierten räumlichen Attribute kaum von Nutzen sind. So drückt z.B. das Prädikat `geo:isEmpty` (Wertebereich: `true | false`) aus, ob die Geometrie einer leeren (Punkt-) Menge gleichkommt. Für Geodatenbanken, deren Tabellen feste und möglicherweise leere Geometriefelder (`NULL`-Werte)

aufweisen, mag das Sinn ergeben, nicht aber im Kontext des Semantic Web, wo die Angabe zur gesamten Geometrie über das Prädikat `geo:hasGeometry` schlichtweg entfallen würde. Desweiteren steckt die Aussage des Prädikats `geo:is3D` bereits im numerischen Wert des Prädikats `coordinateDimension` und ist deshalb ohne Informationsmehrwert. Das GeoSPARQL-Spezifizierungsteam sollte kritisch hinterfragen, ob alle aus der Simple Features-Spezifikation entnommenen räumlichen Attribute auch auf das Semantic Web einwandfrei angewendet werden können.

Es zeigt sich also, dass der GeoSPARQL-Entwurf bereits sehr ausgereift ist und viele Anforderungen erfüllt. Dennoch drängt sich die Frage auf, ob die weitreichende Unterteilung in verschiedene Konformitäts- und Parameterklassen (siehe Abschnitt 2.4.3) nicht dazu führt, dass der kombinierte GeoSPARQL-Ansatz eines Geo-Vokabulars und der dazugehörigen Filterfunktionalität als zu komplex abgelehnt wird. Beispielsweise ist der Gebrauch der Prädikate `geo:asGML` und `geo:asWKT` in SPARQL-Graphenmustern wenig komfortabel. So wird sich ein Endanwender vermutlich nicht um die Geometrieformatierung (GML oder WKT) kümmern wollen, sondern ein einheitliches Prädikat `geo:geom` oder `geo:geomLiteral` bevorzugen. Außerdem enthält der Entwurf noch den logischen Widerspruch, dass die räumlichen Filterfunktionen gemäß ihrer Konformitätsklassen nach Formatierungsversionen unterscheiden (z.B. GML 2.1, GML 3.2.1), demnach müsste das Geo-Vokabular anstelle des Prädikates `geo:asGML` in die Prädikate `geo:asGML210` und `geo:asGML321` differenzieren. Daraus lässt sich das Fazit ziehen, dass sich zuviel Freiheit in der Wahl der Geometrieformatierung kontraproduktiv erweisen kann. Es wäre sicherlich ratsam, weitere Konformitätsklassen (z.B. für GeoJSON, KML) kategorisch auszuschließen, damit die Liste an Konformitätsklassen begrenzt bliebe und GeoSPARQL-Implementierungen jene weitreichend unterstützen könnten. So wäre eine Systemlandschaft aus ähnlich funktionalen GeoSPARQL-Schnittstellen sicherlich auch im Sinne des Endanwenders.

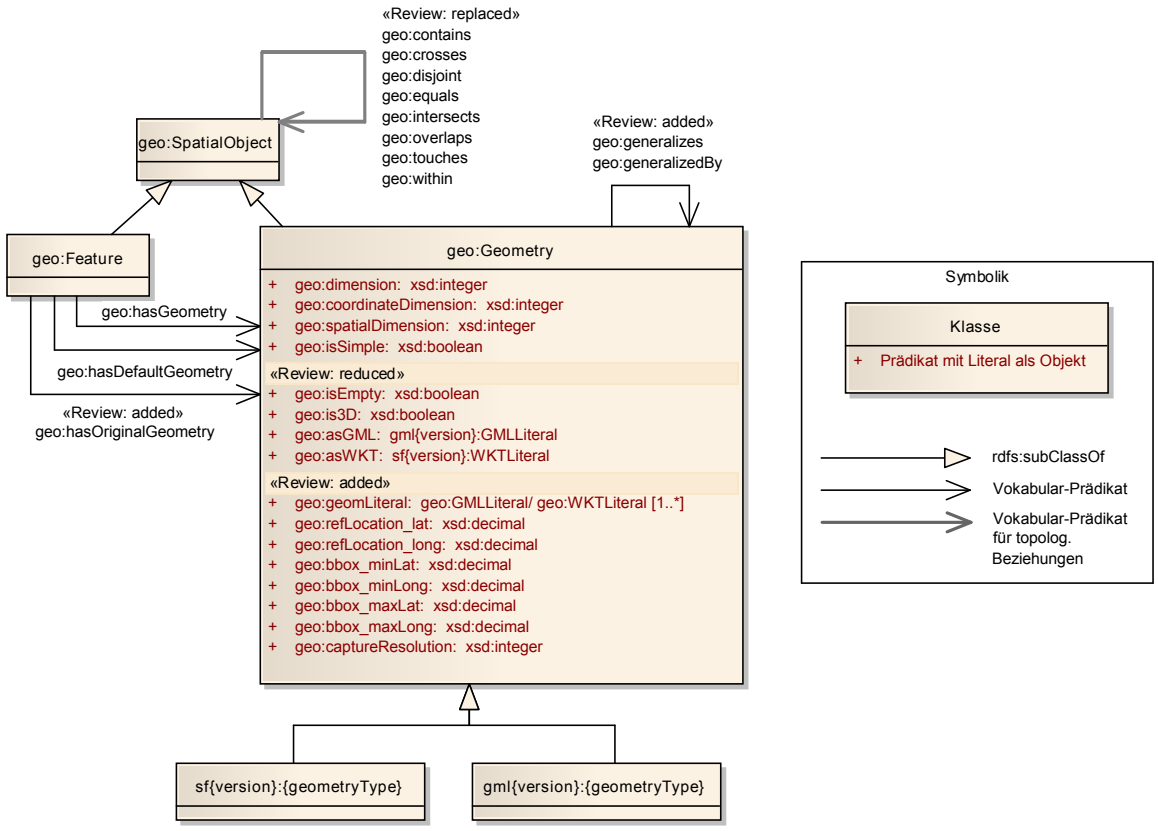


Abbildung 5.4 – GeoSPARQL - Vokabular-Review

5.1.5 Abbildungsregeln auf INSPIRE GML-Applikationsschemata

In diesem Abschnitt werden Abbildungsregeln vorgestellt, die die INSPIRE GML-Applikationsschemata mit den INSPIRE-Themenontologien in Beziehung setzen. Sie bilden die Grundlage für eine Datentransformation von GML-Instanzen in RDF/OWL-Entsprechungen, die entweder losgelöst manuell oder im Rahmen einer Anfragearchitektur (siehe Abschnitt 5.2) ausgeführt werden kann. Um die Abbildungsregeln zu erfassen und zu speichern, sind keine externen Transformationsskripte notwendig. Stattdessen werden die Regeln in die neu gebildeten Themenontologien aufgenommen, und zwar über sogenannte *OWL-Annotationsprädikate* (engl. *OWL-Annotation Properties*). Darunter versteht man Prädikate, die zwecks Charakterisierung an ein Ressource annotiert bzw. angeheftet werden, ohne dass die damit gespeicherten Fakten als gültiger Teil der Wissensbasis verstanden und folglich auch nicht zur Inferenzbildungen herangezogen werden (siehe OWL-Spezifikation [Hitzler et al. 2009]). Anstatt die ontologische Ausdrucksfähigkeit zu steigern, ergänzen sie die Konzepte lediglich mit Metainformationen wie Alternativnamen (Prädikat: `rdfs:label`), Definitionstexten (`skos:definition`) oder Autorenhinweise (`cd:creator`). Im Sinne von Konzeptmetadaten werden hier nun einige Annotationsprädikate speziell für die Abbildungsregeln definiert. Sie gehören exemplarisch dem Namensraum `http://inspire.west.uni-koblenz.de/configuration/` (ontologischer Präfix: `cf`) an. Die Abbildungsregeln bzw. ihre Annotierungen lassen sich unterteilen in:

1. **Regeln auf Ontologie-Ebene**: zur Verlinkung mit themengleichen GML-Schemata
2. **Regeln für OWL-Klassen**: u.a. zur Abbildung auf korrespondierende GML-Objekttypen
3. **Regeln für OWL-Prädikate**: u.a. zur Abbildung auf korrespondierende GML-Propertytypen

Noch ein kleiner Hinweis vorweg:
Der Abschnitt beinhaltet viele Beispiele und Abbildungen. Damit darin keine Verwechslung zwischen GML- und ontologischen Namensräumen auftreten kann, sind die zugehörigen Präfixe klar voneinander unterschieden: Präfixe von ontologischen Namensräumen beginnen im Gegensatz zu GML XML-Namensräumen mit einem `ont`, z.B. `ont_ps` als Abkürzung für `http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/`.

Abbildungsregeln auf Ontologie-Ebene

Der Lösungsansatz in Abschnitt 4.1 erläutert, dass ein oder mehrere GML-Applikationsschemata auf eine Themenontologie abgebildet werden, je nachdem, wie viele GML-Schemata das jeweilige INSPIRE Annex-Thema umfasst. Damit eine zweifelsfreie **Zuordnung der GML-Schemata zu den Ontologien** möglich wird, erfolgt die Annotierung der Themenontologie mit Verweisen auf die XML-Namensräume der zugeordneten GML-Schemata. Es wird also die Ressource der Themenontologie bzw. das Kopfelement des Ontologiedokumentes annotiert, das z.B. identifiziert wird mit der URI `http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/` (siehe Definition der URI-Syntax in Abschnitt 5.1.2). Zudem soll jeder annotierte XML-Namensraum mit einem XML-Präfix abgekürzt werden, so dass sich XML-Ausdrücke nachfolgender Annotationen darauf beziehen können (siehe Regeln für OWL-Klassen und -Prädikate). Beide Informationen, XML-Namensraum und abkürzendes XML-Präfix, bilden den Wert des Annotationsprädikates:

- **cf:xmlNamespace**: Deklaration des behandelten GML XML-Namensraumes, Mehrfachannotation sinnvoll, Beispielswerte sind `xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"` oder `xmlns:gml="http://www.opengis.net/gml/3.2"`

Abbildungsregeln für OWL-Klassen

Die OWL-Klassen der Themenontologien sind den INSPIRE GML-Objekttypen gleichzusetzen. Die Namen der GML-Objekttypen werden darum an die OWL-Klassen annotiert und um die Information der zugehörigen UML-Stereotypen ergänzt. Sofern der korrespondierende GML-Objekttyp vom Stereotyp eines `featureType` oder `dataType` ist, wird die OWL-Klasse zusätzlich mit der anzuwendenden Syntax für die Generierung von URIs augestattet (eine nähere Erläuterung folgt weiter unten). Die Information über den Stereotyp wird insbesondere benötigt, um OWL-Klassen zu detektieren, die einem Feature Type entsprechen und damit die Zugriffsquelle in WFS GetFeature-Aufrufen darstellen.

Die neudefinierten Annotationsprädikate sind:

- **cf:xmlName**: Qualifizierender XML-Name des entsprechenden GML-Objekttyps, Mehrfachannotation sinnvoll, z.B. `ps:ProtectedSite` und `ps-f:ProtectedSite`, `au:AdministrativeBoundary`
- **cf:entityType**: zugehöriger INSPIRE UML-Stereotyp des GML-Objekttyps, z.B. `featureType`, `dataType` oder `codelist`
- **cf:uriSyntax**: URI-Syntax als Anleitung zur automatisierten URI-Generierung, unten folgt dazu ein Beispiel

In Abbildung 5.5 ist das Konzept `ont_ps:ProtectedSite` mit vier angehängten Annotationsprädikaten zu sehen. Die `cf:xmlName`-Annotationsprädikate drücken aus, dass das Konzept `ont_ps:ProtectedSite` auf die GML-Objekttypen `ps:ProtectedSite` und `ps-f:ProtectedSite` abgebildet wird.[112] Zweitens wird der INSPIRE-Stereotyp der genannten GML-Objekttypen über das Prädikat `cf:entityType` annotiert. Drittens wird per `cf:uriSyntax` konfiguriert, wie die URI einer Instanz von `ont_ps:ProtectedSite` lauten soll.

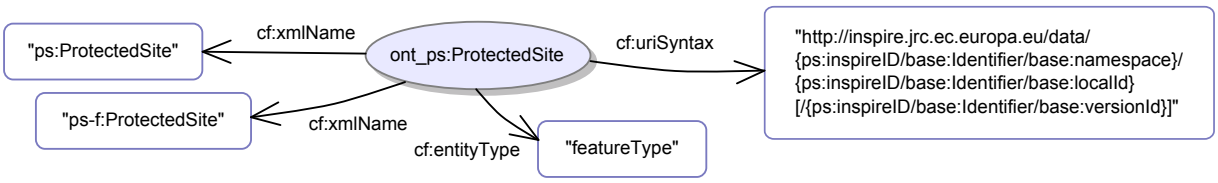


Abbildung 5.5 – Annotierung von OWL-Klassen mit GML/OWL-Abbildungsregeln

Die URI-Bestandteile sind entweder statische Textbausteine oder leiten sich dynamisch aus GML-Elementen und -Attributen ab. Der Abschnitt 5.1.2 *Identifikationsmanagement* legt u.a. die URI-Syntax für unabhängige OWL-Instanzen wie folgt fest:

`http://inspire.jrc.ec.europa.eu/`*`<format>`*`/`*`<namespace>`*`/`*`<localId>`*`[/`*`<versionId>`*`]`

Die Variable `format` kann statisch mit dem Wert `data` vorbelegt werden, der auf eine Informationsressource hinweist (siehe Abschnitt 2.2.4). Die restlichen drei Variablen werden bei der Umformung einer GML- in eine OWL-Instanz bzw. während der **URI-Generierung** der OWL-Instanz aufgelöst und mit konkreten Werten belegt. Sie leiten sich aus der GML-Instanz bzw. deren individuellen Unterelementen ab. Um die Zielpfade ausgehend von der GML-Instanz (-Feature) auf ihre Unterelemente zu bestimmen, werden XPath-Ausdrücke anstelle der Variablennamen konfiguriert:

`http://inspire.jrc.ec.europa.eu/data/{`*`ps:inspireID/base:Identifier/base:namespace`*`}/`
`{`*`ps:inspireID/base:Identifier/base:localId`*`}[/{`*`ps:inspireID/base:Identifier/base:versionId`*`}]`

[112] der Unterschied beider INSPIRE GML-Objekttypen liegt in der detaillierteren Attributierung von `ps-f:ProtectedSite` als Element des Protected Sites Full-Schemas, siehe Abschnitt 2.5.2

Gesetzt dem Fall, dass die Elementpfade zum jeweiligen INSPIRE-Objektidentifikator in allen INSPIRE-Datenspezifikationen vereinheitlicht wären, könnte die hier vorgegebene URI-Syntax in eine Transformationsroutine fest einprogrammiert werden. Leider aber sind die Datenspezifikationen der Annex-I-Themen in dieser Hinsicht nicht harmonisiert und unterscheiden sich im Namen der GML-Property, die den INSPIRE-Basistyp `base:Identifier` referenziert. Genauer gesehen divergieren sowohl die GML-Property-Namen als auch ihre XML-Namensräume: z.B. `ps:INSPIREID`, `ps-f:objectIdentifier` oder `au:INSPIREId`. Aufgrund der Variation muss auch die Konfiguration flexibel bleiben, was mit dem Annotationsprädikat `cf:uriSyntax` bezweckt wird. Damit sich die URI-Syntax von einer Transformationsroutine einlesen und anwenden lässt, sind **syntaktische Regeln** zu definieren, z.B. in der Backus-Naur-Form (BNF):

```
1  <left square bracket> ::= [
2  <right square bracket> ::= ]
3  <left brace> ::= {
4  <right brace> ::= }
5  <static part> ::= <alpha-numeric digits>
6  <variable part> ::= <left brace> <xpath> <right brace>
7  <hash content placeholder> ::= <left brace> 'hashContent' <right brace>
8  <any part> ::= <static part> | <variable part> | <hash content placeholder>
9  <optional part> ::= <left square bracket> <any part> <right square bracket>
10 <any alternative part> ::= <any part> | <optional part>
11 <uri syntax sequence> ::= <any alternative part> | <any alternative part> <uri syntax sequence>
```

Codebeispiel 14 – Syntaxregeln zur URI-Generierung von OWL-Instanzen (Backus-Naur-Form)

Abbildungsregeln für OWL-Prädikate

Die OWL-Prädikate der Themenontologien sind den INSPIRE GML-Properties gleichzusetzen. OWL-Prädikate sind das Bindeglied in einem Subjekt-Prädikat-Objekt Tripel, während GML-Properties ein vergleichbares Bindeglied innerhalb einer Subjekt-Prädikat-Objekt XML-Elementsequenz darstellen. Die Verwendung eines OWL-Prädikats wird eingeschränkt, indem dessen Definitionsbereich auf Subjekttypen (Definitionsprädikat `rdfs:domain` verweist auf OWL-Klassen) und dessen Wertebereich auf Objekttypen festgelegt wird (Definitionsprädikat `rdfs:range` verweist auf OWL-Klassen oder -Literale). Hingegen wird eine GML-Property exklusiv als Subelement eines Subjekttypen (XML-ComplexType) spezifiziert, um den Subjekttypen mit einem Objekttypen zu verbinden (XML-ComplexType oder -SimpleType). Die grundlegende Idee besteht nun darin, die schematische Dreierbeziehung aus 1. OWL-Prädikat, 2. Definitionsbereich und 3. Wertebereich in eine **GML-Pfadangabe mittels XPath-Ausdruck** zu übersetzen. Im ersten XPath-Pfadschritt der GML-Pfadangabe wird der Subjekttyp bzw. der XML-ComplexType genannt, in dessen eindeutigen Kontext die GML-Property gültig ist (vgl. Definitionsbereich des OWL-Prädikats). Danach folgt der davon ausgehende GML-Elementpfad zum Objekttypen, dessen XML-Name den letzten XPath-Pfadschritt bildet (vgl. Wertebereich des OWL-Prädikats). Die hier zum Einsatz kommenden XPath-Ausdrücke sollen konform zur XPath-Version 1.0[113] sein und nur solche XPath-Konstrukte beinhalten, die sich zur eindeutigen Referenzierung von GML-Elementtypen eignen. Deshalb seien Wildcards für Knotentests (*) und jene für Prädikatfilterungen (@*) von der Verwendung ausgeschlossen.

Die Abbildung 5.6 veranschaulicht anhand des OWL-Prädikats `ont_ps:activitiesAndImpacts` die angesprochene schematische Definition mittels `rdfs:domain` und `rdfs:range`. Zusätzlich zu den bereits eingeführten `cf:xmlName`-Annotationen wird dem OWL-Prädikat die erwähnte GML-Pfadangabe über das Annotationsprädikat `cf:xpath` hinterlegt. Zugehörige GML/OWL-Abbildungsinformationen werden in Abbildung 5.6 gleichfarbig dargestellt. Dank der `cf:xpath`-Annotation lassen sich nun Definitions- und Wertebereiche nicht nur auf RDF/OWL-, sondern auch auf GML-Schemaebene bestimmen, da die Verweise auf Subjekt- und Objekttypen implizit im GML-Elementpfad

[113] Spezifikation: http://www.w3.org/TR/xpath/

von `cf:xpath` enthalten sind (erster und letzter XPath-Pfadschritt). Darüber hinaus erlaubt die Methodik, die flexiblen Objektrelationen in RDF/OWL auf GML anzuwenden: vergleichbar mit RDF-Tripeln, die sich zu einem Graphenmuster ergänzen, können die mit `cf:xpath` konfigurierten relativen GML-Elementpfade zu absoluten GML-Elementpfaden zusammengestellt, d.h. über XML-Namensvergleiche miteinander verknüpft werden (siehe hierzu Abschnitt 5.2.1 *Auswertung von SPARQL-Anfragen*).

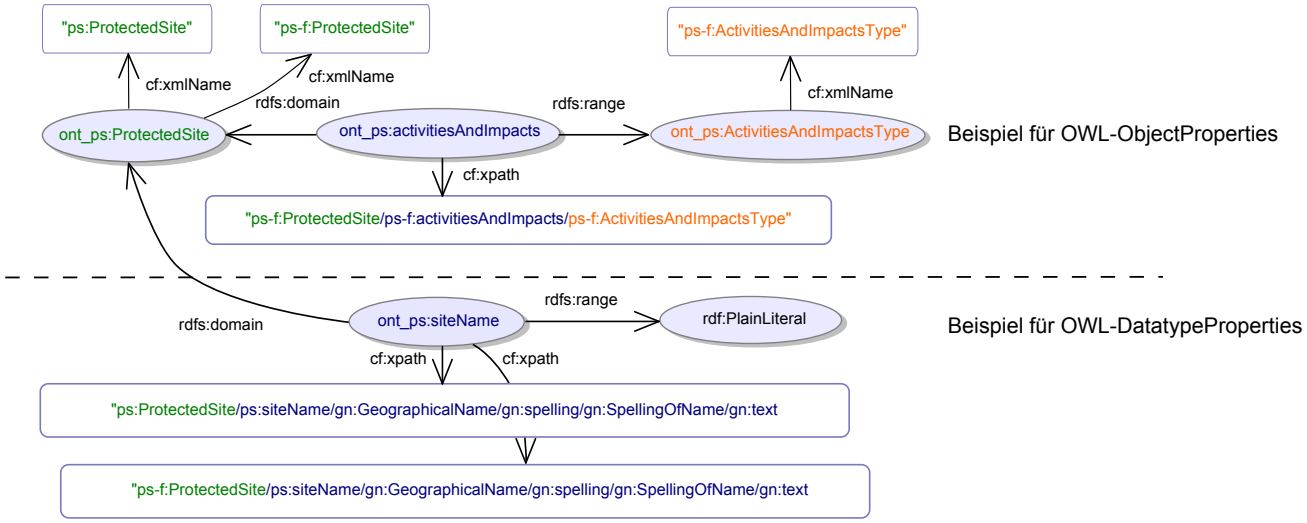


Abbildung 5.6 – Annotierung von OWL-Prädikaten mit GML/OWL-Abbildungsregeln

Für die Modellierung der Themenontologien wird die Strategie verfolgt, gleichnamige GML-Properties in ein und dasselbe OWL-Prädikat zu überführen (siehe Vorgehensweise und Erläuterung in Abschnitt 5.1.1). Die umgekehrte Abbildung von OWL auf GML sorgt demnach in manchen Fällen für **Mehrdeutigkeiten**: einem OWL-Prädikat sind mehrere gleichnamige GML-Properties zuzuordnen, die sich in ihrer Abhängigkeit zum jeweils übergeordneten XML-ComplexType unterscheiden. Ein simple Lösung sieht vor, ein OWL-Prädikat für mehr als eine GML-Property, d.h. mit mehreren `cf:xpath`-Annotationen zu konfigurieren. Aufgrund der **Mehrfachannotationen mit `cf:xpath`** werden andererseits Definitions- und Wertebereiche des OWL-Prädikats mehrdeutig: die Abhängigkeit zu einem GML-Elementpfad geht verloren. Um die Abhängigkeit zu erhalten, werden sogenannte Axiom-Annotationen eingesetzt, die das Annotieren von Axiomen bzw. Statements ermöglichen (Stichwort *Reifikation*). Es werden zwei Axiom-Annotationsprädikate namens `cf:rdfsDomain` und `cf:rdfsRange` eingeführt, die anstelle der Definitionsprädikate `rdfs:domain` und `rdfs:range` die Definitions- und Wertebereiche der OWL-Prädikate in Abhängigkeit zum jeweiligen GML-Elementpfad vorgeben. Die Abbildung 5.7 zeigt die neueingeführten Axiom-Annotationsprädikate am Beispiel des Prädikats `ont_ps:siteName`. Dessen `cf:xpath`-Annotation bildet ein Tripel aus Subjekt-Prädikat-Objekt, an das die Axiom-Annotationsprädikate angehängt werden sollen. Um das Tripel bzw. die damit verbundene Aussage (Axiom) zu identifizieren, verwendet OWL die hier orange eingefärbten Prädikate `owl:annotatedSource`, `owl:annotatedProperty` und `owl:annotatedTarget` ausgehend von einem Blank Node. Dem Blank Node, quasi der Repräsentation des Axioms, werden alle Axiom-Annotationen angefügt. Im Beispiel sind dies `cf:rdfsDomain`, `cf:rdfsRange` und `cf:langXPath` (Erklärung zu Letzterem folgt).

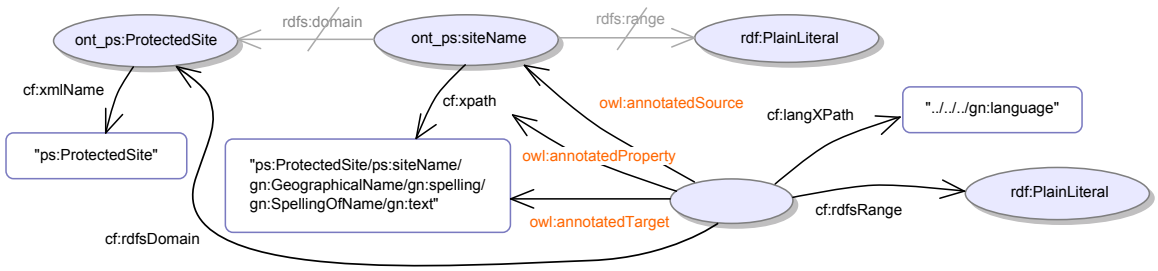


Abbildung 5.7 – Axiom-Annotationen zur Festlegung eindeutiger Definitions- und Wertebereiche für OWL-Prädikate

Ein GML-Elementpfad verweist auf bestimmte Speicherstellen in einem GML-Instanzdokument. Das können GML-Objektinstanzen oder textuelle Elementinhalte, wie z.B. Textwerte, Datumswerte oder numerische Werte, sein. Sofern der GML-Elementpfad einer `cf:xpath`-Annotation auf textuelle Elementinhalte referenziert, können der `cf:xpath`-Annotation weitere wertvolle Transformationshinweise angefügt werden. So wird ein Text durch die Sprache, in der er gehalten ist, charakterisiert. Ebenso ist zum Interpretieren eines Messwertes das Wissen über seine Maßeinheit erforderlich. Derart **abhängige Größen**, wie die Sprachangabe oder die Maßeinheit, sind in einem GML-Instanzdokument relativ zur Speicherstelle des Text- bzw. Messwertes angegeben. Um die abhängigen Größen zusätzlich zum textuellen Inhalt aus dem GML-Dokument herauszulesen und in ein RDF/OWL-Format zu transformieren, bedarf es der Formulierung weiterer relativer XPath-Ausdrücke in Abhängigkeit zur Speicherstelle, die von `cf:xpath` adressiert wird. Zu diesem Zweck werden Axiom-Annotationen für Sprachangaben (`cf:langXPath`) und Maßeinheiten definiert (`cf:uomXPath`, *uom*: *Unit of Measure*). Abbildung 5.7 zeigt ein praktisches Beispiel für `cf:langXPath`.

Das für OWL-Prädikate neudefinierte Annotationsprädikat ist:

- **cf:xpath**: Relativer XPath-Ausdruck, GML-Elementpfad vom übergeordneten XML-ComplexType einer GML-Property bis zum referenzierten Objekttypen (XML-ComplexType oder -SimpleType), Mehrfachannotation sinnvoll, z.B. `ps-f:ProtectedSite/ps-f:activitiesAndImpacts/ps-f:ActivitiesAndImpactsType`

Die neudefinierten und von `cf:xpath` abhängigen Axiom-Annotationen sind:

- **cf:rdfsDomain**: Einschränkung des Definitionsbereichs eines OWL-Prädikats abhängig vom GML-Elementpfad
- **cf:rdfsRange**: Einschränkung des Wertebereichs eines OWL-Prädikats abhängig vom GML-Elementpfad
- **cf:uomXPath**: Relativer XPath-Ausdruck zum GML-Elementpfad, um im Falle von Messwerten als Objektinformation die zugehörige Maßeinheit parsen und in eine RDF/OWL-Entsprechung umwandeln zu können
- **cf:langXPath**: Relativer XPath-Ausdruck zum GML-Elementpfad, um im Falle von textuellen Objektinformationen das angegebene Sprachkürzel parsen und in eine RDF/OWL-Entsprechung umwandeln zu können

Anhand des Codebeispiels 15 soll eine praktische Transformation von GML nach RDF/OWL veranschaulicht werden. Es zeigt ein GML-Fragment, das mit Hilfe der Abbildungsregeln aus Abbildung 5.7 zu parsen ist.

```
<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>
```

Codebeispiel 15 – Beispiel einer INSPIRE-Objektbenennung durch das Element `gn:GeographicalName`

Wird das GML-Fragment aus Codebeispiel 15 mit Hilfe der Abbildungsregeln aus Abbildung 5.7 geparst, geht daraus ein kurzes RDF-Statement hervor[114]: `ex:ProtectedSite1 ps:siteName "Kleinkinzig- und Rötenbachtal"@de`

Vorzüge des Gebrauchs von XPath-Ausdrücken

Abschließend seien einige Vorteile genannt, die sich aus der Konfiguration mit `cf:xpath`-Annotationen und den darin verwendeten XPath-Ausdrücken ergeben:

- **Strukturverflachung bzw. Inhaltsverdichtung**: anhand der Abbildungen 5.6 und 5.7 wird ersichtlich, das OWL-Prädikate aufgrund der ihnen zugewiesenen `cf:xpath`-Annotationen beliebig lange GML-Elementpfade referenzieren können. Bei der Ontologiemodellierung kann deshalb - sofern gewünscht - auf eine 1:1 Abbildung von GML-Properties auf OWL-Prädikate bzw. GML-Objekttypen auf OWL-Klassen verzichtet werden. Eine unstete Abbildung führt zu einer Strukturverflachung der resultierenden Ontologierelationen und zur Inhaltsverdichtung der transformierten Instanzdaten. Dies ermöglicht eine Übersetzung von inhaltsarmen und stark verschachtelten GML-Elementstrukturen in weniger komplexe Ontologien.
- **Konsolidierung von Modellstrukturen**: einige komplexe ISO- und GML-Datentypen bieten alternative Unterelemente zur Speicherung des gleichen Sachverhaltes an. Beispielsweise kann das Speicherelement, das gemäß der GML-Spezifikation den Beginn einer Zeitepoche (Elementtyp `gml:TimeOrdinalEra`) beschreibt, identifiziert werden über den Elementpfad `gml:extent/gml:TimePeriod/gml:beginPosition` oder den Pfad `gml:start/gml:TimeNode/gml:position/gml:TimeInstant/gml:timePosition`. Der Facettenreichtum ist mit Sicherheit hinderlich für Anfragen und das Parsen von Ergebnissen. Er lässt sich reduzieren, indem in den beschriebenen Fällen nur ein OWL-Prädikat gebildet (z.B. `ont_ps:beginPosition`) und mit zwei alternativen `cf:xpath`-Anweisungen annotiert wird.
- **Filterung von Elementpfaden**: in XPath-Ausdrücken können Knotentests enthalten sein, die einen GML-Elementpfad abhängig zu Pfadkriterien, wie z.B. Wertebelegungen oder Elementpositionen, spezifizieren. Die damit einhergende inhaltliche Filterung macht es möglich, spezifische OWL-Prädikate anzulegen, die nur einen Teilaspekt abdecken. Beispielsweise wird über ein OWL-Prädikat mit zugehöriger `cf:xpath`-Annotation `gmd:CI_Citation/gmd:date/gmd:CI_Date[gmd:dateType/gmd:CI_DateTypeCode/@codeListValue = "creation"]/gmd:date/gco:Date` ein Erstellungsdatum (engl. *creation date*) beschrieben, während ein weiteres OWL-Prädikat mit einer modifizierten Annotation (`@codeListValue = "publication"`) ein Veröffentlichungsdatum (engl. *date of publication*) ausdrückt.

[114] die Syntax zur URI-Generierung von OWL-Instanzen (siehe Abschnitt 5.1.2) wurde im Sinne eines kompakten Beispiels ignoriert. Die OWL-Instanz lautet hier stattdessen exemplarisch `ex:ProtectedSite1`

Damit sind alle erforderlichen Abbildungsregeln diskutiert. Die Abbildungsregeln werden erneut in den Abschnitten 5.2.1 *Auswertung von SPARQL-Anfragen* und 5.2.4 *Aufbereitung von WFS-Ergebnissen* aufgegriffen und ihre zugedachte Verwendung in der Programmlogik eines Semantic Web-Proxy vorgestellt.

5.1.6 Informationsvernetzung, Bezüge zu Basiskonzepten

In den vorangegangenen Abschnitten wurden Feinheiten der ontologischen Modellierung herausgegriffen und vorgestellt. Um das Ontologiekonzept abzurunden und die vorherigen Detailsichten in einen Gesamtzusammenhang zu stellen, geht der nun folgende Abschnitt auf potentielle Organisationsstrukturen für INSPIRE-Themenontologien ein. Hierfür wird eine **Architektur aus Vokabularen** skizziert, die in Abbildung 5.8 vereinfacht wiedergegeben ist. Die Themenontologien, abgebildet in der rechten oberen Diagrammecke, sind nicht isoliert, sondern referenzieren externe Vokabulare und binden ihre Konzepte als Grundlagen- oder Basiskonzepte ein. Die dadurch entstehende Vernetzung der Themenontologien entspricht den Grundprinzipien des Semantic Web und Linked Open Data, 1. gleichartige oder ähnliche Konzepte semantisch zu verknüpfen sowie 2. existierende und bewährte Konzepte anderer Ontologien nachzunutzen (siehe Abschnitt 2.2.4).

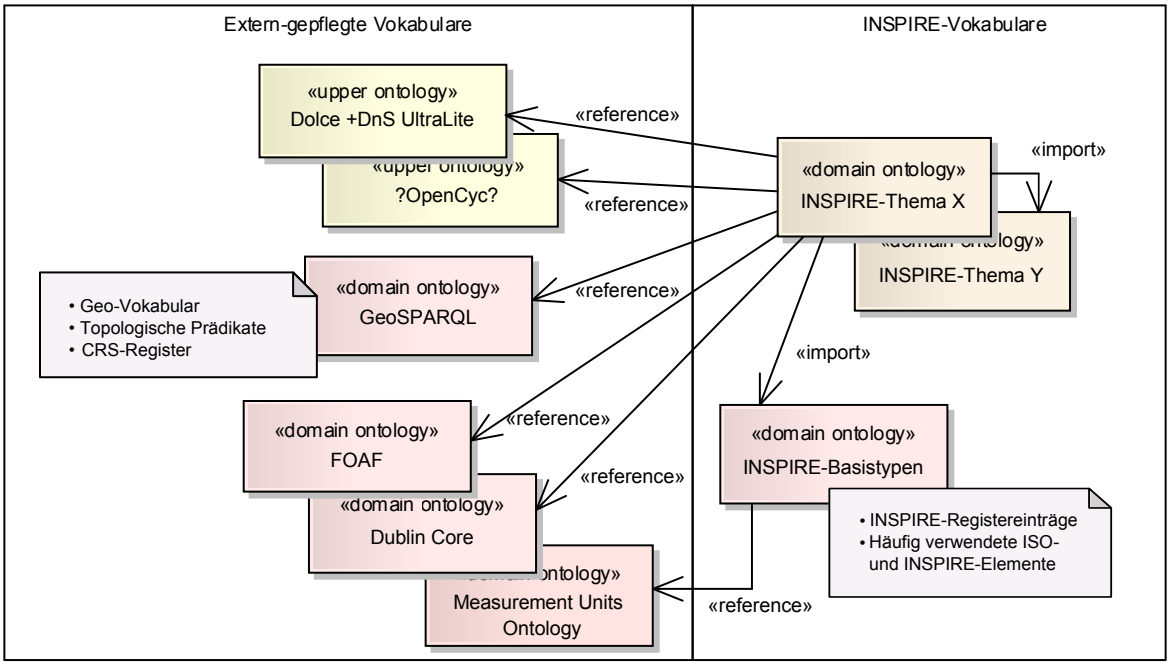


Abbildung 5.8 – Architektur aus verbundenen Ontologien

Zur **Verknüpfung gleichartiger oder ähnlicher Konzepte** (sogenanntes *Ontologie-Alignment*) bieten sich insbesondere Upper-Ontologien an, wie beispielsweise *Dolce +DnS Ultra Lite*[115] oder *OpenCyc*[116]. Das Alignment auf abstrakte Konzepte von Upper-Ontologien (im Diagramm 5.8 gelb eingefärbt) oder fachspezifische Konzepte der Domänenontologien (im Diagramm 5.8 rot eingefärbt) hilft der Klassifizierung von INSPIRE-Konzepten, ob in übergeordnete erkenntnistheoretische oder durch eine Fachgemeinde geprägte Objektkategorien (Definitionsprädikate sind z.B. `rdfs:subClassOf` oder `owl:equivalentClass`). Die Klassifizierung kann einerseits analytischen Zwecken dienen, um Charakteristiken und Besonderheiten der INSPIRE-Datenmodelle zu untersuchen und z.B. verwandte oder komplementäre Konzepte aufzudecken. Andererseits fördert sie die Interoperabilität zwischen den Themenontologien

[115] DnS: Descriptions and Situations, Ontologiepattern für Situationsbeschreibungen; Ultra Lite umfasst als Untermenge die Kernklassen der Dolce-Ontologie
[116] Projektseite: http://www.cyc.com/opencyc

und externen Domänenontologien, schließlich stellen Konzeptzuordnungen eine Art Leitfaden für eine praktische Modelltransformation dar.

Eine **Wiederverwendung existierender und bewährter Konzepte** aus externen Vokabularen erfolgt per Namensreferenz (URI), so dass die Definitionen jener Konzepte nicht in die Themenontologien inkludiert werden müssen. So können vor allem die Definitionselemente der Themenontologien, wie z.B. Klassenrestriktionen von OWL-Klassen (per `owl:onProperty`) oder Definitions- und Wertebereiche von OWL-Prädikaten (`rdfs:domain` und `rdfs:range`), URI-Referenzen auf externe Konzepte enthalten. Der Lösungsansatz sieht u.a. die Wiederverwendung von Konzepten des *GeoSPARQL*-Vokabulars und der *Measurement Units Ontology* vor. In Frage kommen auch die gebräuchlichen Vokabulare *FOAF* und *Dublin Core*.

Zusätzlich zu den Namensreferenzen lassen sich Definitionen externer Konzepte mit einem **OWL-Ontologieimport** inkludieren. Ein OWL-Ontologieimport ist vergleichbar mit dem Import eines XML-Namensraumes und bewirkt, dass alle Aussagen, somit auch die Konzeptdefinitionen einer Ontologie A in eine Ontologie B - und transitiv in eine Ontologie C - eingebunden werden. Das hierzu verwendete Definitionsprädikat `owl:import` setzt im Definitions- und Wertebereich den Ressourcentyp `owl:Ontology` voraus, verknüpft also jeweils die importierende Ontologie-Ressource bzw. -URI mit der für den Import vorgesehenen Ontologie-Ressource. OWL-Ontologieimporte sind ein probates Mittel, die transitiven XML-Importanweisungen der GML-Applikationsschemata in das Semantic Web zu übertragen mit der Folge, dass sich die Themenontologien gegenseitig referenzieren bzw. Konzepte und Annotationen der importierten Ontologien aufnehmen.

Die INSPIRE UML-Modelle verwenden viele gängige **ISO-Datentypen**, die Abschnitt 5.1.1 in primitive und komplexe UML-Datentypen unterscheidet. Während die meisten primitiven UML-Datentypen (z.B. `Boolean`, `DateTime`) eine direkte Semantic Web-Entsprechung haben (z.B. die RDF-Datentypen `xsd:boolean` bzw. `xsd:dateTime`), bestehen komplexe UML-Datentypen wie beispielsweise `CI_ResponsibleParty` und `TM_Period` aus einer Fülle von Detailinformationen und müssen deshalb gesondert behandelt werden. Prinzipiell kommen dieselben Abbildungsregeln zur Anwendung, die für INSPIRE UML-Klassen in Abschnitt 5.1.1 aufgestellt wurden: sie werden umgeformt zu OWL-Klassen und ihre Attribute über OWL-Prädikate wiedergegeben. Jedoch ist hierbei Vorsicht geboten, denn viele ISO-Datentypen haben einen hohen Komplexitätsgrad. Sie weisen u.a. tief verschachtelte XML-Strukturen auf, bieten mehrere Speicheralternativen für equivalente Inhalte an und sind mit Standardelementen und -attributen zur Referenzierung und Namensgebung angereichert. Von einer automatischen Übersetzung all dieser Speicherdetails in RDF/OWL-Repräsentationen sollte abgesehen werden, um die INSPIRE-Themenontologien nicht mit Konzepten zu *überfrachten*, deren Nutzen fraglich erscheint. Stattdessen könnte eine manuelle Modellierung der komplexen UML-Datentypen ins Auge gefasst werden, welche sich die Vorzüge der Annotierung mit XPath-Ausdrücken zu eigen macht und dazu beiträgt, ISO-Elementstrukturen zu vereinfachen, zu konsolidieren und sinnvolle inhaltliche Selektionen vorzunehmen (siehe vorherigen Abschnitt 5.1.5 Absatz *Vorzüge des Gebrauchs von XPath-Ausdrücken*). Die daraus abgeleiteten Ontologiekonzepte sind zweckmäßig in eine separate Ontologie auszulagern, damit sie nur einmal definiert, verschiedenen Themenontologien über den Mechanismus des OWL-Ontologieimports zur Verfügung stehen. Zuzüglich der ISO-Datentypen führt das Generic Conceptual Model im UML-Package *INSPIRE Base Types* weitere grundlegende INSPIRE-Datentypen ein, wie z.B. `Identifier` zum Identifizieren von Objekten. Die Idee ist nun, sowohl ISO- als auch INSPIRE-Datentypen zu vereinen und in einer Domänenontologie namens *INSPIRE-Basistypen* zusammenzufassen (siehe auch Abbildung 5.8).

Die bislang erwähnten Methoden der semantischen Verknüpfung, ob per Ontologie-Alignment, der Namensreferenz in ontologischen Restriktionen oder per Ontologieimport, bezogen sich allesamt auf die schematische bzw. terminologische Ebene der INSPIRE-Themenontologien.[117] Die **Verknüpfung von INSPIRE OWL-Instanzen**, die die ABox bzw. das assertionale Wissen bilden, ist hingegen abhängig vom Verknüpfungsgrad der durch die Downloaddienste

[117] mittels OWL-Ontologieimporten ist prinzipiell auch die Übertragung von Aussagen der ABox (Instanzdaten) möglich

publizierten GML-Features. Die INSPIRE-Datenmodelle weisen zahlreiche themeninterne und -übergreifende Objektreferenzierungen zwischen Feature Types auf, z.B. verweist ein Schutzgebiet (`ProtectedSite`) innerhalb des Schutzgebiete-Datenmodells auf eine für das Management verantwortliche Verwaltung (`ResponsibleAgency`), steht jedoch auch in Beziehung zu themenfremden Objektkategorien, wie z.B. den Bio-geographischen Regionen aus dem gleichnamigen Datenmodell `Bio-GeographicalRegions`. Die verschiedenartigen Objektreferenzierungen werden sich in den späteren GML-Instanzdaten widerspiegeln, sofern es der INSPIRE-Implementierungsstand und die Datenverfügbarkeit seitens der Datenbereitsteller erlauben. Die in den GML-Instanzdaten enthaltenen Objektreferenzen stecken implizit auch in den INSPIRE OWL-Instanzen, die aus der GML/OWL-Datentransformation hervorgehen. Wie im Abschnitt 2.3.2 dargelegt wird, erfolgt die GML-Objektreferenzierung üblicherweise unter Verwendung des XML-Attributes `xlink:href` vergleichbar zur RDF-Ressourcenverlinkung mittels XML-Attribut `rdf:resource` im Format RDF/XML. Desweiteren können Relationen zwischen GML-Features auch über *inline* angegebene bzw. in der XML-Elementhierarchie untergeordnete GML-Features ausgedrückt werden. Beiden GML-Referenzierungsmethoden liegt das GML *Object-Property*-Muster zugrunde, das artverwandt ist mit der RDF Tripel-Syntax, weshalb sich GML-Objektreferenzierungen grundsätzlich problemlos in eine semantische Form überführen lassen.

Die gegenseitigen `xlink-href`-Referenzierungen von GML-Features können sich andererseits als nachteilig in der Kommunikation mit INSPIRE-Downloaddiensten erweisen. Diesem Sachverhalt geht der Abschnitt 5.2.3 *Verteilte GetFeature-Anfragen an WFS-Dienste* auf den Grund. Weitere Verknüpfungsmöglichkeiten der OWL-Instanzdaten werden desweiteren im Abschnitt 5.2.4 *Aufbereitung von WFS-Ergebnissen* behandelt. Die hier aufgezeigten terminologischen wie assertionalen Relationen verhelfen zu einer zweckdienlichen Informationsvernetzung der INSPIRE-Themenontologien und OWL-Instanzdaten nach Linked Open Data-Prinzipien.

5.2 Entwurf eines Semantic Web-Proxy für INSPIRE-Downloaddienste

Der gewählte Architekturansatz entspricht dem **Proxy-Entwurfsmuster**. Die Proxy-Komponente (im Weiteren als *OWS-Proxy* bezeichnet) bedient Semantic Web-Clients, indem sie SPARQL-Anfragen empfängt, sie in abgewandelter Form an INSPIRE-Downloaddienste weiterreicht und die erhaltenen Ergebnisse an den Semantic Web-Client ausliefert. Eine genauere Ablaufbeschreibung ist Thema dieses Abschnittes. Die Gliederung in Unterabschnitte folgt der **Prozesskette der Anfragebearbeitung**, die in Abbildung 5.9 zu sehen ist.

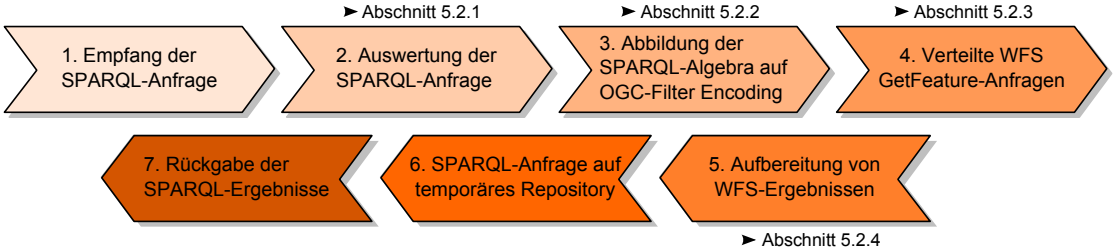


Abbildung 5.9 – Workflow der SPARQL-Anfragebearbeitung im OWS-Proxy

Die SPARQL-Anfragebearbeitung ist geprägt durch ein **doppelstufiges Auswerteverfahren**. Zunächst werden GetFeature-Anfragen an WFS-Dienste gestellt, die eine erste Filterung mit Hilfe von OGC Filter Encoding-Ausdrücken vornehmen (siehe Schritt 4 in Abbildung 5.9). Die WFS-Ergebnisse gleichen einer Obermenge der vom Anwender gewünschten Resultate. Die WFS-Ergebnisse werden in eine Semantic Web-Form (RDF/OWL) überführt und mit der initial vom Benutzer abgeschickten SPARQL-Anfrage ein zweites Mal gefiltert (siehe Schritt 6 in Abbildung 5.9), so dass alle SPARQL-Sprachfeinheiten zur Anwendung kommen. Das In-Reihe-Schalten zweier Filtervorgänge ist nötig, da sich die beiden Anfragesprachen SPARQL und Filter Encoding nicht verlustfrei aufeinander abbilden lassen. Eine tiefergehende Betrachtung liefert der Abschnitt 5.2.2.

Abgesehen vom doppelten Auswerteverfahren ist im Lösungsansatz ein genauer Workflow zur Anfragebearbeitung definiert. Um nun zu den Unterabschnitten überzuleiten, seien die einzelnen Programmschritte aus Abbildung 5.9 hier kurz umrissen. Ein gesonderter Abschnitt 5.2.5 geht darüber hinaus auf die Integration von GIS-Funktionalität ein.

1. **Empfang der SPARQL-Anfrage**: der OWS-Proxy empfängt als regulärer SPARQL-Endpoint (konform zum *SPARQL Protocol for RDF*) die SPARQL-Anfrage des Benutzers
2. **Auswertung der SPARQL-Anfrage**: die SPARQL-Anfrage durchläuft mehrere Auswerteschritte. Zum einen wird die in der *Query*-Form befindliche Anfrage in die strukturell optimierte *Algebra*-Form umgewandelt (siehe Abschnitt 2.2.3). Danach werden die einzelnen Anfragezweige durchsucht, um Hinweise auf INSPIRE-Konzepte aufzuspüren und daraus GML-Elementpfade (XPath-Ausdrücke) für WFS GetFeature-Requests zu generieren. Desweiteren kann ein regelbasiertes *Query Rewriting* sowie das *Reasoning* über Ontologiekonzepte stattfinden; siehe Abschnitt 5.2.1
3. **Abbildung der SPARQL-Algebra auf Filter Encoding**: aus der Operatorenhierarchie des SPARQL-Algebra Ausdrucks geht ein weitestgehend konformes Filter Encoding-Abbild hervor, gegebenenfalls sind je nach Datenangebot der verfügbaren INSPIRE-Downloaddienste mehrere voneinander abweichende Filter Encoding-Ausdrücke bzw. WFS GetFeature-Requests zu bilden. Abstriche müssen aufgrund der unterschiedlichen Mächtigkeit und Zielsetzung beider Sprachen in Kauf genommen werden; siehe Abschnitt 5.2.2
4. **Verteilte WFS GetFeature-Anfragen**: jeder INSPIRE-Downloaddienst, der die angefragten Ressourcentypen bereitstellt, wird vom OWS-Proxy mit individuellen WFS GetFeature-Requests asynchron aufgerufen; siehe Abschnitt 5.2.3

5. **Aufbereitung von WFS-Ergebnissen**: die INSPIRE-Downloaddienste senden ihre Ergebnisse als GML-Instanzdokumente zurück an den OWS-Proxy, der sie verschiedentlich aufbereitet. Zunächst wird die GML/OWL-Transformation nach den Regeln aus Abschnitt 5.1.5 angestoßen und die daraus resultierenden OWL-Ressourcen in ein temporäres Repository überführt. Mit dem temporären Repository ist ein im Arbeitsspeicher geladener flüchtiger RDF-Triplestore gemeint, der nur innerhalb der Session einer Anfragebearbeitung gültig ist. Dort können die Ergebnisdaten mit weiteren Aussagen beliebig in Relation gesetzt und angereichert werden; siehe Abschnitt 5.2.4
6. **SPARQL-Anfrage auf temporäres Repository**: das temporäre Repository wird mit der initial vom Benutzer abgeschickten SPARQL-Anfrage gefiltert und ihr so inhaltlich und bezüglich der Ergebnisform entsprochen.
7. **Rückgabe der SPARQL-Ergebnisse**: der SPARQL-Endpoint schickt die Resultate der SPARQL-Auswertung an den Semantic Web-Client zurück. Je nach Formatwahl/ Anfrageart (z.B. SPARQL `Select` oder `Construct`-Anfragen) erhält der Client die Resultate im Tabellenformat *SPARQL Query Results XML Format* bzw. in der für RDF-Graphen üblichen Serialisierung *RDF/XML*

Die unten aufgeführte exemplarische SPARQL-Anfrage (Codebeispiel 16) ist als zusammenhängendes Auswertebeispiel der folgenden Unterabschnitte gedacht. Die Anfrage startet die Recherche nach Schutzgebieten `ont_ps:ProtectedSite` (siehe Zeilen 11 und 21), deren digitales INSPIRE-Speicherobjekt einen Datumsstempel ab dem 1.1.2009 aufweist (Zeile 17) oder die eine geographische Ausdehnung haben, die mit der Vergleichsgeometrie in Zeile 28 interagiert. Darüber hinaus wird optional die Schutzwidmung der gefilterten Gebiete abgefragt (Zeile 14/15 und 25/26).

```
1  PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
2  PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
3  PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/>
4  PREFIX geof: <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/>
5  PREFIX sf10: <http://www.opengis.net/def/dataType/OGC-SF/1.0/>
6
7  SELECT ?feature ?designation ?beginLifespan ?geomLiteral
8  WHERE
9  {
10   {
11     ?feature a ont_ps:ProtectedSite .
12     ?feature ont_ps:beginLifespanVersion ?beginLifespan .
13     OPTIONAL{
14       ?feature ont_ps:siteDesignation ?designationContext .
15       ?designationContext ont_ps:designation ?designation .
16     }
17     FILTER( ?beginLifespan > "2009-01-01T12:00:00"^^xsd:dateTime )
18   }
19   UNION
20   {
21     ?feature a ont_ps:ProtectedSite .
22     ?feature ont_ps:geometry ?geometry .
23     ?geometry geo:geomLiteral ?geomLiteral .
24     OPTIONAL{
25       ?feature ont_ps:siteDesignation ?designationContext .
26       ?designationContext ont_ps:designation ?designation .
27     }
28        FILTER( geof:within(?geomLiteral, "CRS:84 POLYGON((18 47,18.1 47,18.1 49,18 49,18 47))"^^sf10:WKTLiteral))
29   }
30 }
```

Codebeispiel 16 – Exemplarische SPARQL-Anfrage

5.2.1 Auswertung von SPARQL-Anfragen

Der Workflow zur Anfragebearbeitung beginnt mit der Analyse der empfangenen SPARQL-Anfrage. Dazu wird die als Zeichenkette übertragene SPARQL-Anfrage vom OWS-Proxy geparst und als sogenannter *abstrakter Syntax-Baum* objektorientiert im Arbeitsspeicher abgelegt. Daraufhin findet eine Umwandlung der SPARQL Query-Form in die optimierte *SPARQL Algebra-Form* statt, deren Vorteile gegenüber der SPARQL Query-Form in Abschnitt 2.2.3 geschildert sind. Aus der SPARQL Query in Codebeispiel 16 folgt der SPARQL Algebra-Ausdruck im nachstehenden Codebeispiel 17.

```
PROJECT (?feature ?designation ?beginLifespan ?geomLiteral
  UNION
    FILTER (?beginLifespan > "2009-01-01T12:00:00"^^<http://www.w3.org/2001/XMLSchema#dateTime>)
      LEFTJOIN
        BGP(
           ?feature <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
               <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/ProtectedSite> .
           ?feature <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/beginLifespanVersion>
               ?beginLifespan .
        )
        BGP(
           ?feature <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/siteDesignation>
               ?designationContext .
           ?designationContext <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/designation>
               ?designation .
        )
    FILTER (?geomLiteral <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/within>
                   "CRS:84 POLYGON((18 47,18.1 47,18.1 49,18 49,18 47))"^^
                       <http://www.opengis.net/def/dataType/OGC-SF/1.0/WKTLiteral>)
      LEFTJOIN
        BGP(
           ?feature <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
               <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/ProtectedSite> .
           ?feature <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/geometry> ?geometry .
           ?geometry <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/geomLiteral> ?geomLiteral .
        )
        BGP(
           ?feature <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/siteDesignation>
               ?designationContext .
           ?designationContext <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/designation>
               ?designation .
        )
```

Codebeispiel 17 – SPARQL Algebra-Ausdruck der SPARQL Query aus Codebeispiel 16

Ausgehend von der Algebra Form wird die SPARQL-Anfrage nach INSPIRE-relevanten Bestandteilen analysiert. Gesucht werden Hinweise auf die Konzepte der INSPIRE-Themenontologien, die aufgrund ihrer im Abschnitt 5.1.5 definierten Annotationen über semantisch-verwandte GML-Speicherpfade informieren und dadurch die Voraussetzung schaffen, INSPIRE-Downloaddienste nach den dazugehörigen GML-Inhalten abzufragen. Es ist also aus der SPARQL-Anfrage herauszulesen, welche INSPIRE-Themen und -Objekttypen gefragt sind, damit die im OWS-Proxy registrierten themenbezogenen Downloaddienste ermittelt und individuelle Datenabrufe eingeleitet werden können (siehe Abschnitte 5.2.2 und 5.2.3).

INSPIRE-Konzepterkennung

Die zur Laufzeit des OWS-Proxy bekannten Konzepte der INSPIRE-Themenontologien werden mit den Graphenmustern der SPARQL-Anfrage verglichen. Sollte die SPARQL-Anfrage INSPIRE-Konzepte enthalten, so werden die betroffenen Graphenmuster für spätere Zwecke entsprechend gekennzeichnet. Die Konzepterkennung ist vielschichtig zu gestalten, um verschiedenartige Anhaltspunkte aufzudecken und verwerten zu können.

Primär verwendete SPARQL-Sprachkonstrukte zur INSPIRE-Konzepterkennung sind:

- **Ressourcenklassifizierungen**: die SPARQL-Graphenmuster bestehen aus einzelnen Tripelmustern, deren Subjekte und Objekte zumeist als Variablen deklariert sind, z.B. `?feature ont_ps:beginLifespanVersion ?beginLifespan`. Benachbarte Tripelmuster erlauben das Klassifizieren der Subjekt- bzw. Objektvariablen mittels `rdf:type`-Deklarationen (z.B. `?feature rdf:type ont_ps:ProtectedSite` oder in der offiziellen RDF-Abkürzungsschreibweise `?feature a ont_ps:ProtectedSite`). Enthält eine SPARQL-Anfrage Klassifikationen, ist ihre Übereinstimmung mit OWL-Klassen der INSPIRE-Themenontologien zu prüfen.

- **Benannte Prädikate**: im Allgmeinen sind die Prädikate innerhalb von Tripelmustern einer SPARQL-Anfrage benannt. Sie sind also nicht variabel, sondern im Regelfall mit ihrer eindeutigen URI angegeben oder seltener auch per `rdf:type`-Deklaration bestimmt. Hier ist ebenso leichthin zu prüfen, ob sie namentlich INSPIRE OWL-Prädikaten gleichen. Trifft das zu, lassen sich auch die im gleichen Tripelmuster enthaltenen Subjekt- und Objektvariablen klassifizieren. Die nötige Klassifizierungsinformation, d.h. Definitions- und Wertebereiche eines Prädikats, liefern die `cf:rdfsDomain`- und `cf:rdfsRange`-Axiom-Annotationen, welche den OWL-Prädikaten der Themenontologien als Spezialisierung der Annotation `cf:xpath` angehängt sind (siehe Abschnitt 5.1.5). Sofern also im Tripel `?designationContext ont_ps:designation ?designation`) das Prädikat `ont_ps:designation` als INSPIRE-Konzept detektiert wurde, geben die `cf:rdfsDomain`- und `cf:rdfsRange`-Annotationen Auskunft über die Konzepte der Variablen `?designationContext` und `?designation`.

- **INSPIRE-Ressourcenbezeichner**: der Abschnitt 5.1.2 *Identifikationsmanagement* schlägt eine Namenskonvention zur einheitlichen Benennung von INSPIRE OWL-Instanzen vor. Aus den drei INSPIRE-Identifikatorattributen `namespace`, `localId` und `versionId` wird dabei eine eindeutige Ressourcen-URI zusammengesetzt. Die Angabe der Ressourcen-URI innerhalb eines SPARQL-Graphenmusters erlaubt die scharfe Suche nach den Detailinformationen der benannten Ressource. In diesem Anfrageszenario ist die Information zum INSPIRE-Namensraum (Attribut `namespace`) nützlich, um aufzulösen, welche INSPIRE-Downloaddienste GML-Instanzdaten zu diesem INSPIRE-Namensraum führen. Ein in Frage kommender Downloaddienst gruppiert GML-Instanzen wiederum in eine bestimmte Anzahl an INSPIRE-Feature Types, die OWL-Klassen der Themenontologien entsprechen. Zusammenfassend lässt sich also aus einer INSPIRE Ressourcen-URI auf ein oder mehrere INSPIRE-Konzepte schließen. Einem dieser Konzepte muss die gesuchte Ressource angehören (siehe auch Abbildung 5.2, *Ablauf einer OWL-Instanzanfrage mit URI-Resolving*).

- **Konzeptverlinkungen zu externen Ontologien**: das semantische Verknüpfen der Themenontologien mit externen Domänen- und Upper-Ontologien (*Ontologie-Alignment*) steigert die Interoperabilität der Ontologien untereinander (siehe Abschnitt 5.1.6). Die Resultate eines Ontologie-Alignments, ontologische Aussagen zum Mapping von externen auf INSPIRE-Konzepte, können in die INSPIRE-Themenontologien importiert werden und dienen im Zuge einer Anfragebearbeitung als Übersetzungshilfe. Beinhaltet eine SPARQL-Anfrage externe Konzepte, so werden diese je nach vorhandenen Mapping-Definitionen in INSPIRE-Konzepte übersetzt und betreffende SPARQL-Graphenmuster für die weitere Auswertung modifiziert.

- **Auflösung von Variablen**: anstelle von klassifizierten Ressourcenvariablen und benannten Prädikaten können Tripelmuster auch unbestimmte Ressourcen- oder Prädikatvariablen enthalten, die isoliert betrachtet keinem Konzept zugeordnet werden können. Die Konzeptzugehörigkeit lässt sich aber meist im Kontext des jeweiligen Graphenmusters durch die miteinander in Beziehung stehenden Tripelmuster auflösen. Ein Vergleich der im

Graphenmuster vorgefundenen Konzeptzusammenhänge mit jenen in den Themenontologien lässt Rückschlüsse auf die Variablen zu. Die Abbildung 5.10 zeigt 1. eine Prädikatvariable (`?p2?`) und 2. eine nicht-klassifizierte Ressourcen-Variable (`?r2?`). Die benachbarten Subjekte, Objekte und Prädikate geben mittels `rdf:type`-Statements bzw. `cf:rdfsDomain`- und `cf:rdfsRange`-Annotationen Hinweise auf die Konzepte von `?p2?` und `?r2?`. Zur Erinnerung: die `rdf:type`-Statements sind ebenfalls der SPARQL-Anfrage zu entnehmen, während die `cf:rdfsDomain`- und `cf:rdfsRange`-Annotationen für alle INSPIRE-Konzepte der Themenontologien definiert sind.

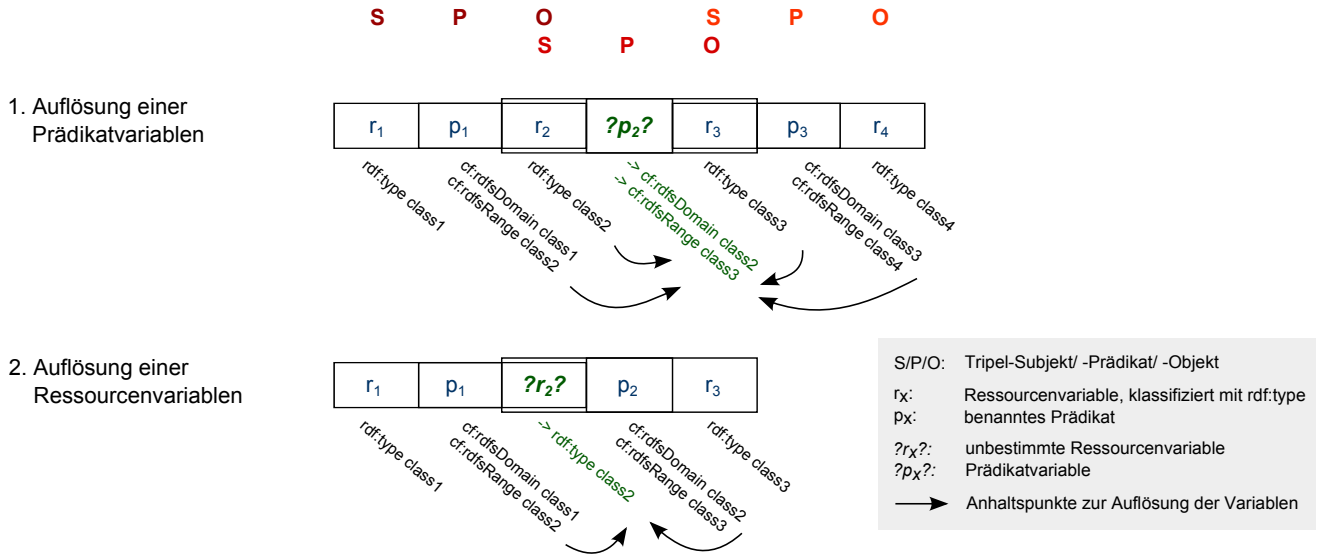


Abbildung 5.10 – Indizien zur Variablenauflösung in SPARQL-Graphenmustern

Für alle hier beschriebenen Kategorien von Anhaltspunkten wird die SPARQL-Anfrage sukzessive durchsucht. Nachfolgend ist in einem Validierungsdurchlauf zu kontrollieren, ob die einzelnen Konzeptzuweisungen **Widersprüche** enthalten. Wird beispielsweise die Ressourcen-Variable $\texttt{r}_1$ in Abbildung 5.10 als Konzept `class13` erkannt, während das Prädikat $\texttt{p}_1$ als Subjekt eine Instanz des Konzepts `class1` verlangt (Hinweis über die Annotation `cf:rdfsDomain class1`), dann liegt ein Widerspruch in der Konzepterkennung vor. Vorausgesetzt die Themenontologien sind korrekt annotiert, lassen derartige Diskrepanzen auf falsch formulierte Graphenmuster der SPARQL-Anfrage schließen. Gemeint sind Graphenmuster, die Konzeptzusammenhänge durcheinander bringen und Knoten-Kanten-Beziehungen filtern, welche nicht exakt den Definitionen der Themenontologien entsprechen. In solchen Einzelfällen scheitert die weitere Auswertung eines Filterpfades oder es wird gar die komplette SPARQL-Anfrage als ungültig deklariert und der Anwender per SPARQL-Fehlerprotokoll informiert. Abgesehen von Widersprüchen können auch **Mehrdeutigkeiten** in Form von doppelten Konzeptzuweisungen auftreten, die vornehmlich von Namenskonflikten der UML-Klassenattribute und -Assoziationen herrühren, sobald jene in OWL-Prädikate der Themenontologien konvertiert werden (siehe hierzu Abschnitt 5.1.1). Die Folge dieser Namenskonflikte sind Mehrfachannotationen der OWL-Prädikate mit `cf:rdfsDomain` und `cf:rdfsRange` (siehe Abschnitt 5.1.5). Ein weiterer Validierungsdurchlauf ist demnach erforderlich, um einen Abgleich zwischen den Konzeptreferenzen der Ressourcenvariablen mit denjenigen der Prädikate herbeizuführen. Falls die Prüfung die Gültigkeit der doppelten Konzeptbelegungen feststellt, ist jede Kombination aus Tripelmustern zu berücksichtigen und danach die Suche zu gestalten.

GML-Elementadressierung

Aus dem Ergebnis der Konzepterkennung, den Konzeptzuweisungen zu den Strukturelementen jedes Tripelmusters, sind im nächsten Schritt GML-Speicherpfade abzuleiten. Diese dienen als Zieladressen der attributiven Filterung innerhalb von WFS GetFeature-Anfragen (siehe nächster Abschnitt 5.2.2, Codebeispiel 19) oder zum Auslesen der WFS-Ergebnisse (siehe Abschnitt 5.2.4). Gemäß ihrer Konzeptzugehörigkeit und der mit dem jeweiligen Konzept verknüpften Information `cf:xmlName` und `cf:xpath`, sind allen Subjekten, Prädikaten und Objekten eines Graphenmusters GML-Objekttypen bzw. GML-Elementpfade zuzuordnen. Die Abbildung 5.11 zeigt das Vorgehen am Beispiel der ersten vier Tripelmuster aus Codebeispiel 16, berücksichtigt also alle Tripelmuster vor dem `UNION`-Operator. Die Tripelmuster (Elemente schwarz dargestellt) sind Konzepten zugeordnet (blau), die wiederum GML-Objekttypen und relativen GML-Elementpfaden entsprechen (grün).

In Graphenmustern tauchen Sequenzen von zusammengehörigen Tripelmustern auf, z.B. die Sequenz aus den Tripelmustern 3 und 4: `?feature/ont_ps:siteDesignation/?designationContext/ont_ps:designation/?designation`. Solche Sequenzen sollen im Weiteren als **Prädikatpfade** bezeichnet werden. Den Prädikatpfaden vergleichbar ist die Aneinanderreihung relativer GML-Elementpfade, deren jeweils erste bzw. letzte XPath-Pfadschritte übereinstimmen. Lässt man die sich überschneidenden XPath-Pfadschritte weg, so ergeben sich 1-n absolute GML-Elementpfade als Entsprechungen eines Prädikatpfades. Beispielsweise erhält man für den oben genannten Prädikatpfad den absoluten GML-Elementpfad `ps-f:ProtectedSite/ps:siteDesignation/ps:DesignationType/ps:designation`.

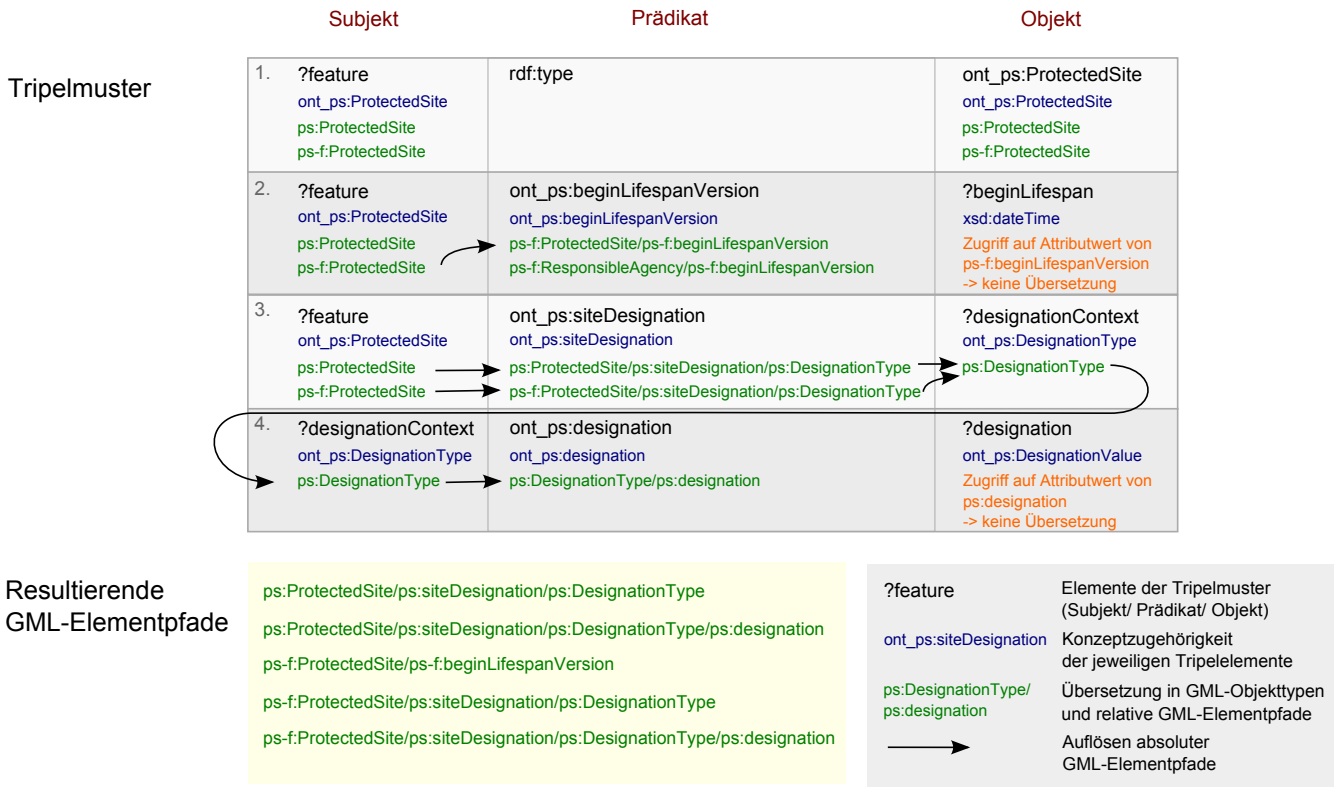


Abbildung 5.11 – Herleitung von GML-Elementpfaden aus SPARQL-Graphenmustern

Im Hinblick auf die spätere Verwendung in WFS GetFeature-Anfragen, müssen alle GML-Elementpfade ausgehend von einem Feature Type formuliert sein, z.B. angefangen mit `ps:ProtectedSite`. Denn genau genommen sind WFS-Dienste **Record-basierte Systeme**, die jegliche Auskunft nur in Abhängigkeit zu ihren Records erteilen, hier also den Feature Types. Daraus lässt sich die **allgemeine Bedingung** ableiten, dass SPARQL-Graphenmuster Subjekte bzw. Subjektvariablen enthalten müssen, die sich zu Feature Types auflösen lassen, damit sich WFS GetFeature-Anfragen

auf jene Feature Types beziehen können. Diejenigen GML-Elementpfade, die keine Feature Types enthalten, sind für die WFS-Dienstekommunikation und ihre -Ergebnisverwertung nicht von Belang und werden aussortiert.

Mit der Herleitung der GML-Elementpfade könnte die Analyse der SPARQL-Anfrage im Grunde abgeschlossen sein. Die bisherigen Auswertungen identifizieren die angefragten Inhalte und stellen umfassende Zusammenhänge her. Was jedoch noch unberücksichtigt bleibt sind die strukturellen Aspekte einer SPARQL-Anfrage, auf deren Eigenheiten kurz eingegangen werden soll. Betrachtet man sich die SPARQL-Anfrage in Codebeispiel 16, so fällt auf, dass Variablen in verschiedenen Graphenmustern gleichbenannt sein dürfen (z.B. `?feature`, siehe Zeile 11 und 21). Aufgrund der freien Namenswahl in SPARQL können Namenskonflikte auftreten, falls die gleichbenannten Variablen unterschiedliche Konzepte repräsentieren oder einem anderen Anfragekontext angehören. Die beiden `?feature`-Variablen beispielsweise unterscheiden sich in ihrem Anfragekontext, der jeweils charakterisiert ist durch individuelle Filterungen (Zeilen 17 bzw. 28) oder abhängige Prädikatpfade (Zeile 12 bzw. 22/23). Gleichfalls problematisch sind SPARQL-Variablen, die in Graphenmustern oder Filterfunktionen isoliert auftreten, z.B. die Variable `?geomLiteral` innerhalb der `geof:within`-Filterfunktion (Zeile 28). Ohne die Kenntnis über ihren Prädikatpfad (Zeile 22/23) ist die Variable `?geomLiteral` nicht zu lokalisieren und aufgrunddessen nicht semantisch einzuordnen. Derartige Namenskonflikte oder Bestimmungsschwierigkeiten lassen sich auflösen, indem die **Sichtbarkeit von SPARQL-Variablen** innerhalb der SPARQL-Operatorenhierarchie beachtet wird. Die SPARQL-Variablen wirken *von innen nach außen*, d.h. ihre Sichtbarkeit reicht von den innersten Operatoren, den Graphenmustern, in denen sie deklariert wurden, bis zu äußeren Operatoren, wie z.B. `Filter` und `LeftJoin`.

Anhand der Operatorenhierarchie des SPARQL Algebra-Ausdrucks aus Codebeispiel 17 soll dieser Umstand verdeutlicht werden. Die Operatorenhierarchie ist zur besseren Übersicht in das Codebeispiel 18 extrahiert, d.h. ohne Vergleichsmuster und Filterausdrücke dargestellt. Die Operatoren sind in der Reihenfolge ihres Auftretens nummeriert. Zum Vergleich: die Bedeutung der Variablen `?geomLiteral` ist definiert im `BGP`-Operator 3 und wird verwendet im `Filter`-Operator 2. Die beiden gleichbenannten `?feature`-Variablen, die in den `BGP`-Operatoren 1 bzw. 3 definiert wurden, haben ihre individuelle Sichtbarkeit begrenzt auf den `Filter`-Operator 1 bzw. 2. Der `Union`-Operator in Zeile 2 vereint schließlich die Ergebnismengen (*Solution sequences*) beider `?feature`-Variablen und vereinheitlicht den Zugriff auf die Variable `?feature` in den höherrangigen Operatoren (hier z.B. der `Projekt`-Operator in Zeile 1).

```
1  PROJECT1
2    UNION1
3      FILTER1
4        LEFTJOIN1
5          BGP1
6          BGP2
7      FILTER2
8        LEFTJOIN2
9          BGP3
10         BGP4
```

Codebeispiel 18 – SPARQL Algebra-Operatorenhierarchie aus Codebeispiel 17

Für die weitere Anfragebearbeitung sind also nicht nur die resultierenden GML-Elementpfade und die zugehörigen Prädikatpfade zu speichern. Zusätzlich ist den GML-Elementpfaden eine Strukturinformation mitzugeben, die sich aus der Platzierung der korrespondierenden SPARQL-Variable bzw. Prädikatpfades ableitet. Demgemäß ist z.B. der GML-Elementpfad `ps-f:ProtectedSite/ps-f:beginLifespanVersion` mit dem Verweis auf den Operator `BGP1` zu markieren. Wird im Anschluss daran die Sprachabbildung von SPARQL auf Filter Encoding ausgeführt (siehe nächster Abschnitt 5.2.2), so kann während des Durchlaufes durch die SPARQL-Operatorenhierarchie beim Erreichen des Operators `BGP1` oder seiner übergeordneten Operatoren `LeftJoin1` oder `Filter1` der relevante Prädikatpfad bzw. der genannte GML-Elementpfad aufgelöst und in einen Filter Encoding-Ausdruck eingesetzt werden.

5.2.2 Abbildung der SPARQL-Algebra auf OGC-Filter Encoding

Die Abbildung von SPARQL auf Filter Encoding gehört zusammen mit der Ergebnistransformation von GML nach RDF/OWL zu den beiden Kernaufgaben des OWS-Proxy. Das Ziel der Sprachabbildung ist es, möglichst **präzise WFS GetFeature-Requests** zu formulieren, damit die Ergebnisse der INSPIRE-Downloaddienste im Idealfall genau den angeforderten Informationen entsprechen. Das verringert die Latenzzeiten, schmälert den Datentransfer und führt nicht zuletzt zu einer geringeren Systemauslastung des OWS-Proxy während der Ergebnisaufbereitung.

Beide Anfragesprachen liegen in **verschiedene Spezifikationsversionen** vor, die mehr oder minder stabil sind. Um Verwechslungen vorzubeugen, wird nun eine Festlegung für je eine bestimmte SPARQL- bzw. Filter Encoding-Version getroffen, die es besonders zu untersuchen gilt (bedeutende Differenzen und Neuerungen anderer Versionen werden ansatzweise genannt). Mit Blick auf SPARQL kommt nur die W3C-Empfehlung der Version 1.0 [Prud'hommeaux & Seaborne 2008] in Betracht, da derzeit alle operativen SPARQL-Endpoints diese Version unterstützen und sich die Version 1.1 [Harris & Seaborne 2012] noch in einem - wenn auch sehr fortgeschrittenen - Entwicklungsstadium befindet. Die Version 1.1 erweitert konsequent die sprachlichen Mittel von SPARQL, u.a. mit *Subqueries*, die vergleichbar zu *SQL-Subselects* sind, oder Negierungsoperatoren, für die es in der Version 1.0 nur einen Workaround gibt (sogenanntes *Negation as Failure*). Nichtsdestotrotz bleibt der Sprachstamm von SPARQL ausnahmslos erhalten. Die hier konzipierte Sprachabbildung kann sich also auf jene Kernfunktionalität der Version 1.0 konzentrieren, ohne dabei die Aufwärtskompabilität zur Version 1.1 zu verlieren.

Bezüglich der Sprache Filter Encoding (FE) ist die Wahl leider etwas komplizierter. Die FE-Versionen, die alle bereits als finale Implementierungsspezifikationen vorliegen, sind eng verwandt mit den Versionen der WFS-Dienste bzw. müssen sich für gültige WFS GetFeature-Anfragen gleichen, damit der WFS-Dienst spezifikationsgetreu antworten kann. So bedingt die WFS-Version 1.1 die Verwendung der FE-Version 1.1. Eine GetFeature-Anfrage setzt sich aus einer *WFS-Query*, die die Feature Types benennt und geringfügige Ergebnismodifikatoren zulässt, und einem optionalen, für eine Filterung aber unabdingbaren FE-Ausdruck zusammen. Im Grunde ist also SPARQL mit der Kombination aus WFS-Query und FE zu vergleichen. Jedoch decken die FE-Operatoren einen Großteil der Ausdrucksmittel von GetFeature-Anfragen ab, weshalb sich der SPARQL-Vergleich überwiegend auf den FE-Sprachumfang konzentriert. Beim Identifizieren einer zweckmäßigen FE-Version fällt auf, dass die Version 1.0 aus dem Jahr 2001 gegenüber der Version 1.1 [Vretanos 2005*a*] kaum inhaltliche Differenzen aufweist. Die FE-Version 2.0 [Vretanos 2010*a*] führt zwar neue Operatoren ein [118], verändert Filter Encoding in seiner Ausdrucksstärke jedoch nur unwesentlich. Der Entwurf der INSPIRE Technical Guidance für Downloaddienste [INSPIRE „Network Services" Drafting Team 2009] empfiehlt hinsichtlich des INSPIRE-Diensttyps *Direct Access Download Service* den WFS 2.0 [Vretanos 2010*b*] in Kombination mit FE 2.0 einzusetzen. Demnach sollte FE 2.0 als die zukünftige Referenz-Anfragesprache im Umfeld von INSPIRE auch hier als die Zielsprache der Sprachabbildung gelten. Allerdings ist auch zu berücksichtigen, dass bislang noch keine WFS 2.0-Implementierung auf dem Markt ist. Im Hinblick auf die prototypische Umsetzung und unter dem Gesichtspunkt, dass FE 1.1 eine repräsentative und in der Praxis bewährte funktionale Untermenge von FE 2.0 ist, wird die SPARQL-Abbildung auf FE 1.1 vorgenommen.

Sprachvergleich

Damit nun beide Sprachen mit einem Mindestmaß an Objektivität gegenübergestellt werden können, erweist sich eine neutrale Vergleichsgrundlage als zielführend. Hierfür ist zweifelsohne die meistgebrauchte Datenbanksprache *Structured Query Language* (SQL) eine gute Wahl. Sie ermöglicht dank ihrer *Data Query Language*-Befehle (DQL) das Durchsuchen relationaler Datenbanken. Eine seltener anzutreffende Speicherform sind XML-Datenbanken und dateibasierte Speicherungen von XML-Dokumenten, für die die Anfragesprache XQuery [Boag et al. 2010] entwickelt wurde. Jedoch

[118] Operatoren für zeitliche Filterung, zur Ressourcen-Identifikation sowie schematische Platzhalter zwecks individueller Operatorenerweiterung

muss man SQL vor XQuery und anderen deklarativen Anfragesprachen den Vorzug geben, denn einerseits hat SQL aufgrund seines großen Bekanntheitsgrades ein natürliches Potential zur *Vergleichssprache*, andererseits weisen beide hier zu untersuchenden Sprachen eine große Verwandtschaft zur SQL-Syntax auf. Die SQL-Syntax beruht auf dem theoretischen Fundament der sogenannten **Relationalen Algebra** [Elmasri & Navathe 2002] und erweitert diese um nur wenige Ausdrücke beispielsweise zur Ergebnisdarstellung. Die Relationale Algebra definiert ein Operatorenset aus 3 unären Operatoren (*Projektion*, *Selektion*, *Umbenennung*), die auf eine einzelne Relation bzw. Datenbanktabelle angewendet werden, und 6 binären Operatoren für zwei unterschiedliche Eingaberelationen (*Vereinigung*, *Durchschnitt*, *Differenz*, *Division*, *Kartesisches Produkt* und verschiedene *Join*-Operatoren). Darüber hinaus können manche der Operatoren (z.B. Durchschnitt, Division und Joins) aus den restlichen Operatoren abgeleitet werden. Aus unären wie aus binären Operationen folgt eine Ergebnisrelation.

Die Tabelle 5.4 zeigt die Funktionalitäten der SPARQL-Algebra und des Filter Encoding projiziert auf die Operatoren der Relationalen Algebra. Zusätzlich sind der Tabelle 5.4 die drei Spezialrubriken *sortierende Operatoren*, *räumliche Operatoren* und *temporale Operatoren* angefügt, die den Komfort in der Datenverarbeitung steigern helfen und über den neutralen Operatorensatzes der Relationalen Algebra hinausreichen. Aus der Tabelle 5.4 geht hervor, dass beide Sprachen, SPARQL als auch Filter Encoding, nicht den kompletten relationalen Sprachumfang abdecken, dennoch weitreichende Ausdrucksmittel besitzen - mit einem rein optischen Vorteil für SPARQL. Hingegen besticht Filter Encoding hinsichtlich der vielen domäneneigenen Operatoren für spatiotemporale Datenanalysen (siehe in Tabelle 5.4 die Rubrik *spezielle Opertorenfamilien*). Dadurch lässt sich eine Anfrage in ihrer Komplexität verringern und ist fachlich leichter nachvollziehbar.

Die in der Tabelle 5.4 dargestellten Konformitäten zu den relationalen Operatoren werden hier nicht im Einzelnen besprochen und sind mit Einsicht in die jeweilige Spezifikation nachzuvollziehen. Obwohl die Tabelle 5.4 einen Einblick in die Ausdrucksfähigkeit der jeweiligen Sprache erlaubt, liefert sie kein klares Bild über die Möglichkeiten einer Sprachabbildung von SPARQL auf Filter Encoding. Eine Analyse zu den wesentlichen Sprachcharakteristiken soll hierzu beitragen. Die Analyse bezieht sich auf drei Teilbereiche: 1. die Datenquelle bzw. die *Basiseinheit* der jeweiligen Sprache, 2. ihre *Filtervorgänge* und 3. ihre Ausdrucksmittel zur *Ergebnisaufbereitung.*

Basiseinheit

Mittels Filter Encoding wird eine Filterung von *Features* durchgeführt, welche die Basiseinheit im Rahmen der WFS GetFeature-Suche bilden. Features sind in sich abgeschlossene Objekte als Abstraktion der Realwelt. Sie entsprechen der schematischen Vorgabe von Feature Types, deren Ausgestaltung von flachstrukturierten und wenig attributierten bis hin zu hierarchisch sehr komplexen Objekttypen mit vielen Detailinformationen reicht. Bei INSPIRE-Features Types (Spatial objects types) handelt es sich um überwiegend komplexe Feature Types. Als GML kodiert, gleichen INSPIRE-Feature Types demnach XML-Complex Types mit einer Fülle von XML-Unterelementen (GML-Properties bzw. untergeordnete GML-Objekte).

Die Sprache SPARQL filtert stattdessen auf der Basiseinheit eines *RDF-Tripel.* In RDF-Tripel werden einzelne Fakten bzw. Aussagen gespeichert, die losgelöst voneinander noch kein Gesamtbild zu einem Realweltobjekt ergeben. Verglichen mit einem Feature und seiner umfänglichen Detailinformation kann ein RDF-Tripel lediglich eine einzige Objekteigenschaft verkörpern. Ein Feature wird deshalb auf 1-n RDF-Tripel abgebildet (vgl. Abschnitt 5.1).

Filtervorgänge

WFS-Dienste operieren vorwiegend auf relationalen (Geo-)Datenbanken, die Features in einer flachen Geometrietabelle oder normalisiert über mehrere Tabellen persistieren. Aus diesem Grund setzen WFS GetFeature-Anfragen mit eingebetteten Filter Encoding-Ausdrücken üblicherweise relationale Filterprozesse in Gang. Seltener dienen als Datenquelle dateibasierte Geoformate, die von WFS-Diensten temporär geladen und objektorientiert durchsucht werden. In beiden Fällen führt das attributive Filtern dazu, dass sich zwar die Ergebnismenge (*Feature Collection*) verkleinert,

Tabelle 5.4 – Funktionalität von SPARQL und Filter Encoding - Relationale Algebra als Vergleichsmaßstab
(* SPARQL-Geltungsbereiche: *GM*: innerhalb/zwischen Graphenmustern; *F*: Filterausdrücke; *EM*: Ergebnismodifikatoren (engl. Solution Modifier); ** WFS/FE-Geltungsbereiche: *Q*: WFS-Query; *FE*: Filter Encoding; *** nach den Gesetzen der relationalen Algebra leiten sich diese Operatoren aus den restlichen ab)

Relationale Algebra	**SPARQL-Algebra**		**WFS-Query/ Filter Encoding**	
Operator (R,S: Relationen)	Operator	Geltungsbereich *	Operator	Geltungsbereich **
Vereinigung ($R \cup S$)	**Union** (Achtung: gleicht einem *Outer Union*)	GM	**<Or>**	FE
	boolscher Operator '||'	F		
Durchschnitt/Schnittmenge *** ($R \cap S$)	–		**<And>**	FE
Differenz ($R \setminus S$ oder $R - S$)	*SPARQL1.0:* Workaround: *Negation as Failure*	GM	**<And>** + **<Not>**	FE
	SPARQL1.1: **Minus**	GM		
Division** ($R \div S$)	–		–	
Kartesisches Produkt/ Kreuzprodukt ($R \times S$)	–		–	
Join**				
Theta Join ($R\ \theta\ S$)	–		–	
Equi Join (R eq S)	–		–	
Natural Join ($R * S$)	inner join: **Join** bzw. **BGP** (durch Verkettung von Tripelmustern)	GM	Join-Prädikate	Q/ FE
Outer Join (left:$R \ltimes S$;right:$R \rtimes S$;full:$S \bowtie R$)	left outer join: **LeftJoin**	GM	–	
Projektion ($\pi_\alpha(R)$)	**BGP** (durch Variablenauswahl)	GM	**<PropertyName>** (Achtung: verpflichtende Schemaelemente sind stets Teil des Ergebnisses)	Q
	Project	EM		
Selektion ($\sigma_\beta(R)$)	**BGP** (durch Tripelmuster)	GM	alle Filteroperatoren: logische, vergleichende, räumliche und temporale Operatoren	FE
	Graph	GM		
	Filter (alle Filterausdrücke, spez. boolscher Operator '**&&**')	F/ GM		
	Distinct, **Slice**, **Reduced**	EM	*FE1.1:* Attribut **maxFeatures** *FE2.0:* Attribute **startIndex**, **count**	Q
Umbenennung ($\rho_\gamma(R)$)	implizit mit Variablennamen	GM	–	
	ab SPARQL1.1: per Select Expressions	EM		
Spezielle Operatorenfamilien				
Sortierende Operatoren	**OrderBy**	EM	**<SortBy>**	Q
Räumliche Operatoren (Filter/Join-Operatoren)	*Lücke wird durch GeoSPARQL-Filterausdrücke geschlossen*	F	<Equals>, <Disjoint>, <Touches>, <Within>, <Overlaps>, <Crosses>, <Intersects>, <Contains>, <DWithin>, <Beyond>, <BBox>	FE
Temporale Operatoren (Filter/Join-Operatoren)	–		*ab FE2.0:* <After>, <Before>, <Begins>, <BegunBy>, <TContains>, <During>, <TEquals>, <TOverlaps>, <Meets>, <OverlappedBy>, <MetBy>, <EndedBy>, <AnyInteracts>	FE

die Basiseinheit *Feature* jedoch nach wie vor bestehen bleibt. Dies trifft auch auf Join-Operationen zu, die anhand attributiver Kriterien mehrere Features als sogenannte *Join-Members* in Tupel gruppieren.

SPARQL-Endpoints operieren ebenfalls zumeist auf relationalen Datenbanken (*RDF-Triplestores*). Die Speicherform und damit auch die Auswertestrategie unterscheidet sich jedoch grundlegend von jener der WFS-Dienste. RDF-Triplestores speichern RDF-Tripel meist in einer Haupttabelle, und zwar Tripel-**S**ubjekt, -**P**rädikat und -**O**bjekt jeweils getrennt in einer Tabellenspalte, d.h. ein RDF-Tripel bildet als SPO-Tupel eine Tabellenzeile. Der erste SPARQL-Auswerteschritt führt *Graphenvergleiche* auf den SPO-Tupeln durch, die die SPO-Tupel mit relationalen `Join`-Operationen in Beziehung setzen (vgl. Abschnitt 2.2.3; siehe hierzu auch das Auswertebeispiel in Anhang A). Aus den Graphenvergleichen, die über Graphenmuster aus SPARQL-Algebra `BGP`-Operatoren eingeleitet werden, gehen neue Informationseinheiten hervor: *Ergebnistupel* (*Solution Sequence*) in einer RDF-Relation (Begriffe aus Cyganiak [2005]). Eine RDF-Relation ist eine neustrukturierte temporäre Tabelle, deren Spalten aus den im BGP-Operator verwendeten Variablen gewonnen werden. Jeder weitere SPARQL-Algebra Operator, z.B. `Join` oder `Union`, filtert relational auf den Ergebnistupeln der temporären RDF-Relationen. Es lässt sich also feststellen, das während einer SPARQL-Auswertung neue Basiseinheiten, die besagten Ergebnistupel, entstehen, die sich in ihrer Speicherung von den ursprünglichen SPO-Tupeln unterscheiden. Dieses ambivalente Verhalten von SPARQL ist problematisch für die richtige Interpretation und Anwendung der Anfragesprache. Im Gegensatz dazu bleibt die Basiseinheit eines Features bei allen WFS-Filterprozessen stets gewahrt.

Ergebnisaufbereitung

Eine WFS GetFeature-Anfrage kann mit einigen wenigen Ergebnismodifikatoren ausgestattet werden, um z.B. die Ergebnismenge (Feature Collection) zu sortieren, in der Anzahl ihrer Features zu begrenzen oder eine bestimmte Auswahl an Feature-Attributen vorzunehmen. Die Wahlmöglichkeiten, Feature-Attribute explizit als Teil des Ergebnisses zu bestimmen (relationaler Operator: `Projektion`) beschränken sich leider auf optionale Attribute. Diejenigen Feature-Attribute, die nach dem zugrunde liegenden GML-Applikationsschema verpflichtend sind, müssen laut WFS-Spezifikation immer in einer GetFeature-Antwort enthalten sein, was zu teils unerwünscht großen GML-Ausgaben führt - nicht zuletzt wegen umfangreicher Geometriedaten. Abgesehen davon ist eine Ergebnismodifikation zeitlich klar von den vorher ablaufenden Filtervorgängen abgegrenzt, was die Auswertung von GetFeature-Anfragen insgesamt gesehen leicht verständlich macht.

SPARQL unterstützt deutlich mehr Ergebnismodifikatoren. So können beispielsweise Ergebnisvariablen frei benannt (relationaler Operator: `Umbenennung`) oder doppelt vorkommende Elemente aus der Ergebnissequenz herausgefiltert werden (SPARQL-Operator `Distinct`). Anstelle der Features in einer GML Feature Collection, liefern SPARQL-Ergebnisse per se keine vorgefertigten semantischen Kompositionen, woraus Vor- und Nachteile erwachsen. Das Formulieren von SPARQL-Anfragen ist dadurch erschwert, dass Variablen innerhalb von Graphenmustern geschickt eingesetzt werden müssen, um die *lose Faktensammlung*, bestehend aus sich referenzierenden RDF-Tripeln, in einen sinnvollen Sachzusammenhang zu bringen. Nur die mit Variablen adressierten Inhalte, ob in obligatorischen oder optionalen Graphenmustern, sind letztlich Bestandteil der Ergebnistupel. In letzter Konsequenz muss der SPARQL-Anwender gleichzeitig eine inhaltliche Ergebniszusammenstellung und -Filterung betreiben und dies mittels ineinander verschachtelten SPARQL-Operatoren ausdrücken, was eine gute Kenntnis über das betreffende Datenschema voraussetzt. Andererseits ist der Anwender nicht begrenzt durch etwaige Spezifikationsvorschriften und besitzt alle Freiheiten in der Wahl und Verschneidung von Ergebnisinhalten.

Schlussfolgerung und Lösungsvorschlag

Die Gegenüberstellung lässt die Schlussfolgerung zu, dass sich beide Sprachen deutlicher voneinander unterscheiden, als es die Vergleichstabelle 5.4 erkennen lässt. Im Wesentlichen manifestiert sich der Unterschied in den abweichenden

Informationseinheiten, die sich auf Filtervorgänge und Prozesse der Ergebnisaufbereitung auswirken und somit sprachbestimmend sind. Hinsichtlich der Konzeption eines OWS-Proxy sind einzig diejenigen SPARQL-Sprachmerkmale von besonderem Interesse, die in FE entweder keine oder nur eine weniger ausdrucksstarke Entsprechung haben. Denn schließlich muss zur transparenten Integration von INSPIRE-Datenquellen im Semantic Web eine unidirektionale Übersetzung von SPARQL auf FE geleistet werden und nicht umgekehrt. Die Übersetzung birgt einige Tücken, allen voran die durch SPARQL ermöglichte feinergranulare Informationssuche und Objektverschneidung. Desweiteren sind einige SPARQL-Funktionalitäten speziell auf das Semantic Web ausgelegt und nicht auf FE übertragbar, wie z.B. das Prüfen von Blank Nodes (Funktion `isBLANK()`), vorhandenen Variablenbelegungen (Funktion `BOUND()`) oder dem Vergleich von RDF-Datentypen (z.B. per Funktion `isLITERAL()`). Doch selbst allgemeine Funktionalitäten, wie z.B. das Verwenden regulärer Ausdrücke (per Funktion `regex()`) lässt das FE-Repertoire vermissen. Problematisch sind darüber hinaus SPARQL-Ergebnismodifikatoren, die WFS-Query funktional nicht unterstützt. All diese Übersetzungshürden lassen den Schluss zu, dass eine vollständige, lückenlose Übersetzung nicht realisierbar ist. Neben dem Verlust an SPARQL-Filterinformationen ist obendrein darauf zu achten, dass aus den originären SPARQL-Anfragen keine restriktiveren FE-Ausdrücke abgeleitet werden, was insbesondere mit optionalen SPARQL-Graphenmustern der Fall sein kann.

Damit trotz genannter Sprachbarrieren eine Abbildung realisiert werden kann, wird ein **pragmatischer Lösungsweg** gewählt. Er zeichnet sich durch ein **doppelstufiges Auswerteverfahren** aus. Im ersten Schritt wird die SPARQL-Anfrage - so weit es die FE-Sprachmittel zulassen - in einen FE-Ausdruck bzw. einen WFS GetFeature-Aufruf übersetzt und dieser gegen alle anfragerelevanten INSPIRE-Downloaddienste gerichtet. Die INSPIRE GML-Ergebnisse, die der OWS-Proxy zurückerhält, werden sodann transformiert in RDF/OWL-Repräsentationen und mit der initial vom Benutzer abgeschickten SPARQL-Anfrage durchsucht. Die Verkettung einer anfänglich eher grobfilternden WFS GetFeature-Anfrage mit einer nachgelagerten feineren SPARQL-Auswertung ist in mehrfacher Hinsicht vielversprechend. Die dabei erzielten positiven Effekte sind:

- die Sprachabbildung von SPARQL auf FE kann sich auf die Filterfunktionalität konzentrieren, die die Verringerung der potentiellen GML-Ergebnismenge bewirkt, ohne dabei Ergebnismodifikatoren wie z.B. Sortierkriterien beachten zu müssen. Das Hauptaugenmerk liegt hierbei auf der Vermeidung übermäßigen Datentransfers.
- trotz der Integration von Geo Web-Datenquellen und den damit einhergehenden inkompatiblen Umgebungsbedingungen kommen die Semantic Web-eigenen *Informationseinheiten* (in RDF/OWL-Instanzen transformierte GML-Ergebnisse) sowie *Filtervorgänge* mitsamt Graphenvergleichen in der nachgelagerten SPARQL-Auswertung zur Geltung.
- eine Begrenzung auf einige wenige SPARQL-Operatoren kann vermieden werden. Die Ergebnismodifikatoren (*Solution Modifier*) und SPARQL-Anfrageformen (*Query forms*) kommen zur Anwendung. Ebenso stehen die speziell auf Semantic Web-Speicherformen ausgelegten SPARQL-Funktionen (`isBLANK()` etc.) uneingeschränkt zur Verfügung.
- zusätzlich zu den behördlichen INSPIRE GML-Ergebnisdaten lassen sich Benutzerinformationen (Stichwort: *Social Tagging*) oder Linked Data-Referenzen mit INSPIRE-Instanzen verlinken und in einer gemeinsamen SPARQL-Auswertung zu einem Ergenis verschmelzen (vgl. Abschnitt 5.2.4).

Abbildungsregeln

Im obigen Absatz wird angedeutet, dass sich die Sprachabbildung auf die Filterfunktionalität konzentriert. In SPARQL wirken **Filter in unterschiedlichen Geltungsbereichen**: 1. den Graphenmustern (SPARQL-Algebra: *BGP*-Operator) und 2. den Filterausdrücken (SPARQL-Algebra: enthalten in *Filter*- und *LeftJoin*-Operatoren); siehe auch Tabelle 5.4 Spalte 3 *Geltungsbereich*). Es ist jedoch im Hinblick auf die anschließenden Abbildungsregeln

erforderlich, eine weitere begriffliche Unterscheidung einzuführen, die die SPARQL-Filtereigenschaften etwas konkreter charakterisiert. Es wird differenziert in:

- **Objektwertvergleiche**: vergleichen eine Ressource bzw. ein Literal mit einem konkreten Wert, womit eine URI, ein Zahlen- oder Textwert gemeint ist. Sie treten sowohl in Graphenmustern (z.B. `?siteDesignation ps:percentageUnderDesignation "100"^^xsd:float .`) als auch in Filterausdrücken auf (z.B. `Filter(?percentage = "100"^^xsd:float)` bzw. `Filter(?percentage < "50"^^xsd:float)`)
- **Pfadmustervergleiche**: vergleichen Knoten-Kanten-Muster in einem Graphen (Prädikatpfade), ohne dabei Objektwertvergleiche anzustellen. Sie treten nur in Graphenmustern auf (z.B. `?protectedSite ps:siteManagementPlan ?plan . ?plan ps:siteManagementPlanReference ?referenceDocument .`)

Die **Abbildungsregeln von SPARQL nach Filter Encoding** sind:

- die **Objektwertvergleiche** aus SPARQL-Filterfunktionen sind entsprechend der Tabelle 5.5 in FE-Ausdrücke zu übersetzen. Objektwertvergleiche aus Graphenmustern, die z.B. Codelist-Werte oder konstante Literale abfragen, prüfen nur auf einen Wertevergleich und sind deswegen auf den FE-Operator `<PropertyIsEqualTo` abzubilden.
- die **Pfadmustervergleiche** tauchen nur in Graphenmustern auf und überprüfen das Vorhandensein von Prädikatpfaden/ Knoten-Kanten-Beziehungen und damit verbundener Detailinformationen in einem Graphen. Die Prüfung lässt sich prinzipiell auch auf WFS-Datenquellen bzw. GML-Features übertragen, wobei anstelle der Prädikatpfade die abgeleiteten GML-Elementpfade auf jene Stellen verweisen, an denen GML-Attributinformationen eingefordert werden. Die Prüfung erfolgt mit der Kombination der FE-Operatoren `<Not>` und `<PropertyIsNull>`. Falls der Datenbestand ein Feature mit dem nachgefragten Attribut enthält, ergibt der Vergleichstest mit `<PropertyIsNull>` ein negatives Zwischenresultat (*false*), das mit dem umschließenden `<Not>` positiviert wird (*true*) und damit das Vorhandensein des Attributes bestätigt. Das geprüfte Feature ist somit Teil der Lösungsmenge.
- die **SPARQL-Operatorenhierarchie** ist zu befolgen und in einen vergleichbar strukturierten FE-Ausdruck zu übersetzen. Die diesbezüglichen Umformungsmodalitäten sind in Tabelle 5.6 wiedergegeben. Die innersten SPARQL Algebra-Operatoren sind stets `BGP`-Operatoren für Graphenmustervergleiche. In der Operatorenhierarchie in Codebeispiel 18 sind dies die Operatoren `BGP1`, `BGP2`, `BGP3` und `BGP4` (Zeilen 5, 6, 9 und 10). Für die genannten Operatoren wird die Abbildungsregel `BGP(`*`Tripelmuster`*`)` aus Tabelle 5.6 angewendet, d.h. es erfolgt eine Übersetzung aller in der SPARQL-Anfrage enthaltenen Objektwert-/ Pfadmustervergleiche auf entsprechende FE-Operatoren und die anschließende Verknüpfung der FE-Operatoren mit `<AND>`. Auf der nächsten Hierarchiestufe ist beispielsweise `LEFTJOIN1(BGP1, BGP2, <NULL>)` auszuwerten (Codebeispiel 18; Zeile 4), auf der übernächsten `FILTER1("?beginLifespan > ...", LEFTJOIN1)` (Codebeispiel 18; Zeile 3) usw.
- die **Sichtbarkeit von Variablen**, die in den Graphenmustern eingeführt und benannt werden, bestimmt sich aufgrund ihrer Platzierung in der SPARQL-Operatorenhierarchie. Je nach Level und Hierarchiepfad stehen Variablen auch übergeordneten SPARQL-Operatoren und -Filterausdrücken zur Verfügung (siehe Abschnitt 5.2.1; Absatz *GML-Elementadressierung*).
- **optionale Graphenzweige** der SPARQL-Anfrage bleiben unberücksichtigt, um das FE-Ergebnis der Sprachabbildung nicht restriktiver zu machen als seine SPARQL-Quellinformation. Die Regelung bezieht sich lediglich auf den SPARQL-Operator `LeftJoin`, dessen optionaler Zweig `P2` als auch die darauf Einfluss nehmenden Filterausdrücke `F` für die Sprachabbildung ignoriert werden; siehe Tabelle 5.6.

Tabelle 5.5 – Leitpfaden zur Abbildung von SPARQL-Filterausdrücken auf Filter Encoding
(* A, B: Werte in SPARQL-Filterausdrücken; ** verwendbar innerhalb eines FE-Vergleichsoperators)

SPARQL-Filterausdrücke * (eingeschränkt auf SPARQL 1.0)	**Abbildung auf Filter Encoding (Filter Encoding 1.1 & 2.0)**
Unäre SPARQL-Filterausdrücke	
!*A*	FE-Operator <NOT> in Kombination mit einem FE-Vergleichsoperator
+*A*	arithmetischer FE-Ausdruck <ADD> mit Literal = 1 **
-*A*	arithmetischer FE-Ausdruck <SUB> mit Literal = 1 **
BOUND(*A*)	keine Anwendung im WFS-GetFeature Aufruf
isIRI(*A*)	”
isURI(*A*)	”
isBLANK(*A*)	”
isLITERAL(*A*)	”
STR(*A*)	falls A einem Literal gleicht, wird der Operator übergangen (STR(A) → A); keine Auswertung, falls A vom Typ URI ist
LANG(*A*)	keine Anwendung im WFS-GetFeature Aufruf
DATATYPE(*A*)	”
Binäre SPARQL-Filterausdrücke	
A **||** *B*	FE-Operator <Or>, ggf. werden A und/oder B jeweils mit einem FE-Operator <And> gruppiert
A **&&** *B*	FE-Operator <And>, ggf. werden A und/oder B jeweils mit einem FE-Operator <And> gruppiert
A = *B*	FE-Operator <PropertyIsEqualTo>
A **!=** *B*	FE-Operator <PropertyIsNotEqualTo>
A < *B*	FE-Operator <PropertyIsLessThan>
A > *B*	FE-Operator <PropertyIsGreaterThan>
A <= *B*	FE-Operator <PropertyIsLessThanOrEqualTo>
A >= *B*	FE-Operator <PropertyIsGreaterThanOrEqualTo>
A * *B*	arithmetischer Ausdruck <MUL> **
A / *B*	arithmetischer Ausdruck <DIV> **
A + *B*	arithmetischer Ausdruck <ADD> **
A - *B*	arithmetischer Ausdruck <SUB> **
A = *B* (basierend auf RDF-Term)	keine Anwendung im WFS-GetFeature Aufruf
A **!=** *B* (basierend auf RDF-Term)	”
sameTerm(*A*, *B*)	ggf. per FE-Operator <PropertyIsEqualTo>
langMatches(*A*, *B*)	keine Anwendung im WFS-GetFeature Aufruf
regex(*Analysetext*, *regulärer Ausdruck*)	FE-Operator <PropertyIsLike> (Achtung: Filter Encoding erlaubt nur Textvergleiche mit *Wildcard*-Zeichen, keine *regulären Ausdrücke*)
Ternäre SPARQL-Filterausdrücke	
regex(*Analysetext*, *regulärer Ausdruck*, *Flags*)	FE-Operator <PropertyIsLike> (Achtung: Filter Encoding erlaubt nur Textvergleiche mit *Wildcard*-Zeichen, keine *regulären Ausdrücke*)

Tabelle 5.6 – Leitpfaden zur Abbildung von SPARQL-Algebra Operatoren auf Filter Encoding
(* P, P1, P2: Algebra-Graphenmuster; F: Filterausdrücke; L: Ergebnissequenz aus SPARQL-Ergebnistupeln)

SPARQL Algebra-Operatoren * (eingeschränkt auf SPARQL 1.0)	Abbildung auf Filter Encoding (Filter Encoding 1.1 & 2.0)
Algebra-Graphenmuster	
BGP(*Tripelmuster*)	1. Abbildung der in den Tripelmustern enthaltenen Objektwert-/Pfadmustervergleiche auf entsprechende FE-Operatoren (siehe Tabelle 5.5) 2. Verknüpfung der FE-Operatoren mit FE-Operator **<And>**
Join(*P1*, *P2*)	Verknüpfung der aus P1 und P2 resultierenden FE-Operatoren mit FE-Operator **<And>**
LeftJoin(*P1*, *P2*, *F*)	*der optionale Zweig P2 und eventuelle Filterausdrücke in F werden nicht berücksichtigt, deshalb können alle aus P1 resultierenden FE-Operatoren unverändert angewendet werden*
Filter(*F*, *P*)	Verknüpfung der aus F und P resultierenden FE-Operatoren mit FE-Operator **<And>**
Union(*P1*, *P2*)	Verknüpfung der aus P1 und P2 resultierenden FE-Operatoren mit FE-Operator **<Or>**
Graph(*IRI*, *P2*)	*dieser SPARQL-Operator wird nicht berücksichtigt, vielmehr geschieht die Zugehörigkeit der Vergleichsmuster zu ihrem Kontext bzw. Graphen im vorangegangenen Analyseschritt (siehe Abschnitt 5.2.1)*
Algebra-Ergebnismodifikatoren	
ToList(*P*)	*alle Ergebnismodifikatoren werden nicht innerhalb eines WFS GetFeature-Aufrufes, sondern erst im Zuge der späteren SPARQL-Anfrage angewendet*
OrderBy(*L*, *Bedingung*)	”
Project(*L*)	”
Distinct(*L*)	”
Reduced(*L*)	”
Slice(*L*, *Beginn*, *Anzahl*)	”

Anwendungsbeispiel

Eine exemplarische Sprachabbildung soll nun die aufgestellten Abbildungsregeln entsprechend veranschaulichen. Als Quellinformation dient die SPARQL-Anfrage aus Codebeispiel 16, die in eine WFS Get Feature-Anfrage inklusive FE-Filterausdrücke in Codebeispiel 19 überführt wird. Die SPARQL-Anfrage bezweckt die Filterung von Schutzgebieten. Das zugehörige INSPIRE-Ontologiekonzept lautet `ont_ps:ProtectedSite` und entspricht den beiden WFS Feature Types `ps:ProtectedSite` und `ps-f:ProtectedSite`. Jeder WFS Feature Type erfordert ein `wfs:Query`-Element, eingebettet in die WFS Get Feature-Anfrage (Element: `wfs:GetFeature`). Beide `wfs:Query`-Elemente gleichen sich bezüglich ihrer FE-Filterausdrücke bis auf die Verwendung der GML-Property `ps-f:beginLifespanVersion`, die aus dem OWL-Prädikat `ont_ps:beginLifespanVersion` des SPARQL-Graphenmusters hervorgeht. Da die GML-Property nur im INSPIRE-Datenmodell ProtectedSites-Full spezifiziert ist (XML-Präfix: `ps-f`), darf lediglich die WFS-Query des Feature Types `ps-f:ProtectedSite` eine Filterung über `ps-f:beginLifespanVersion` vornehmen. Zu beachten ist auch die FE-Operatorenhierarchie mit ineinander verschachtelten `<ogc:Or>`- und `<ogc:And>`-Operatoren, die aus der Abbildung der SPARQL Algebra-Ausdrücke gemäß Tabelle 5.6 hervorgeht. Die Pfadmustervergleiche in den Zeilen 12 und 22 der SPARQL-Anfrage werden in die Operatorenkombinationen `<ogc:Not>` und `<ogc:PropertyIsNull>` überführt. Optionale Zweige in den Zeilen 14/15 sowie 25/26 der SPARQL-Anfrage finden in den FE-Ausdrücken keine Anwendung.

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:gml="http://www.opengis.net/gml"
    xmlns:ogc="http://www.opengis.net/ogc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
    xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
    xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"  version="1.1.0"
    resultType="results" outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">
  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
      typeName="ps-f:ProtectedSite" srsName="CRS:84">
    <ogc:Filter >
      <ogc:Or>
        <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:And>
        <ogc:And>
          <ogc:Not>
            <ogc:PropertyIsNull>
              <ogc:PropertyName>ps:geometry</ogc:PropertyName>
            </ogc:PropertyIsNull>
          </ogc:Not>
          <ogc:Within>
            <ogc:PropertyName>ps:geometry</ogc:PropertyName>
            <gml:Polygon srsName="CRS:84">
              <gml:exterior>
                <gml:LinearRing>
                  <gml:posList>18 47 18.1 47 18.1 49 18 49 18 47</gml:posList>
                </gml:LinearRing>
              </gml:exterior>
            </gml:Polygon>
          </ogc:Within>
        </ogc:And>
      </ogc:Or>
    </ogc:Filter>
  </wfs:Query>
  <wfs:Query xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
      typeName="ps:ProtectedSite" srsName="CRS:84">
    <ogc:Filter>
      <ogc:And>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:geometry</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:Within>
          <ogc:PropertyName>ps:geometry</ogc:PropertyName>
          <gml:Polygon srsName="CRS:84">
            <gml:exterior>
              <gml:LinearRing>
                <gml:posList>18 47 18.1 47 18.1 49 18 49 18 47</gml:posList>
              </gml:LinearRing>
            </gml:exterior>
```

```
            </gml:Polygon>
          </ogc:Within>
        </ogc:And>
      </ogc:Filter>
    </wfs:Query>
  </wfs:GetFeature>
```

Codebeispiel 19 – WFS GetFeature-Anfrage als Ergebnis der Sprachabbildung

5.2.3 Verteilte GetFeature-Anfragen an WFS-Dienste

Die einleitenden Abschnitte 5.2.1 und 5.2.2 haben dargelegt, mit welcher Methodik SPARQL-Anfragen hinsichtlich ihrer INSPIRE-relevanten Anfragebestandteile ausgelesen und in WFS GetFeature-Anfragen umgewandelt werden können. Damit steht fest, nach *was* bzw. nach welchen Inhalten und *wie* bzw. über welche attributiven Einschränkungen gesucht wird. Der Programmablauf aus Abbildung 5.9 ist jetzt bei Schritt 4 angelangt und es ist der Frage nachzugehen, *wo* die angeforderten Inhalte aufgefunden werden können. Für die verteilte Suche (*Federation*) über INSPIRE-Datenquellen ist der OWS-Proxy mit Informationen zu den verfügbaren Downloaddiensten und deren Merkmalen auszustatten. Im Wesentlichen handelt es sich dabei um die Kontaktadressen und Themenbezüge der Downloaddienste, die in einem **Ressourcen-Repository** registriert werden.

Die Abbildung 5.12 zeigt ein für das Ressourcen-Repository in Frage kommendes Datenmodell. Auf der linken Seite sind die Modellobjekttypen und auf der rechten Seite beispielhafte Objektinstanzen dargestellt. Der Objekttyp namens `DownloadService` dient zur Speicherung der Dienstadresse (`serviceUrl`), der WFS-Ausprägung bzw. -Version (`serviceType`) und beschreibender Metainformationen für Beschriftungszwecke in Benutzeroberflächen (`name` und `description`). Ein `DownloadService` wird mit 1-n INSPIRE-Themenbezügen (`AnnexTheme`) konfiguriert. Ein `AnnexTheme` ist abgesehen von gleichartigen Metainformationen und dem Namen des betreffenden INSPIRE Annex-Themas `inspireName` mit drei relevanten Filterkriterien ausgestattet. Die im Downloaddienst für jedes Thema zugeordneten INSPIRE-Namensräume `inspireNamespace` erlauben eine Vorfilterung der Datenquellen über Identifizierungsmerkmale, die sich im Kontext des URI-Resolving von INSPIRE Semantic Web-Instanzen als nützlich erweist (siehe Abschnitt 5.1.2 *Identifikationsmanagement*). Das Attribut `ontologyNamespace` referenziert auf die jeweilige INSPIRE-Themenontologie, fungiert also als Bindeglied zwischen dem Ressourcen-Repository und den Themenontologien. Diese Verlinkung dient der thematischen Vorfilterung der Datenquelle, indem sie darüber Auskunft gibt, welche ontologischen Konzepte aus einem Downloaddienst abgeleitet werden können und über welche GML/OWL-Abbildungsregeln dies geschieht. Das letzte Filterkriterium (`hasSpatialExtent`) speichert die räumliche Ausdehnung (Objekttyp `SpatialExtent`) eines Datenthemas mittels Bounding Box-Koordinaten. Die räumliche Ausdehnung ist sinnvollerweise auch themenübergreifend für einen `DownloadService` konfigurierbar.

Unter Verwendung des Ressourcen-Repository ist der OWS-Proxy in der Lage, diejenigen Dienste zu bestimmen, die aus thematischen, räumlichen und objektidentifizierenden Gesichtspunkten geeignet sind, Informationen zur Beantwortung der SPARQL-Anfrage beizusteuern. Sofern mehrere Downloaddienste brauchbare Ergebnisse liefern können, ist auch die Reihenfolge der einzelnen Dienstaufrufe festzulegen. Ein **Ablaufplan** (engl. *Query Execution Plan*) ist hilfreich, um anhand von Querbeziehungen zwischen den Datenquellen die Dienstaufrufe derart hintereinander zu takten, dass zwischenzeitliche Anfrageergebnisse gewinnbringend - zur Minderung der Netzlast und der Latenzzeiten - in noch ausstehende Dienstaufrufe einfließen können und sich deren Anfragefilter verfeinern lassen. Ob mit oder ohne Berücksichtigung eines Ablaufplanes sollten so viele Aufrufe wie möglich parallelisiert vonstattengehen und asynchron abgearbeitet werden (Stichwort: *Multithreading*).

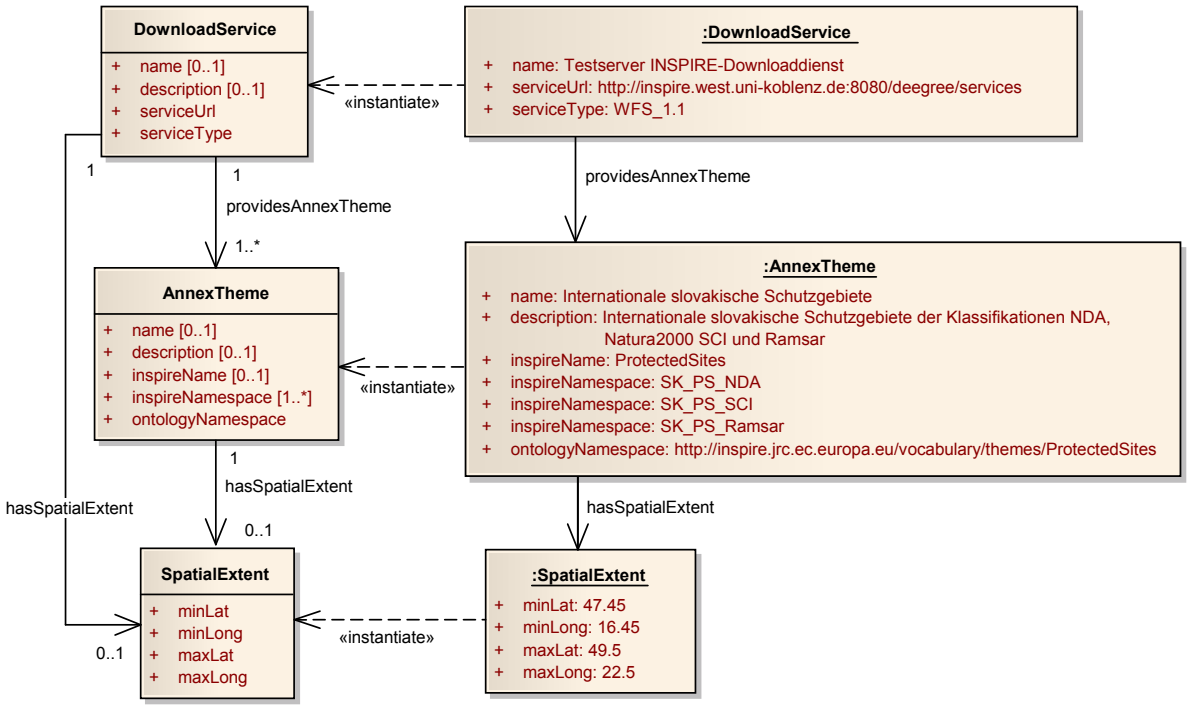


Abbildung 5.12 – Datenmodell und Instanzbeispiele eines Ressourcen-Repository

Eine automatische Ablaufplanung zu konzipieren kommt einem eigenständigen Arbeitsthema gleich. Anstelle einer ausführlichen Behandlung werden hier nur einige Gestaltungshinweise gegeben. So rühren beispielsweise Querbeziehungen zwischen Datenquellen daher, dass innerhalb einer SPARQL-Anfrage mehrere SPARQL-Variablen, die über Prädikatpfade zueinander in Beziehung stehen, sich in unterschiedliche Feature Types übersetzen lassen (z.B. im Tripelmuster: `?protectedSite ont_ps:isInRegion ?biogeographicalRegion`). Um die Aufrufvorgänge zu optimieren, könnte eine erste Anfrage nur dem übergeordneten Feature Type gelten (hier im Beispiel: `ps-f:ProtectedSite`). Die zurückgelieferte Ergebnismenge enthält aller Wahrscheinlichkeit nach GML-Referenzen auf die Features des nachgelagerten Feature Types (hier: `bgr:Bio-GeographicalRegion`). Handelt es sich bei den GML-Referenzen um *inline*-deklarierte oder im Ergebnisdokument intern verlinkte Features (interne XLink-Referenz), dann liegen bereits alle gesuchten Inhalte im OWS-Proxy zur weiteren Auswertung vor. Stattdessen sind externe GML-Referenzen (externe XLink-Referenzen) aufzulösen, indem mehrere Features abhängig von ihrer Speicheradresse entweder in einem gebündelten GetFeature- oder in einzelnen GetGmlObject-Aufrufen angefordert werden. Eine weitere Optimierungsstrategie schaltet vor den eigentlichen Datendownloads je einen Kontrollaufruf, der die Anzahl der jeweiligen Ergebnis-Features erfragt (WFS GetFeature-Anfrage mit Attributbelegung `resultType="hits"`). Aufgrund dieser Kenntnis können statistische Auswertungen hinsichtlich der potentiellen Verschneidung von Objektmengen durchgeführt und ein *Königsweg* gewählt werden, um die resultierenden Verschneidungsergebnisse, schlimmstenfalls Kreuzprodukte, möglichst gering zu halten. Die Anzahl der resultierenden Features könnte auch als Abbruchbedingung dienen, die zusammen mit einer definierten maximalen Dienstantwortzeit dabei einschätzen hilft, ob ein Benutzer möglicherweise eine zu allgemein formulierte SPARQL-Anfrage gestellt hat und zu einer Präzisierung seiner Anfragebedingungen aufgefordert werden sollte. Letztlich ist also die Erstellung eines Ablaufplanes zu empfehlen, dessen weitere konzeptuelle Ausgestaltung jedoch zukünftigen Arbeiten vorbehalten bleiben muss.

Eine kurze Exkursion soll abschließend auf eine Problematik aufmerksam machen, die abhängig von Spezifika der jeweiligen WFS-Datenhaltung eine automatisierte Anfrageplanung erschwert. Ein WFS-Client kann eine WFS GetFeature-Anfrage mit einer Reihe an Steuerungsparametern beeinflussen, die im Element `wfs:GetFeature` an-

zugeben sind (siehe Codebeispiel 19). Für konkrete Datenabrufe können viele der **Steuerungsparameter** mit Standardwerten vorbelegt werden, so z.B. die WFS-Version (`version="1.1.0"`), der Ergebnistyp für die Datenausgabe (`resultType=results"`) und das Ausgabeformat (`outputFormat=text/xml; subtype=gml/3.2.1"`). Eine gewisse Einflussnahme erlauben dagegen die Parameter `traverseXlinkDepth` und `traverseXlinkExpiry`. Sie geben die Verschachtelungstiefe und die maximale Antwortzeit vor, an die sich ein WFS-Dienst zur Auflösung lokaler oder externer **GML-Objektreferenzierungen der Quelldaten** zu halten hat, um referenzierte GML-Objekte frei von Verlinkungen in die WFS-Ergebnismenge aufzunehmen. Die INSPIRE-Datenmodelle machen regen Gebrauch von GML-Objektreferenzierungen, weshalb insbesondere der Parameter `traverseXlinkDepth` an Bedeutung gewinnt. Da ein Anwender üblicherweise keine tiefergehenden Kenntnisse zu einer fremden INSPIRE-Datenhaltung besitzt und jene selbst sehr heterogen ausfallen kann, indem Objektinstanzen mal *inline*-deklariert und mal per XLink-Referenzen eingebunden sind, werden bei fest vorgegebener Verschachtelungstiefe (z.B. `traverseXlinkDepth="2"`) gegebenenfalls nicht alle gewünschten Dateninhalte mit der WFS-Ergebnismenge transportiert. Weitere WFS-Anfragen sind dann vonnöten. Der Anwender hat die Option, die Verschachtelungstiefe unbestimmt zu lassen (`traverseXlinkDepth="*"`), riskiert jedoch damit, dass im Falle gut vernetzter Informationen zu viele Referenzen aufgelöst und die WFS-Ergebnismenge ins schier Unermessliche wächst. Für einen technischen WFS-Client, wie den OWS-Proxy, erschwert die unzureichende Kontrolle über die Zusammenstellung der WFS-Ergebnisse leider in erheblichem Maße die Anfrageplanung und damit indirekt eine automatisierte Ergebnisverarbeitung. Ein optimales Mittelmaß bzw. einen geeigneten Standardwert für `traverseXlinkDepth` zu finden, ist angesichts der INSPIRE-Datenvielfalt keine leichte Aufgabe. Die Testläufe, über die der Abschnitt 5.3.4 berichtet, wurden der Einfachheit halber mit der Einstellung `traverseXlinkDepth="*"` durchgeführt.

5.2.4 Aufbereitung von WFS-Ergebnissen

Die INSPIRE-Downloaddienste senden ihre GetFeature-Antworten als GML-Instanzdokumente zurück an den OWS-Proxy. Ein exemplarisches GML-Instanzdokument, das GML-Instanzen zu INSPIRE-Schutzgebieten beinhaltet, ist dem Codebeispiel 4 zu entnehmen. Gemäß der gewählten Auswertestrategie (siehe Einleitung zu Abschnitt 5.2) setzt sich der Programmablauf der Anfragebearbeitung mit folgenden Schritten fort:

1. **Ergebnistransformation von GML nach RDF/OWL**: die GML-Ergebnisse werden mit Hilfe von XML-Techniken ausgelesen und in RDF/OWL-Ressourcen bzw. -Aussagen überführt
2. **Ergebnisannotation/-verlinkung mit externen Quellen**: die resultierenden RDF/OWL-Ressourcen lassen sich z.B. mit bestehenden Linked Data-Datentöpfen oder Benutzerinhalten verknüpfen.
3. **Speicherung in einem temporären Repository**: sowohl die abgeleiteten INSPIRE RDF/OWL-Ressourcen und -Aussagen als auch die semantischen Verlinkungen mit externen Quellen sind in ein temporäres Repository, d.h. in einen SPARQL-unterstützenden RDF-Triplestore, zu überführen. Gegen das temporäre, im Arbeitsspeicher befindliche Repository wird sodann die anfängliche SPARQL-Anfrage gerichtet (siehe Schritt 6 aus Abbildung 5.9).

Der letzte Prozessschritt ist hier nochmals der Vollständigkeit halber erwähnt, bedarf jedoch keiner weiteren inhaltlichen Ausgestaltung. Stattdessen konzentriert sich dieser Abschnitt in den nächsten Absätzen auf die beiden erstgenannten Prozesse.

Ergebnistransformation von GML nach RDF/OWL

GML ist eine XML-Auszeichnungssprache, demzufolge sind GML-Instanzdokumente wie XML-Dokumente mit einer XML-Software zu interpretieren und auszulesen, einem *XML-Parser*. Dabei stehen verschiedene Arten von XML-Parsern zur Auswahl, je nachdem, ob ein *sequentielles Lesen* aller oder ein *wahlfreier Zugriff* (engl. *random-access*) auf

ausgewählte Teile eines XML-Dokumentes erwünscht ist. Im vorliegenden Fall legt der Benutzer mit seiner SPARQL-Anfrage präzise fest, welche Inhalte er abrufen möchte. Und zwar verwendet er dazu SPARQL-Variablen und Prädikatpfade, die sich in GML-Elementpfade umwandeln lassen (siehe Abschnitt 5.2.1). Bei den GML-Elementpfaden handelt es sich um XPath-Ausdrücke, die bestimmte Teile eines GML-Instanzdokumentes adressieren, weshalb **XPath-fähige XML-Parser** XPath-Ausdrücke als Leseanweisung von XML-Dokumentbestandteilen verwenden können. Dank der GML-Elementpfade sind die gewünschten GML-Inhalte während der Anfragebearbeitung im OWS-Proxy bekannt und können gezielt aus den WFS-Ergebnissen ausgelesen werden (wahlfreier Zugriff). Ein sequentieller Lesevorgang sowie die speicherlastige Abbildung aller GML-Inhalte in RDF/OWL-Repräsentationen ist dadurch nicht erforderlich.

Es ist an dieser Stelle darauf hinzuweisen, dass auch **andere XML-Parsertechniken** durchaus einen wahlfreien und programmgesteuerten Elementzugriff ermöglichen (sogenannte *Pull-Parser*), ohne sich dabei auf XPath-Ausdrücke zu stützen. Eine weitere Alternative bietet das sogenannte *XML Data Binding*. Hierbei wird ein XML-Dokument vollständig in objektorientierte Programmstrukturen übertragen, deren Zugriffsmethoden nach den individuellen XML-Schemaelementen benannt sind und die dann wahlweise aus dem Programmcode heraus aufgerufen werden können. Die Realisierung mit XPath-Parsing ist allerdings aus zwei triftigen Gründen zu bevorzugen. Einerseits bietet XPath eine mächtige Sprachbasis zum Referenzieren von XML-Inhalten. So ist anwendungsbezogene Programmlogik, die im Falle des Einsatzes anderer Pull-Parser oder XML Data Binding hardkodiert werden muss, weitestgehend über XPath-Ausdrücke zu ersetzen, was die Konfiguration flexibler werden lässt und die Wartung des Quellcodes vereinfacht. Andererseits verwendet der OWS-Proxy bereits bei der Zusammenstellung von WFS GetFeature-Anfragen XPath-Ausdrücke. Diese im Zuge der Auswertung und -umformung von GetFeature-Antworten nachzunutzen vereinheitlicht zugleich beide GML-Zugriffsmechanismen.

Die Vorteile eines XPath-Parsers liegen demnach auf der Hand. Es ist jedoch noch ein wichtiger Aspekt zu beachten und die Frage aufzuwerfen, wie es sich mit der Speichereffizienz von XPath-Parsern verhält? Denn schließlich operieren INSPIRE-Downloaddienste auf komplexen GML-Datenmodellen. Konforme GML-Instanzdokumente können schnell eine Speichergröße von mehreren Megabyte annehmen. Aus diesem Grund werden WFS-Dienste, wie z.B. der Deegree-WFS (siehe Abschnitt 5.3.1) mittlerweile mit Streaming-Fähigkeiten ausgestattet, um den Speicherabdruck im WFS-Dienst während der Bearbeitung der GetFeature-Antwort möglichst gering zu halten. GetFeature-Antworten werden bei Streaming-Vorgängen nicht in vollem Umfang, sondern sukzessive in einem fortwährenden Datenstrom an den WFS-Client versandt. Dieser erhöhte Datentransfer muss auch in einem WFS-Client wie dem OWS-Proxy berücksichtigt werden. XPath-Parser, die auf sogenannten *DOM*-Parsern basieren, sind für ein vermehrtes Datenaufkommen weniger gut geeignet, da sie vor einem Leseprozess das zu behandelnde XML-Dokument speicherintensiv als Objektbaum (*Document object model*: DOM) im Arbeitsspeicher anlegen. Eine lohnenswerte Alternative sind **nicht-extrahierende Parser**, die anstelle eines Objektbaumes eine kompakte Indexdatei zum Indizieren aller XML-Strukturelemente und -Inhalte erzeugen. Die Methode führt zu einer geringeren Belastung des Arbeitsspeichers und erweist sich als deutlich performanter bei Datenzugriffen. Perspektivisch sollten deshalb nicht-extrahierende Parser zum Einsatz kommen.

Die Verwendung von XPath-Parsern ermöglicht einen ungebundenen, wahlfreien Elementzugriff. Wegen der damit gewonnenen Freiheitsgrade ist eine günstige **Lesestrategie** in der Anwendungslogik des OWS-Proxy festzulegen, mit der die Reihenfolge der einzelnen Datenzugriffe auf ein GML-Instanzdokument vorgegeben wird. Die Herausforderung besteht darin, mit a) möglichst wenigen Datenzugriffen b) alle geforderten Inhalte auszulesen und c) die inhaltlichen Relationen zu berücksichtigen bzw. verlustfrei von GML in RDF/OWL zu übertragen. Zur weiteren Argumentation hilft der Blick auf die GML-Elementpfade, die aus der SPARQL-Auswertung in Abbildung 5.11 resultieren. Bezogen auf einen Feature Type, z.B. ps-f:ProtectedSite, sind dies die Elementpfade:

1. `ps-f:ProtectedSite/ps-f:beginLifespanVersion`
2. `ps-f:ProtectedSite/ps:siteDesignation/ps:DesignationType`
3. `ps-f:ProtectedSite/ps:siteDesignation/ps:DesignationType/ps:designation`

Die GML-Elementpfade referenzieren alle GML-Inhalte, die der Anfragebearbeitung helfen. So ist die Vorgabe b), alle geforderten Inhalte auszulesen, dadurch zu erfüllen, dass alle GML-Elementpfade geparst werden. Die inhaltlichen Abhängigkeiten zwischen Informationsobjekten und Datenwerten im GML-Instanzdokument sind auf die XML-Elementhierarchie und im Speziellen auf das GML Object-Property-Muster zurückzuführen. In den oben aufgeführten GML-Elementpfaden zeigt sich die Abhängigkeit anhand gleicher Teilpfade, u.a. beginnen alle drei Pfade mit dem Knotentest bzw. XPath-Pfadschritt `ps-f:ProtectedSite`. Ebenso ist der zweite Elementpfad vollständig im dritten enthalten oder anders ausgedrückt, der letzte XPath-Pfadschritt des dritten Elementpfades (`ps:designation`) relativ zum zweiten. Abgesehen davon, dass sie in Relation zueinander gesetzt werden können, sind die GML-Elementpfade von den Prädikatpfaden der SPARQL-Anfrage hergeleitet (siehe Abschnitt 5.2.1). Daher repräsentiert ein relativer Elementpfad implizit eine Konzeptrelation aus den Themenontologien.

Die Idee ist nun, die XML-Elementhierarchie des GML-Instanzdokumentes mit relativen XPath-Ausdrücken rekursiv zu durchlaufen. Der bei jedem Rekursionsschritt abgegriffene Inhalt eines GML-Elementpfades wird unverzüglich in ein oder mehrere RDF/OWL-Instanzen umgewandelt, die wiederum der nächsten Rekursion als Information für ontologische Verknüpfungen übergeben werden. Dadurch ist Bedingung c) nach Beibehaltung inhaltlicher Bezüge erfüllt. Mittels der *rekursiven Lesevorgänge* wird ein beliebiges GML-Instanzelement, das einem GML-Elementpfad entspricht, nur ein einziges Mal aufgesucht und verarbeitet, wodurch auch der Vorgabe a) nach möglichst wenigen Datenzugriffen nachgekommen wird. Letztlich führt auch die Suche mit relativen Pfaden gegenüber der Suche ausgehend von absoluten XPath-Ausdrücken zu einer weiteren Performanzsteigerung, da die Suchbasis mit jedem relativem Pfadschritt sukzessive verkleinert wird.

Anhand des Pseudocodes in Codebeispiel 20 wird nun ein vereinfachter Auslese- und Transformationsvorgang demonstriert, der in komplexerer Form im Prototypen implementiert wurde (siehe Abschnitt 5.3.3). Die abgeleiteten RDF/OWL-Ressourcen werden dabei direkt in den erwähnten temporären RDF-Triplestore eingespeist, der in Zeile 1 initialisiert wird. Danach beginnt der Schleifendurchlauf über jeden Feature Type (Zeilen 2-15), der während der SPARQL-Anfragenanalyse detektiert wurde und nach erfolgtem GetFeature-Aufruf im GML-Ergebnisdokument zu erwarten ist, z.B. der Schutzgebietetyp `ps-f:ProtectedSite`. Als vorbereitende Maßnahme werden zuerst all diejenigen GML-Elementpfade ausgewählt (Zeile 3), alphabetisch und in aufsteigender Pfadlänge sortiert (Zeile 4), deren Pfad relativ ist zum aktuell in der Schleife behandelten Feature Type; z.B. ist `ps-f:beginLifespanVersion` ein relativer Pfad zum Feature Type `ps-f:ProtectedSite`. Die erste XPath-Evaluierung findet in Zeile 5 statt. Alle dem aktuellen Feature Type zugehörigen Feature-Instanzen werden nun der Reihe nach in einer zweiten Schleife (Zeile 7-14) durchlaufen und zu jedem Feature eine Ressourcen-URI generiert (Zeile 8, vgl. Abschnitt 5.1.2 *Identifikationsmanagement*). Nun werden alle GML-Elementpfade, die nicht relativ zu einem weiteren Elementpfad sind, gesondert ausgewählt (Zeile 9) und in einer dritten Schleife (Zeile 11-13) durchiteriert. Bezogen auf die oben genannten drei Beispielspfade fallen Pfad 1 und 2 unter die engere Auswahl, Pfad 3 ist relativ zu Pfad 2 und wird deshalb aussortiert. Die Analyse von Pfad 3 findet abhängig zu der von Pfad 2 in einem n-ten Rekursionsschritt statt.

Die dritte Schleife (Zeilen 11-13) ruft in Zeile 12 die elementare Funktion `EvaluateGMLPath` auf, die als rekursive Funktion in den Zeilen 17-47 deklariert ist. Parametrisiert wird sie mit dem jeweils auszuwertenden GML-Elementpfad (`currentGMLPath`) und der URI der übergeordneten GML- bzw. RDF/OWL-Instanz (`subjectResourceURI`), die jeweils die Subjektrolle in den zu erstellenden ontologischen Aussagen (RDF-Tripel) einnehmen (vgl. Zeilen 29, 33 und 43). In Zeile 12 wird der `EvaluateGMLPath`-Funktion die URI der Feature-Instanz übergeben (`featureURI`; z.B. `inspiredata:DE.BW.PS/335`). Die Funktion beginnt in Zeile 21 mit dem Identifizieren des OWL-Prädikats (`relatedPredicate`), das für die nächste ontologische Aussage eingesetzt werden soll (vgl. ebenso Zeilen 29, 33 und 43). Das gesuchte OWL-Prädikat ist das letzte im Prädikatpfad, aus dem der aktuelle GML-Elementpfad abgeleitet wurde; beispielsweise das Prädikat `ont_ps:designation` im Prädikatpfad `?feature/ont_ps:siteDesignation/?designationContext/ont_ps:designation/?designation`, aus dem der GML-Elementpfad `ps-f:ProtectedSite/ps:siteDesignation/ps:DesignationType/ps:designation` geschlussfolgert wurde (vgl. Abschnitt 5.2.1, Absatz

GML-Elementadressierung). Somit liegen Tripelsubjekt und -prädikat fest und der restliche Funktionsrumpf behandelt das Einlesen und Aufbereiten des Tripelobjektes.

```
1  INIT temporaryRDFTriplestore
2  FOR each featureType
3    SET gmlPathList of all GML-element paths relative to featureType
4    order gmlPathList alphabetically // with ascending path length
5    CALL ParseResults for featureType RETURNING featureIndexList
6
7    FOR each featureIndex in featureIndexList
8      CALL ComputeResourceURI with featureIndex RETURNING featureURI
9      SET gmlPathSublist filters all non-relative GML-element paths in gmlPathList
10
11     FOR each gmlPath in gmlPathSublist
12       CALL EvaluateGMLPath with gmlPath and featureURI
13     ENDFOR
14   ENDFOR
15 ENDFOR
16
17 FUNCTION EvaluateGMLPath is:
18 input: currentGMLPath and subjectResourceURI
19 output: nothing
20
21   CALL GetRelatedPredicate with currentGMLPath RETURNING relatedPredicate
22   CALL ParseResults for currentGMLPath RETURNING entityIndexList
23
24   FOR each entityIndex in entityIndexList
25
26     CASE currentGMLPath OF
27       refers to an INSPIRE featureType:
28           CALL ComputeResourceURI with entityIndex RETURNING resourceURI
29           store within temporaryRDFTriplestore the statement "subjectResourceURI relatedPredicate resourceURI"
30
31       refers to an INSPIRE dataType:
32           CALL ComputeResourceURI with entityIndex RETURNING resourceURI
33           store within temporaryRDFTriplestore the statement "subjectResourceURI relatedPredicate resourceURI"
34
35           SET relativeGMLPathList with all GML-element paths relative to currentGMLPath from gmlPathList
36           FOR each relativeGMLPath in relativeGMLPathList
37             CALL EvaluateGMLPath with relativeGMLPath and resourceURI
38           ENDFOR
39
40       OTHERS: // of simple datatypes: e.g. date-, numeric- or Codelist-values
41           CALL GetDataValue with entityIndex RETURNING dataValue
42           CALL GetSemanticWebDataValue with dataValue RETURNING dataValue
43           store within temporaryRDFTriplestore the statement "subjectResourceURI relatedPredicate dataValue"
44     ENDCASE
45   ENDFOR
46
47 ENDFUNCTION EvaluateGMLPath
```

Codebeispiel 20 – Parsen der WFS GetFeature-Antwort

In Zeile 22 wird die zweite XPath-Evaluierung angestoßen und die XML-Entities (XML-Elemente bzw. -Attribute) aufgesucht, die dem GML-Elementpfad `currentGMLPath` zugehören. Die gefundenen XML-Entities werden durchiteriert (Zeile 24) und nach INSPIRE-Typenkategorien bzw. -Stereotypen getrennt weiterbehandelt (Zeile 26-44). Für INSPIRE *featureType*-Instanzen (Zeile 27) ist eine Ressourcen-URI zu bilden (Zeile 28), die als Objekt (z.B.

inspiredata:DE.BW.RA/201) eines neuen RDF-Tripels Verwendung findet, wie z.B. inspiredata:DE.BW.PS/335 ont_ps:isManagedBy inspiredata:DE.BW.RA/201. Mit der Speicherung der Verknüpfungsinformation als RDF-Tripel im temporären RDF-Triplestore (Zeile 29) endet hiermit die Verarbeitung von abhängigen INSPIRE Feature-Instanzen, die in einem erneuten Schleifendurchlauf (Zeile 2) gesondert berücksichtigt werden. Im Gegensatz dazu erfolgt bei der Auswertung von INSPIRE dataType-Instanzen zusätzlich der rekursive Aufruf der Funktion EvaluateGMLPath (Zeile 37), und zwar mehrfach je nachdem, wieviele relativen GML-Elementpfade jenem der Variablen currentGMLPath folgen (Zeile 35). In den obigen drei Musterbeispielen ist der dritte vom zweiten Elementpfad abhängig und löst einen weiteren Rekursionsschritt in Zeile 37 aus. Diejenigen XML-Entities, die in eine andere INSPIRE-Typenkategorie fallen (z.B. codeList) oder simplen Datentypen entstammen (z.B. DateTime oder CharacterString), werden ausgelesen (Zeile 41) und für den Gebrauch im Semantic Web-Kontext aufbereitet (Zeile 42, vgl. Abschnitt 5.1 *Ontologie-Modellierung*). Letzten Endes erfolgt ihre Speicherung in einem RDF-Statement (Zeile 43), wie z.B. inspiredata:DE.BW.PS/335 ont_ps:beginLifespanVersion "2005-08-15T00:00:00Z"^^xs:dateTime.

Die im Codebeispiel 20 abstrahierte Programmlogik verzichtet auf die Darstellung mancher relativer XPath-Auswertungen, die vonnöten sind, um u.a. neben numerischen Werten die charakteristische Maßeinheit abzuspeichern oder textuelle Informationen - sofern vorhanden - mit der zugehörigen Sprachangabe auszulesen. Die diesbezüglichen Abbildungsregeln sind in den cf:langXPath-, cf:uomXPath- und weiteren Annotationen der Themenontologien enthalten (siehe Abschnitt 5.1.5).

Ergebnisannotation/-verlinkung mit externen Quellen

Die WFS-Ergebnisinhalte liegen nach der Formatkonvertierung in RDF/OWL im temporären Repository vor, einer RDF-Datenbank mit SPARQL-Anfrageschnittstelle. An die Schnittstelle wird in einem letzten Auswerteschritt die ursprüngliche SPARQL-Anfrage gestellt (siehe Workflow in Abbildung 5.9). In die finale SPARQL-Auswertung können zusätzliche Informationsquellen einfliessen, um die INSPIRE-Inhalte anzureichern und ihre andernfalls isolierte Betrachtung auf weitere ontologische Aussagen, Einschränkungen und Verknüpfungen auszuweiten. Die Motivation dazu ist vielgestaltig, beschränkt sich im Wesentlichen aber auf zwei Kernaspekte. Einerseits einen Informationsmehrwert für INSPIRE-Inhalte zu schaffen, der gemäß der Linked Data-Maxime über die Vernetzung zu anderen Semantic Web-Datentöpfen zu erzielen ist. Andererseits helfen Konzeptbezüge die Semantik von INSPIRE-Inhalten zu festigen und zu interpretieren, was nicht zuletzt dem grundlegenden Charakter des Semantic Web entgegenkommt.

Für die genannten Zwecke kommen unterschiedliche Informationsquellen in Betracht. Die Abbildung 5.13 überblickt vier verschiedene Quellen inklusive den WFS-Ergebnissen (Kategorie C), die nun im Einzelnen kurz beleuchtet werden:

- **INSPIRE Themenontologien (A)**: die Themenontologien verfügen über das schematische Grundgerüst der Annex-Themen. Sie definieren die INSPIRE-Konzepte und ontologische Einschränkungen, beispielsweise ont_ps:ProtectedSite rdfs:subClassOf geo:Feature
- **Verlinkungen im Zuge des Ontologie-Alignment (B)**: die Konzepte der Themenontologien können mit externen Konzepten verknüpft werden, um ein semantisches Reasoning zu fördern und die semantische Interoperabilität zu fremden Konzeptwelten herzustellen, z.B. ont_ps:ProtectedSite rdfs:subClassOf upperOntology:RestrictionZone
- **Virtuelles INSPIRE-Repository (C)**: der OWS-Proxy integriert virtuell alle INSPIRE-Inhalte der angebundenen Downloaddienste und stellt sie bezogen auf konkrete SPARQL-Anfragen bereit, z.B. inspiredata:DE.RP.PS/101 rdf:type ont_ps:ProtectedSite und inspiredata:DE.RP.PS/101 ont_ps:siteName SSchmidtenhöhe"

- **Verlinkungen mit Linked Open Data und Benutzer-Annotationen (D)**: sie erweitern das Datenangebot der INSPIRE-Downloaddienste mit Informationen aus dritter Hand, d.h. aus bereits verfügbaren Linked Data-Datenquellen oder zusätzlich erfassten benutzergenerierten Inhalten von interessierten Privatpersonen (Stichwort: *Crowd Sourcing*). Verlinkt werden die in ihrer Syntax harmonisierten und beständigen INSPIRE Ressourcen-URIs, z.B. `inspiredata:DE.RP.PS/101 owl:sameAs externalLinkedData:schmidtenhöhe`, `inspiredata:DE.RP.PS/101 rdfs:seeAlso <http://rlp.nabu.de/projekte/lfa-weidelandschaft/schmittenhoehe-koblenz/>` oder `inspiredata:DE.RP.PS/101 dc:description "die Besichtigung des Beweidungsprojektes Schmidtenhöhe ..."`.

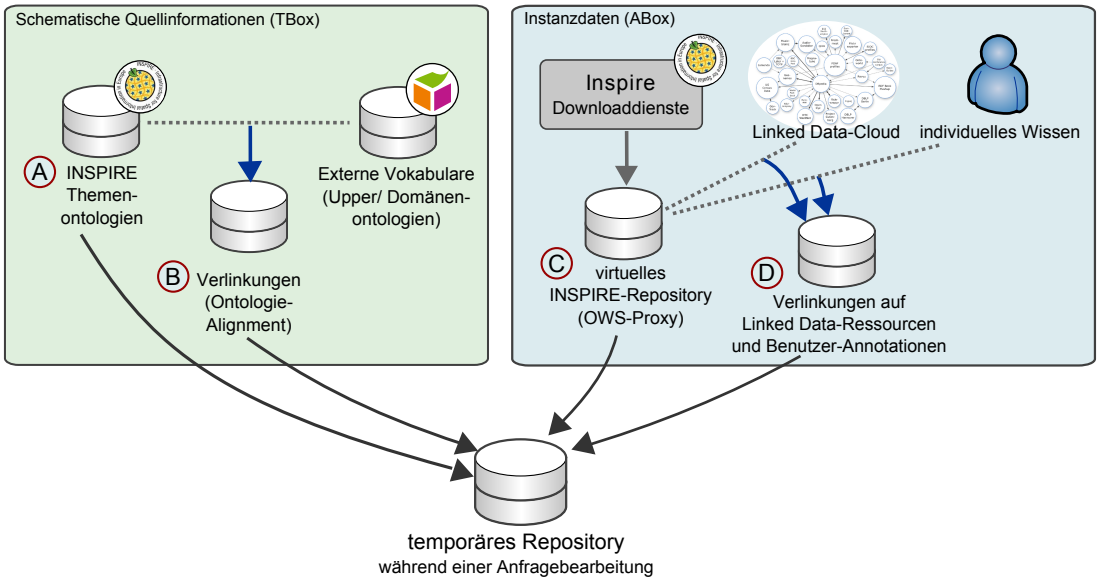


Abbildung 5.13 – Datenquellen zur weiterführenden Annotierung von Anfrageergebnissen

Die vorgestellten Quellkategorien sind über explizit zugewiesene **Graphen-URIs** zu unterscheiden, die als Zusatzinformation an jede ontologische Aussage angehängt werden (vierte Information in einem Quad, siehe Abschnitt 2.1 *Begriffsdefinitionen*). Ein Beispiel einer Graphen-URI liefert die Schutzgebiete-Themenontologie (Kategorie A): `http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/`. Ein weiteres Beispiel ist die Graphen-URI des Linked Data-Projekts DBPedia (Kategorie D): `http://dbpedia.org/ontology/`. Die Graphen-URI eignet sich gewissermaßen als Quellangabe (*Graphenkontext*), über die in einer SPARQL-Anfrage z.B. zwischen benutzergenerierten Inhalten und solchen behördlicher Erfassung oder zwischen der Konzeptwelt (Kategorie A und B) und den Instanzdaten (Kategorie C und D) differenziert werden kann.

Für die **Wissensvernetzung von INSPIRE-Ressourcen** sind die Verlinkungen der Kategorien B und D angedacht, die statisch vorzuliegen haben und bei jeder Anfragebearbeitung je nach Relevanz und Bezug zu den WFS-Ergebnissen (Teilmenge von Quelle C) in das temporäre Repository eingeladen werden. Die semantischen Verlinkungen sind manuell, teilautomatisiert mit manueller Nachbearbeitung/ Sichtkontrolle oder vollständig über automatische Ähnlichkeitsanalysen bereitzustellen. Abgesehen von aussichtsreichen schematischen Verknüpfungen mit geeigneten Upper- und Domänenontologien (siehe Abschnitt 5.1.6) bieten sich hinsichtlich der Instanzverlinkung sogar einige europäische Linked Data-Datentöpfe an, die einen engen Bezug zu INSPIRE aufweisen. Angefangen von der *Reporting Obligations Database*[119], in der europäische Richtlinien registriert stehen, über *EuroStat*-Codes[120] bis hin zu Verschlagwortungen des europäischen *GEMET*-Thesaurus[121] reicht das Informationsangebot. Im Bereich der populären Semantic Web-

[119] gepflegt bei der European Environmental Agency (EEA); siehe: http://rod.eionet.europa.eu/obligations.rdf
[120] siehe: http://ec.europa.eu/eurostat/ramon/rdfdata/void.rdf
[121] siehe: http://www.eionet.europa.eu/gemet/void.rdf

Projekte sind vor allem DBPedia, GeoNames und LinkedGeoData in Betracht zu ziehen. Je nach INSPIRE-Thematik sind domänenspezifische Semantic Web-Projekte zu identifizieren und die Möglichkeiten und Verlinkungspotentiale individuell zu erörtern.

5.2.5 Geographische Filterfunktionen

Die räumliche Dimension ist ein hervorragendes Vergleichskriterium für Objekte der Realwelt, die dank ihres Raumbezugs unabhängig von der speziellen Objektklasse räumlich miteinander in Beziehung gesetzt werden können. Das Grundlagenkapitel weist in Abschnitt 2.3.3 *Topologische Beziehungen* auf einen markanten Unterschied hin: zwischen der **statischen Speicherung** von Lagebeziehungen einerseits und der **Berechnung zur Laufzeit** andererseits. Beide Praktiken können zwar auf nicht-topologische geometrische Sachverhalte übertragen werden, wie z.B. Schnittmengen- oder Buffergeometrien von Objekten, die ebenso entweder statisch oder erst dynamisch zur Laufzeit berechnet vorliegen. Geometrische Sachverhalte eignen sich aber nur bedingt zum Filtern und dienen meist als Hilfestellung anschließender topologischer Überprüfungen. Deshalb sollte das Hauptaugenmerk eher den topologischen Beziehungen gelten.

Das Semantic Web förderte bislang primär Vokabulare und Datenbestände zu Tage, die in die erste Kategorie fallen. So wurden verschiedenste **Geo-Vokabulare für statische Speicherzwecke** definiert (siehe Abschnitt 2.4.1). Die topologischen Prädikate, wie z.B. `gn:nearby` oder `gn:parentFeature` aus dem Geonames-Vokabular, sind mit herkömmlichen SPARQL-Auswertungen zu interpretieren, ohne dabei auf GeoSPARQL-Funktionalitäten zurückgreifen zu müssen. Die Anfrageschnittstellen (SPARQL-Endpoints) sind gegebenenfalls mit Mechanismen des Query-Rewriting ausgestattet, um die SPARQL-Anfragen in Bezug auf geographische Fragestellungen zu optimieren (vgl. hierzu Abschnitt 2.4.2). Im Semantic Web-Umfeld üblich ist demnach das erste in Aufstellung 5.14 wiedergegebene Auswerteverfahren, das SPARQL-Graphenmuster (BGP_{geo}) einsetzt und über RDF/OWL-Statements mit topologischen Prädikaten (`SPO + TP`) filtert, um Lagebeziehungen festzustellen und davon abhängige Ergebnisse (`R`) auszugeben.

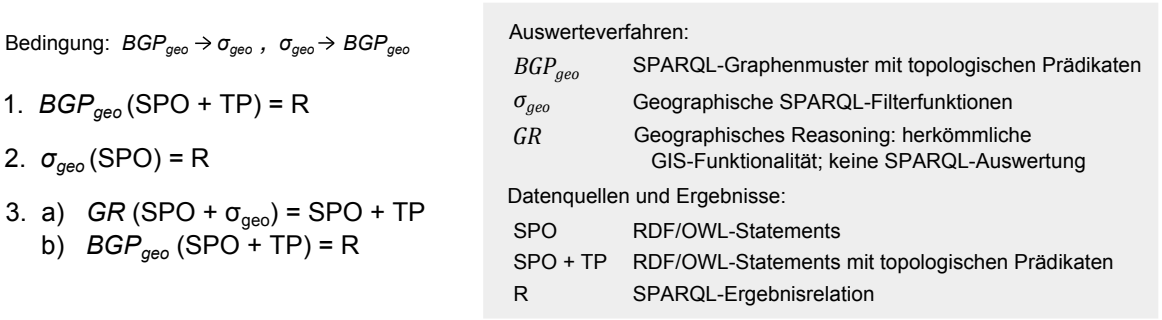


Abbildung 5.14 – GIS-Auswertestrategien im Zuge einer SPARQL-Anfragebearbeitung

Die Spezifikation **GeoSPARQL** erweitert die GIS-Auswertemöglichkeiten des Semantic Web, indem per GeoSPARQL-Filterfunktionen auch Berechnungen zur Laufzeit unterstützt werden (σ_{geo}). Zwar müssen nach wie vor Objektgeometrien in der Wissensbasis gespeichert sein, jedoch kommt man ohne zusätzliche topologische Prädikate aus ($\rightarrow$ `SPO`). Das ist sehr vorteilhaft, bedenkt man die vielen räumlichen Vergleichsmöglichkeiten von Objekten, die insbesondere durch das Bilden von Kreuzprodukten einen erheblichen Speicherauwand nach sich ziehen würde und nur für ausgewählte Objektbeziehungen vertretbar sind. GeoSPARQL ermöglicht somit auch das zweite Auswerteverfahren in Aufstellung 5.14. Darüber hinaus bietet GeoSPARQL Regeln für ein Query Rewriting zur Transformation

von Graphenmustern in GeoSPARQL-Filterfunktionen an ($BGP_{geo} \rightarrow \sigma_{geo}$; siehe die exemplarische Transformationsregel in Codebeispiel 10). Der umgekehrte Fall ($\sigma_{geo} \rightarrow BGP_{geo}$) ist ebenso praktikabel. Es kann also angenommen werden, dass wenn eine SPARQL-Anfrage zugunsten einer der beiden Auswerteformen (ob BGP_{geo} oder σ_{geo}) formuliert ist, die jeweils andere Auswerteform daraus hergeleitet werden kann. Es folgt eine kritische Betrachtung der geographischen Filterfunktionalität von GeoSPARQL, in deren Rahmen auch auf eine mögliche dritte Auswertestrategie hingewiesen wird.

Analyse der GeoSPARQL-Filterfunktionalität

Als beständige Größe im Bereich räumlicher Filteroperationen haben sich die 8 *Egenhofer-Relationen* erwiesen, deren geringfügige Abwandlungen in den OGC-Anfragesprachen *Simple Feature Access* (Wirkbereich: Geodatenbanken) und OGC *Filter Encoding* (Wirkbereich: Geodatentransfer) enthalten ist. GeoSPARQL unterstützt parallel die **Filter-Operatorensätze** nach a) *Egenhofer*, b) *Simple Feature Access* (SFA) und c) *Region Connection Calculus* (RCC). Die Vielzahl an teils gleichbedeutenden Operatoren (vgl. Abschnitt 2.3.3) spiegelt sich in den GeoSPARQL-Funktionsbezeichnern wieder (z.B. `geo:eh-equals`, `geo:sf-equals`, `geo:rcc8-eq`) und ist angesichts des Anspruches, eine einfache räumliche SPARQL-Erweiterung zu bilden, recht verwirrend. Genügen würde ein Operatorensatz, wie z.B. die SFA-Operationen. Der SFA-Operatorensatz enthält zugleich die wertvolle und in GeoSPARQL gesondert definierte `geo:relate`-Operation, die mit einer DE-9IM Schnittmatrix parametrisiert, alle denkbaren topologischen Beziehungen prüfen kann.

Ein anderes Manko, auf das bereits der Abschnitt 5.1.4 eingeht, ist die fehlende **Rückwärtskompabilität** der GeoSPARQL-Filteroperationen. Diese lassen sich per Definition nur mit geometrischen Literale aus dem GeoSPARQL Geo-Vokabular parametrisieren, nämlich den RDF-Datentypen `sf:WKTLiteral`, `gml:GMLLiteral`. Herkömmliche geometrische Speicherformen des Semantic Web, z.B. die verbreiteten Basic Geo-Prädikate `wgs84_pos:lat` und `wgs84_pos:long` zur Speicherung von Punktkoordinaten, finden keine Berücksichtigung. Im Semantic Web jedoch und speziell in der Linked Open Data-Cloud sind zahlreiche Datentöpfe mit eben jenen einfachen Geo Tagging-Attributen verortet. Ohne größeren Aufwand ließen sie sich in GeoSPARQL-Filterfunktionen verwenden. Dazu müssten nur die Filterfunktionen mit weiteren Parametern für die Übergabe von Punktkoordinaten überladen werden, so dass auch Objektwerte der Prädikate `wgs84_pos:lat` und `wgs84_pos:long` in Frage kommen. Beispielsweise lässt sich die Operation der Signatur `geo:sf-equals( geo:GeomLiteral, geo:GeomLiteral )` mit drei oder vier Parametern überladen, damit anstelle von kompakt kodierten Geometrien `geo:GeomLiteral` auch Punktgeometrien bestehend aus je zwei numerischen Variablen zulässig sind: z.B. `geo:sf-equals( geo:GeomLiteral, xsd:decimal, xsd:decimal)`.

Die zur Laufzeit stattfindende Analyse von topologischen Beziehungen verlangsamt die SPARQL-Auswertung, die ihre Stärken üblicherweise im Bereich der Graphenvergleiche hat. Diesbezüglich scheint die GeoSPARQL-Spezifikation auf SPARQL-Endpoints ausgelegt zu sein, die nicht nur auf RDF-Triplestores, sondern auch gleichzeitig auf Geodatenbanken aufsetzen und operieren. Geodatenbanken erlauben dank räumlicher Indizes schnelle Geometriezugriffe und -filterungen. Durch die Aufteilung der Daten in verschiedenartige Speichertöpfe können gewöhnliche Graphenmustervergleiche auf RDF-Triplestores und räumliche Auswertungen separiert auf Geodatenbanken ablaufen, wodurch sich eine akzeptable Performanz erzielen lässt. Wie sieht es aber in anderen **Systemkonstellationen** aus, deren Systemkomponenten vergleichbar zum OWS-Proxy eine verteilte Suche (Federation) über diverse externe Datenquellen starten und die Ergebnisse gegebenenfalls geographisch auswerten und verschneiden müssen? In einem solchen Szenario ist zum Zeitpunkt der Auswertung keine Geodatenbank verfügbar bzw. müsste temporär gefüllt werden.

Sowohl im Falle eines autarken SPARQL-Endpoints als auch für Anfrageverteiler (Federatoren) kann das gleiche Auswerteverfahren (siehe 3. in Aufstellung 5.14) dazu dienen, einen SPARQL-Prozessor zu entlasten. Angenommen sei der Regelfall, dass die Wissensbasis keine Aussagen mit topologischen Prädikaten enthält: also `SPO`. Demzufolge

müsste der SPARQL-Prozessor GeoSPARQL-Filterfunktionen (Verfahren 2) anwenden und dabei viele einzelne GIS-Operationen durchführen, um Lagebeziehungen dynamisch zu detektieren. Die Auswertung nach Verfahren 1 ist allein über Graphenvergleiche (BGP_{geo}) möglich, bedingt aber vorhandene Aussagen mit topologischen Prädikaten (`SPO + TP`). Es ist deshalb eine Aufteilung in zwei Schritte zu befürworten. Erstens eine **geometrische Vorprozessierung**, die basierend auf den geographischen Filteroperatoren (σ_{geo}) gültige topologische Aussagen generiert (siehe 3a). Zweitens eine abschließende SPARQL-Auswertung über die der Wissensbasis gezielt hinzugefügten topologischen Aussagen (siehe 3b).

Die geometrische Vorprozessierung kann sich bei dieser Vorgehensweise ausschließlich um die Geodatenverarbeitung, sprich topologische Filterungen, räumliche Indizierung, Koordinatentransformationen etc. kümmern und den SPARQL-Prozessor darin entlasten. Die geometrische Vorprozessierung muss nicht unbedingt auf das Anwendungsfeld statischer Geodatenbanken beschränkt sein. Denkbar im Anwendungsszenario einer verteilten Suche ist auch das Delegieren der Aufgaben an entfernte Schnittstellen, die ein *geographisches Reasoning* mit herkömmlicher GIS-Funktionalität ausüben. Im OGC-Kontext ist insbesondere der OGC Web Processing Service (WPS) für diese Art von externalisierten GIS-Berechnungen vorgesehen (siehe Abschnitt 2.3.4) und könnte, angepasst mit entsprechenden Ein- und Ausgabeformaten (vgl. 3a), in der Ergebnisaufbereitung einer verteilten Suche zum Einsatz kommen. Damit schließt an dieser Stelle die Betrachtung über geographische Filterfunktionen. Auf die konkrete Realisierung im Rahmen der prototypischen Implementierung geht der Abschnitt 5.3.3 ein.

5.3 Implementierung eines Prototypen

Die Anwendbarkeit der entwickelten Konzepte wurde im Rahmen der Arbeit an einem Prototypen erprobt. Da die vorgeschlagene Infrastruktur das Vorhandensein lauffähiger INSPIRE-Quelldienste voraussetzt, diese aber zum Zeitpunkt der Implementierung und der Testphase noch nicht verfügbar waren[122], musste eine INSPIRE-Testplattform bestehend aus einem INSPIRE-Downloaddienst und Testdatenbeständen eigenhändig aufgebaut werden. Hierauf geht der erste Unterabschnitt 5.3.1 ein. Auf der INSPIRE-Datenbasis setzen die weiteren Infrastrukturkomponenten auf, bei denen es sich erstens um die praktische Umsetzung beispielhafter Themenontologien (Abschnitt 5.3.2) und zweitens um eine Testrealisierung des Proxy-Konzeptes handelt (Abschnitt 5.3.3). Schließlich wird in Abschnitt 5.3.4 das Zusammenspiel der technischen Infrastruktur unter die Lupe genommen und die Praxistauglichkeit der Konzepte anhand verschiedener Testfälle untersucht.

5.3.1 Einrichtung einer INSPIRE-Testplattform

Der folgende Abschnitt beschreibt zuerst die für den INSPIRE-Downloaddienst gewählte Software-Implementierung, die damit einhergende Strategie zur Befüllung mit INSPIRE-konformen Datensätzen und andere bedeutende Systemmerkmale. Der zweite Absatz beschäftigt sich sodann mit den Datensätzen, die zu Testzwecken eingesetzt wurden, und welche davon provisorisch neuerstellt bzw. einer vorherigen Aufbereitung bedurften, um die Konformität mit den INSPIRE-Datenspezifikationen zu erlangen.

Betrieb eines INSPIRE-Downloaddienstes

Während der Implementierungsphase bot sich mit dem *Deegree-Framework*[123] der Firma *latlon* die einzigste Open-Source Implementierung eines INSPIRE-Downloaddienstes. Gemeint ist im Speziellen die Ausführung **Deegree3 inspireNode**[124], die keinesfalls eine Notlösung zum Vermeiden kommerzieller Produkte darstellt, wie z.B. die in Abschnitt 2.5.4 vorgestellte Software der Firma Conterra. Ganz im Gegenteil machte der gelungene Entwurf des Deegree3 die Wahl recht leicht, da er einige konzeptionelle Stärken aufweist, die speziell die Datenspeicherung und -anbindung und das stabile Antwortzeitverhalten auf Datenanfragen betreffen. Der Deegree3 inspireNode ist eine Java-Webapplikation[125], die unter einem Java-Applikationsserver, wie z.B. dem *Apache Tomcat*[126], lauffähig ist. Der Apache Tomcat ist der am häufigsten eingesetzte Java-Applikations- und Webserver. Er wurde aufgrund seines stabiles Laufzeitverhaltens für die INSPIRE-Testplattform verwendet, um die Webapplikation Deegree3 inspireNode auszuführen.

Das Systemsetup Deegree3 inspireNode beinhaltet sowohl einen INSPIRE-Downloaddienst (WFS 1.1.0) als auch einen INSPIRE-Kartendienst (*View Service*; WMS 1.3.0). Beide Dienste greifen auf die gleiche Datenbasis zu, den sogenannten *INSPIRE Feature Store*, siehe Abbildung 5.15. Der INSPIRE Feature Store ist hilfreicherweise vorkonfiguriert mit den Datenmodellen der INSPIRE Annex-I-Themen und lässt sich mit verschiedenen relationalen Datenbanksystemen betreiben, z.B. wahlweise PostgreSQL[127] oder Oracle[128]. Für die Testumgebung wurde die Open-Source-Datenbank PostgreSQL mit ihrer räumlichen Erweiterung PostGIS[129] verwendet. Die Befüllung des Feature Stores ist wohldurchdacht. Alles was der Administrator für einen Importvorgang benötigt sind INSPIRE-konforme GML-Datensätze. Diese lassen sich entweder per WFS Transaction-Operation oder bequem über eine Web-GUI einladen. Der Administrator wird hierdurch in mehrfacher Hinsicht entlastet, denn er muss weder die Datensätze über

[122] die initiale Bereitstellung (*initial operational capability*) ist geplant für Mitte 2012, die vollständige Einsatzfähigkeit für Ende 2012. Jedoch betrifft dies inhaltlich nur die Annex-I-Themen; siehe INSPIRE-Roadmap: http://inspire.jrc.ec.europa.eu/index.cfm/pageid/44
[123] Projektseite: http://www.deegree.org/
[124] Dokumentation: http://wiki.deegree.org/deegreeWiki/InspireNode?action=show&redirect=deegree3%2FInspireNode
[125] Java ist eine der führenden, höheren Programmiersprachen und erfreut sich inbesondere in der Webprogrammierung großer Beliebtheit; siehe Dokumentation zu Java: http://www.oracle.com/technetwork/java
[126] Projektseite: http://tomcat.apache.org/
[127] Projektseite: http://www.postgresql.org/
[128] Produktseite: http://www.oracle.com/us/products/database/index.html
[129] Projektseite: http://postgis.refractions.net/

externe Importroutinen in die Datenbank überführen noch hat er sich um die Spezifika der Diensteanbindung zu kümmern (Stichwort: *Objekt-relationales Mapping*, ORM). Die Importschnittstelle macht sich das standardisierte Datenformat - INSPIRE-konformes GML - zu nutze und entkoppelt damit den Prozess der anfänglichen INSPIRE-Datenaufbereitung von der anschließenden Publizierung über INSPIRE-Dienste. Aus diesem Grund lassen sich auch die hier eingesetzten Testdatensätze separat von den Diensten betrachten (siehe nächster Absatz).

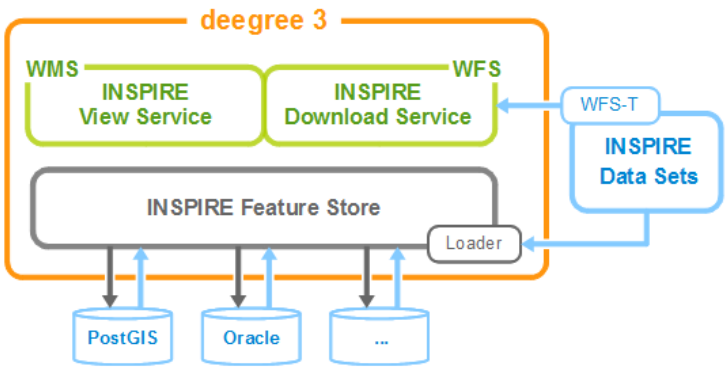


Abbildung 5.15 – Architekturskizze des Deegree3 inspireNode[130]

Die GML-Instanzdaten liegen im Feature Store im sogenannten **BLOB-Modus** vor, d.h. sie werden in eine Tabellenspalte des Datentyps *Binary Large Object* (BLOB) abgelegt und nicht etwa relational-normalisiert über mehrere Tabellen verteilt. Dies hat zur Folge, dass sich zusammenhängende GML-Objekte sehr performant an anfragende WFS-Klienten übertragen lassen. Darüber hinaus ist der inspireNode-Downloaddienst mit **Streaming**-Funktionalitäten ausgestattet, um Datenexporte sukzessive zusammenstellen und ausliefern zu können. Dadurch zeigt der Dienst selbst bei größeren Datenmengen (GML-Dateigröße >= 20 MB) eine stabile Leistungskurve, was nicht zuletzt angesichts der komplexen INSPIRE-Datenmodelle und umfangreichen -Datenmengen von Vorteil ist. Eine komfortable Administrationsoberfläche rundet das Gesamtpacket des inspireNode ab und beinhaltet neben Testoberflächen für Diensteanfragen und einem integrierten Kartenbetrachtungswerkzeug auch diverse Konfigurationsdialoge beispielsweise für Dienstemetadaten, Datenbankverbindungsparameter oder Symbolisierungsvorschriften. Der Deegree3 inspireNode macht insgesamt also einen sehr guten Eindruck, indem er eine unkomplizierte und gleichzeitig effiziente INSPIRE-Datenbereitstellung gestattet und sich demzufolge hervorragend im Kontext der prototypischen Implementierung eignet.

Datenbeschaffung und Aufbereitung

Die Suche nach INSPIRE-konformen Datensätzen gestaltete sich überaus schwierig. Zwar brachte die INSPIRE-Testphase der Annex-I-Datenspezifikationen, die von Oktober 2008 bis Februar 2009 andauerte, einige aus nationalen Datentöpfen abgeleitete INSPIRE-Datensätze hervor. Die bis dato noch sehr variablen Datenspezifikationen lagen aber erst in Version 1.0 (Stand Oktober 2008) und Version 2.0 vor (Stand Dezember 2008), während die aktuellen Datenspezifikationen, die als stabil anzusehen sind und in die der Erfahrungsschatz der Testphase einfließen konnte, mittlerweile bei Version 3.x angekommen sind (Stand Mai 2010). Die in der INSPIRE-Testphase erstellten Datensätze verblieben nach Konformitätstests überwiegend in der Version 2.0 und wurden nicht in die aktuelle Form der Version 3.x überführt. Darüber hinaus ist das heute bereits verfügbare INSPIRE-Datenangebot immernoch recht spärlich, obgleich die INSPIRE-Implementierungspläne die Publikation von Datensätzen der Annex-I-Themen ab Dezember 2012 vorschreiben.

[130] Quelle: Deegree-Dokumentation unter:
http://wiki.deegree.org/deegreeWiki/InspireNode?action=show&redirect=deegree3%2FInspireNode

Für das in der Masterarbeit primär behandelte INSPIRE-Thema *Schutzgebiete* (*Protected Sites*) konnte lediglich ein einziger umfassender und fertiger Datensatz ausfindig gemacht werden, der nach Anfrage und Vermittlung durch das Joint Research Center (JRC) von der Slovakischen Umweltagentur (SAZP[131]) bereitgestellt wurde (siehe Testdatensatz I in Tabelle 5.7. Der bereitgestellte Datensatz befand sich in der Version 2.0, so dass eine nachträgliche Konvertierung zur aktuellen Version 3.1.0 zu leisten war und manuell in einem XML-Editor durchgeführt wurde. Neben der Beseitigung einfacher Mängel, wie z.B. dem Fehlen eindeutiger GML-Attribute (`gml:id`) für Features und Geometrien oder einer zu korrigierenden XML-Elementreihenfolge, mussten auch neue Inhaltsvorgaben beachtet werden. So wurde aus dem einfachen Klassifizierungselement `ps:designationType` ein komplexes INSPIRE-Element namens `ps:siteDesignation`, das zusätzlich zur Klassifzierung (z.B. *siteOfCommunityImportance*) das übergeordnete Klassifizierungsschema (z.B. *Natura2000DesignationValue*) benennt. Neu in der Version 3.1.0 hinzugekommen ist das INSPIRE-Element namens `ps:siteProtectionClassification`, über das eine sehr pauschale Aussage zur Schutzgebietswidmung getroffen wird (siehe auch das UML-Modell in Abbildung 2.9). Die Elemente zum Klassifizierungsschema und zur Schutzgebietswidmung wurden - sofern möglich - aus vorhandenen Werten geschlussfolgert oder andernfalls mit Testwerten belegt und damit letztlich die INSPIRE-Konformität zur Version 3.1.0 hergestellt.

Tabelle 5.7 – Übersicht der verwendeten INSPIRE-Testdatensätze

Nr.	INSPIRE Annex-Thema	Beschreibung	Datenherkunft	Datenaufbereitung erforderlich?
I	Schutzgebiete	Internationale und nationale slovakische Schutzgebiete (419 Features, GML-Größe: 11,4MB)	Slovakische Umwelt-agentur (SAZP)	ja
II	Schutzgebiete	Testdatensatz zu ausgewählten Schutzgebieten Baden-Württembergs (14 Features, GML-Größe: 1,7MB)	Landesanstalt für Umwelt, Messungen und Natur-schutz BW (LUBW)	ja
III	Administrative Einheiten	Administrative Einheiten der Niederlande (443 Features, GML-Größe: 23MB)	Niederländisches Kataster	nein
IV	Biogeographische Regionen	Testdatensatz zu ausgewählten europäischen Regionen (7 Features, GML-Größe: 929KB)	Europäische Umwelt-agentur (EEA)	ja

Zeitgleich zur Kommunikation mit dem JRC fand seit Ende 2010 eine reger Informationsaustausch mit dem *Fachnetzwerk Schutzgebiete* der GDI-DE[132] statt, um auch an repräsentative deutsche Schutzgebietedaten zu gelangen. Im Fachnetzwerk Schutzgebiete waren bis zum Frühjar 2011 nur die Bundesländer Nordrhein-Westfalen[133] und Baden-Württemberg[134] in der Lage, INSPIRE-konforme Schutzgebietedaten (Version 3.1.0) zu generieren und zu Testzwecken geeignete geringe Datenmengen (jeweils ca. 20 Features) zur Verfügung zu stellen. Beide GML-Datensätze bzw. deren Geometrien befanden sich im geographischen Datum *Deutsches Hauptdreiecksnetz* (DHDN) mit metrischen Projektionen, einerseits Gauß-Krüger Streifen 2 (NRW) bzw. Streifen 3 (BW). Mit Blick auf den erwünschten Einsatz der Proxy-Anwendung und der damit einhergenden Anfragen und Umwandlungen von Geometrien im ETRS89- bzw. WGS84-Datum (siehe Konzeptabschnitt 5.1.4), erweist sich das DHDN-Datum als problematisch, denn es erzwingt

[131] Webauftritt: http://www.sazp.sk
[132] WIKI des Fachnetzwerkes: https://wiki.gdi-de.org/display/fnsg
[133] Dienststelle: Landesamt für Natur, Umwelt und Verbraucherschutz NRW (LANUV), Webauftritt: http://www.lanuv.nrw.de/
[134] Dienststelle: Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg (LUBW); Webauftritt: http://www.lubw.baden-wuerttemberg.de/

während des Datenabrufes eine aufwendige Koordinatentransformation im INSPIRE-Downloaddienst (hier Deegree3 inspireNode). Um Transformationen zur Laufzeit zu vermeiden, wurde einer der Datensätze, und zwar jener aus Baden-Württemberg, vorab in das WGS84-Datum mit geographischer Koordinatenangabe transformiert und in dieser Form für die Testläufe des Prototypen verwendet (siehe Testdatensatz II in Tabelle 5.7).

Um Testläufe auch mit anderen INSPIRE-Thematiken durchzuführen, konnte zum einen der INSPIRE-Datensatz des niederländischen Katasters[135] herangezogen werden. Er beinhaltet alle administrativen Einheiten der Niederlande und liegt erfreulicherweise bereits konform zur aktuellen Datenspezifikation *Administrative Einheiten* (*Administrative Units*) Version 3.0.1 vor (Testdatensatz III). Zum anderen sollte ein Datensatz zum Annex-Thema *Biogeographische Regionen* (*Bio-geographical Regions*) dazu dienen, den Umgang mit Objektreferenzierungen zu prüfen, die Objekte unterschiedlicher INSPIRE-Datenthemen miteinander verknüpfen. So kann gemäß der Spezifikation ein Schutzgebiet (`ps-f:ProtectedSite`) eine Region (`bgr:Bio-geographicalRegion`) über die UML-Assoziation `ps-f:isInRegion` referenzieren (siehe UML-Modell in Abbildung 2.9). Die Datenspezifikation Biogeographische Regionen befindet sich bislang im Entwurfsstadium (Version 2.0, Stand Dezember 2011), so dass sich erwartungsgemäß keine fertigen Datensätze auftreiben ließen. Der hier zum Einsatz kommende INSPIRE-Testdatensatz mit biogeographischen Regionen (Testdatensatz IV) wurde basierend auf dem Datensatz *Biogeographische Regionen, Europa 2001*[136] der Europäischen Umweltagentur (EEA) neuerstellt. Da die EEA-Daten nur als ESRI Coverage-Format heruntergeladen werden können, wurde eine Konvertierung in das GML-Format notwendig. Die Konvertierung geschah in zwei Schritten: 1. der Umwandlung des ESRI Coverage- in das Shape-Format unter Verwendung des Desktop-GIS ESRI ArcGIS[137] und 2. der Umwandlung des Shape- in das GML-Format unter Zuhilfenahme des Desktop-GIS *Quantum GIS*[138]. Die wenigen erforderlichen Attributierungen konnten entweder übernommen werden, wie z.B. der Natura2000-Klassifizierungscode, oder wurden schemakonform mit Testwerten belegt. Nachträglich erfolgte die manuelle Verknüpfung von Schutzgebietedaten der Testdatensätze I und II mit den biogeographischen Regionen des Testdatensatzes IV über die genannte UML-Assoziation `ps-f:isInRegion`. Die Testdatensätze wurden über die Web-GUI des Deegree3 inspireNode importiert und komplettierten damit die einsatzbereite INSPIRE-Testplattform. Eine visuelle Darstellung aller Testgeometrien sowie einige exemplarische Features der Testdatensätze sind im Anhang B aufgenommen.

5.3.2 Praktische Modellierung von INSPIRE-Themenontologien

Die Überführung der INSPIRE UML-Modelle in OWL-konforme Themenontologien ist in Übereinstimmung mit den Abbildungsregeln aus dem Konzeptkapitel vorzunehmen (siehe insbesondere Tabellen 5.2 und 5.3). Die praktische Umsetzung sollte nach Möglichkeit programmgesteuert ablaufen, um jene zahlreichen systematischen Abbildungsprozesse zu unterstützen, die aufgrund der reichhaltigen INSPIRE-Datenmodelle zu leisten sind. Im Idealfall bietet eine **automatisierte Lösung** die individuelle Konfiguration der abzubildenden UML-Sprachkonstrukte und günstigerweise auch eine Option auf manuellen Eingriff, um in Ausnahmefällen von der Systematik der Abbildungsregeln abweichen zu können. Eine derartige Ausnahme sind z.B. komplexe ISO-Datentypen, die als INSPIRE-Basistypen in Gebrauch sind und eine Sonderbehandlung erfahren sollten, um sie auf ein nötiges Maß zu komprimieren oder aus den modellierten Ontologien gänzlich auszusparen (vgl. Abschnitt 5.1.6 *Informationsvernetzung, Bezüge zu Basiskonzepten*).

Die Recherche nach vorhandenen Systemlösungen ergab, dass sich zwar etliche Forschungsansätze und eine Handvoll einsatzfähiger Frameworks der Abbildung von XML-Schema (XSD) nach OWL widmen (z.B. Bedini et al. [2008] oder Bohring & Auer [2005]), so dass immerhin die in XSD verfassten INSPIRE GML-Applikationsschemata durch bewährte Programme transformiert werden könnten. Für eine UML/OWL-Abbildung, die hier wegen der UML-Zusatzinformationen angestrebt wird, fällt das Software-Angebot hingegen spärlich aus und beschränkt sich genau

[135] Webauftritt: http://kadaster.nl/
[136] heruntergeladen vom Datenportal der EEA unter: http://www.eea.europa.eu/data-and-maps/data
[137] Produktseite: http://esri.de/products/arcgis
[138] Produktseite: www.qgis.org/

genommen auf zwei mögliche Systemlösungen. Bevor sie näher erläutert werden, ist zunächst ein kurzer Blick auf die konkrete Speicherform der INSPIRE UML-Modelle hilfreich. Die Modelle wurden - bzw. werden derzeit noch für Annex-II/III-Themen - mit dem UML-Modellierungswerkzeug *Enterprise Architect* (EA[139]) der Firma *Sparx Systems* erstellt, das aufgrund seiner funktionellen Vielseitigkeit und geringen Produktkosten im Rahmen von INSPIRE als primäre UML-Software im Einsatz ist. Der EA speichert die Modellinformationen in einem proprietären Format, den *EA-Projektdateien*, und erlaubt zusätzlich den Modellexport im standardisierten Transferformat *XML Metadata Interchange* (XMI[140]).

Die **erste Systemlösung** entspricht dem Ansatz mit der Konvertierungssoftware *ShapeChange*[141] von der Firma *interactive instruments*. ShapeChange nutzt anstelle der genannten Transferformate direkt eine durch den Enterprise Architect bereitgestellte Programmierschnittelle (engl. *Application Programming Interface*: API), um die darüber ausgelesenen UML-Modellinformationen (Klassen, Relationen, Tagged values, OCL-Restriktionen etc.) zu analysieren und in GML-Applikationsschemata umzuwandeln. Die Software wird insbesondere zum Generieren der INSPIRE GML-Schemata verwendet. Dem Beispiel von ShapeChange folgend, könnten in einer eigenen Programmlogik anstelle der GML-Schemata auch OWL-Ontologien generiert werden. Dieser denkbare Lösungsweg wurde in der Masterarbeit aber nicht weiter verfolgt, weil er einer Neuimplementierung auf Basis der Enterprise Architect-API gleichkäme. Da sich die Masterarbeit bereits auf die prototypische Implementierung eines OWS-Proxy konzentrierte, war von einem zusätzlichen Programmieraufwand im Bereich der Ontologiemodellierung abzusehen.

Die **zweite Systemlösung** ist eine Kombination zweier Programme, der Software *FullMoon*[142] und dem *RDFS Schema generator* (RDFS-SG[143]). FullMoon ist im Umfeld des Projektes *Hollow World*[144] beheimatet, das sich mit der UML-Modellierung der ISO 19100-Standardreihe beschäftigt, und ergänzt das Projekt u.a. um eine Konvertierungsroutine, die - vergleichbar zu ShapeChange - eine UML/GML-Abbildung vornimmt. Die Konsolenanwendung RDFS-SG verwendet als Eingabeformat die von FullMoon erzeugten GML-Schemata, die FullMoon-spezifische Namensgebungen für XML-Elemente oder XSD-Elementdeklarationen aufweisen. Davon abhängig erstellt der RDFS-SG Ontologien in der RDFS-Syntax. Einfachste Konfigurationsmittel des RDFS-SG erlauben die Wahl des Ausgabeformats der generierten Ontologien (z.B. RDF/XML, Turtle etc.) oder beinhalten beispielsweise die Option, verschachtelte GML-Elemente wie z.B. GML-Geometrien auszusparen und nicht auf ontologische Konzepte abzubilden. Die Kombination aus FullMoon und RDFS-SG zeigt sich als ein gangbarer Lösungsweg, sie ist jedoch auch mit einigen Nachteilen behaftet. So trägt die durch FullMoon bewirkte Namensgebung dazu bei, dass sich die späteren RDFS-Konzeptnamen mehr als nötig von den INSPIRE UML-Elementnamen unterscheiden. Desweiteren erzeugt der RDFS-SG nur RDFS-Schemata, die implizit zwar OWL-Ontologien darstellen, wodurch aber die Chance verpasst wird, sinnvolle OWL-Restriktionen anzuwenden. Der Lösungsansatz zur Vokabularentwicklung sieht jedoch u.a. qualifizierende OWL-Klassenrestriktionen vor. Darüber hinaus sind die vom RDFS-SG erstellten RDFS-Schemata nicht OWL-DL konform und somit nicht kompatibel für Inferenzbildungen, da die meisten Reasoner OWL-DL voraussetzen (siehe Abschnitt 5.1). Betrachtet man die genannten Defizite, sind letztlich doch entweder Abstriche vom Lösungskonzept zu machen oder eine aufwendige programmtechnische Anpassung beider Softwarebausteine vonnöten.

Die obigen Software-Beschreibungen sind als Anregung zukünftiger Umsetzungen zu verstehen. Was die Entwicklung von musterhaften Vokabularen im Zuge der Masterarbeit anbelangt, wurde den beschriebenen automatischen Ansätzen ein **manueller Lösungsweg** vorgezogen, um *auf sicherem Wege* zu praktischen und maßgeschneiderten Ergebnissen zu gelangen. Dazu wurde der Ontologie-Editor *Protégé* [145] eingesetzt. Protégé ist eine weitverbreitete Open-Source Desktop-Anwendung, die über eine Benutzer-freundliche GUI diverse Perspektiven auf Konzepte und Instanzdaten gestattet. Das Ontologie-Engineering wird durch Protégé enorm erleichtert, indem etliche Benutzerhilfen zur Verfügung

[139] Produktseite: http://www.sparxsystems.com/products/ea/
[140] Standard der Object Management Group (OMG), Spezifikation: http://www.omg.org/spec/XMI/
[141] Produktseite: http://www.interactive-instruments.de/index.php?id=28&L=2
[142] Projektseite: http://projects.arcs.org.au/trac/fullmoon/wiki/FullMoon
[143] Projektseite: http://tiger.dl.ac.uk:8080/rdfsgenerator/
[144] Projektseite: https://www.seegrid.csiro.au/wiki/AppSchemas/HollowWorld
[145] Projektseite: http://protege.stanford.edu/

stehen, z.B. zur URI-Vervollständigung, für die Organisation von lokalen Ontologiedateien, die Manipulation und Restrukturierung von Konzepten etc. Die Version 4.1 ist in weiten Zügen sogar schon OWL2-konform und bietet eine vollständige Unterstützung für OWL1-Sprachkonstrukte der Teilsprache DL, um die Voraussetzungen für entscheidbare Inferenzbildungen mit gängigen Reasonern zu schaffen. In Protégé sind zu diesem Zweck gleich mehrere Reasoner standardmäßig integriert oder lassen sich nachträglich hinzuladen. Mit dem Reasoner *Pellet* wurden beispielsweise nach Abschluss der Vokabularentwicklung die vom Prototypen abgefragten Instanzdaten validiert.[146]. Die Abbildung 5.16 zeigt einen Screenshot des Editors. Im Bild dargestellt ist die Konzeptklasse `ProtectedSite` der INSPIRE-Themenontologie Schutzgebiete mit den zugehörigen Konzeptannotationen (im rechten oberen Fensterbereich) und einigen Klassenrestriktionen (rechts unten).

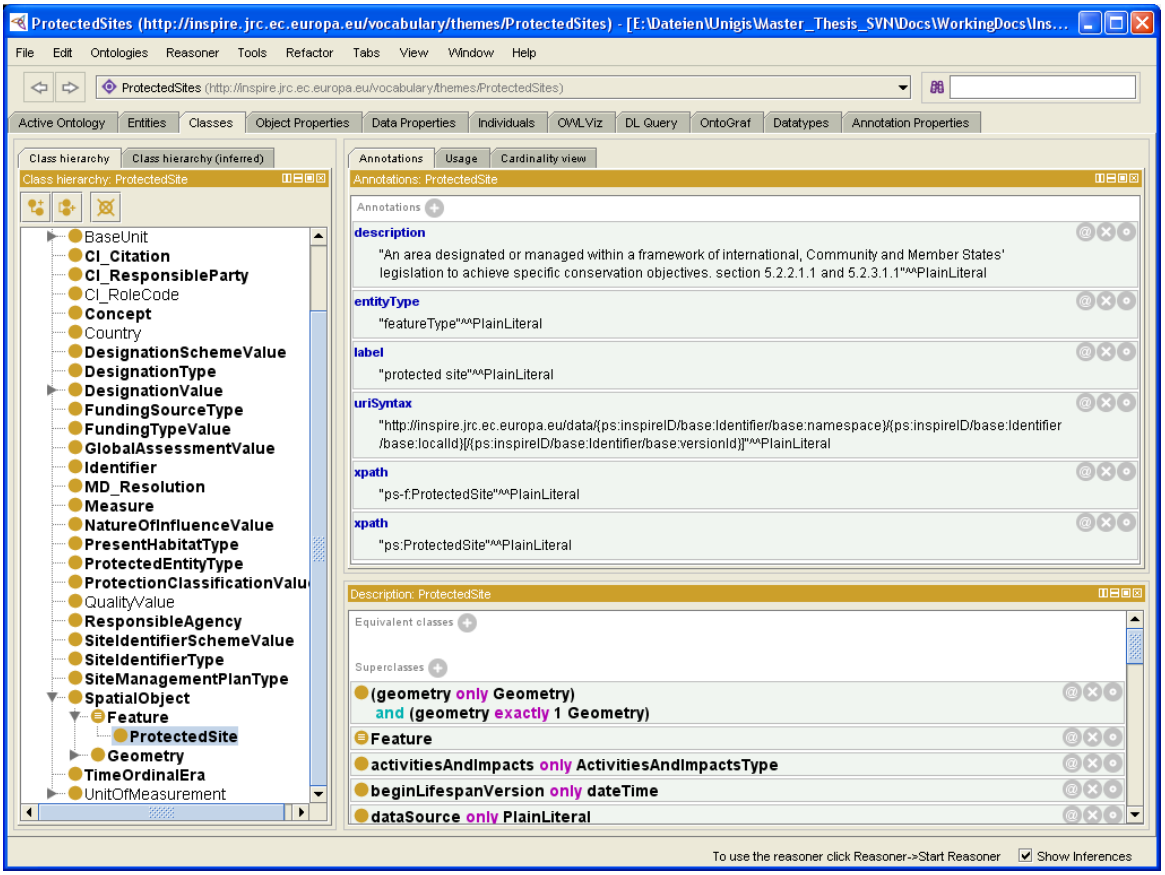


Abbildung 5.16 – Ontologie-Editor Protégé im Praxiseinsatz

Mit Hilfe von Protégé wurden **drei Themenontologien modelliert**, die die Grundlage für die Testfälle in Abschnitt 5.3.4 bilden. Dies sind die Ontologien zu den INSPIRE Annex-Themen a) Schutzgebiete (*Protetected Sites*, Annex-I) b) Verwaltungseinheiten (*Administrative Units*, Annex-I), und c) Biogeographische Regionen (*Biogeographical Regions*, Annex-III). Die Schutzgebiete-Ontologie diente als primäres Experimentierfeld, während mit den beiden restlichen Ontologien die Übertragbarkeit des Modellierungskonzeptes auf weitere INSPIRE-Thematiken nachgewiesen werden sollte und konnte. Desweiteren sind generalisierte Konzepte in eine eigenständige Ontologie *INSPIRE-Basistypen* ausgelagert analog zu Abschnitt 5.1.6. In Ermangelung eines fertigspezifizierten GeoSPARQL Geo-Vokabulars sind dem GeoSPARQL-Entwurfsdokument [Perry & Herring 2011] Konzeptdefinitionen zu geographischen Speicherungen und topologischen Prädikaten entnommen und an die eigenen Verbesserungsvorschläge angepasst worden. Daraus entstand eine separate GeoSPARQL-Ontologie für Testzwecke. Die Themenontologien sind auszugsweise in Anhang C wiedergegeben.

[146] mit der SPARQL-Anfrage in der Query form `Construct` lassen sich (INSPIRE-)Ressourcen im Format RDF/XML ausgeben, die in Protégé eingeladen einer Validierung unterzogen werden können

5.3.3 Entwicklung eines Semantic Web-Proxy

Für die Entwicklung des OWS-Proxy kamen diverse Software-Bausteine und Systembibliotheken zur Anwendung. Die Programmlogik ist aufgeteilt in sechs verschiedene Module, die in der Architekturskizze 5.17 blau eingefärbt sind. Der OWS-Proxy ist - abgesehen vom Server-Betriebssystem (hier eine Linuxdistribution) - integriert in zwei verschiedene Laufzeitumgebungen. Da er in der Funktion eines SPARQL-Endpoints mit SPARQL-Clients über das Internet kommuniziert, ist für den OWS-Proxy der Einsatz eines Webserver erforderlich. Hierfür wird der erwähnte Applikations- und Webserver *Apache Tomcat*[147] verwendet. Auf dem Apache Tomcat ist ein *Sesame HTTP-Server* als Java-Webapplikation eingerichtet, in die der OWS-Proxy eingebettet ist. Der Sesame HTTP-Server ist Teil des **Sesame** Semantic Web-Frameworks[148], das APIs für Semantic Web-Funktionalitäten bereitstellt, um z.B. RDF-Graphen zu parsen und zu speichern. Gegenüber dem ebenfalls auf Java-basierenden und ähnlich populären *Jena*-Framework[149] überzeugte Sesame in mehrfacher Hinsicht. Sesame ist aufgrund seiner weitreichenden und eleganten Modularisierung gut verständlich und performanter als Jena implementiert, was sich u.a. in der schnelleren Umwandlung von SPARQL-Anfragen der Query-Form in SPARQL Algebra-Ausdrücke äußert. Darüber hinaus enthält Sesame mit dem bereits erwähnten Sesame HTTP-Server die Realisierung eines SPARQL-Endpoints als integrativen Bestandteil, während das Jena-Framework für diese Aufgabe ein separiertes, von der Jena-Codebasis differenziertes Unterprojekt namens *Joseki*[150] pflegt.

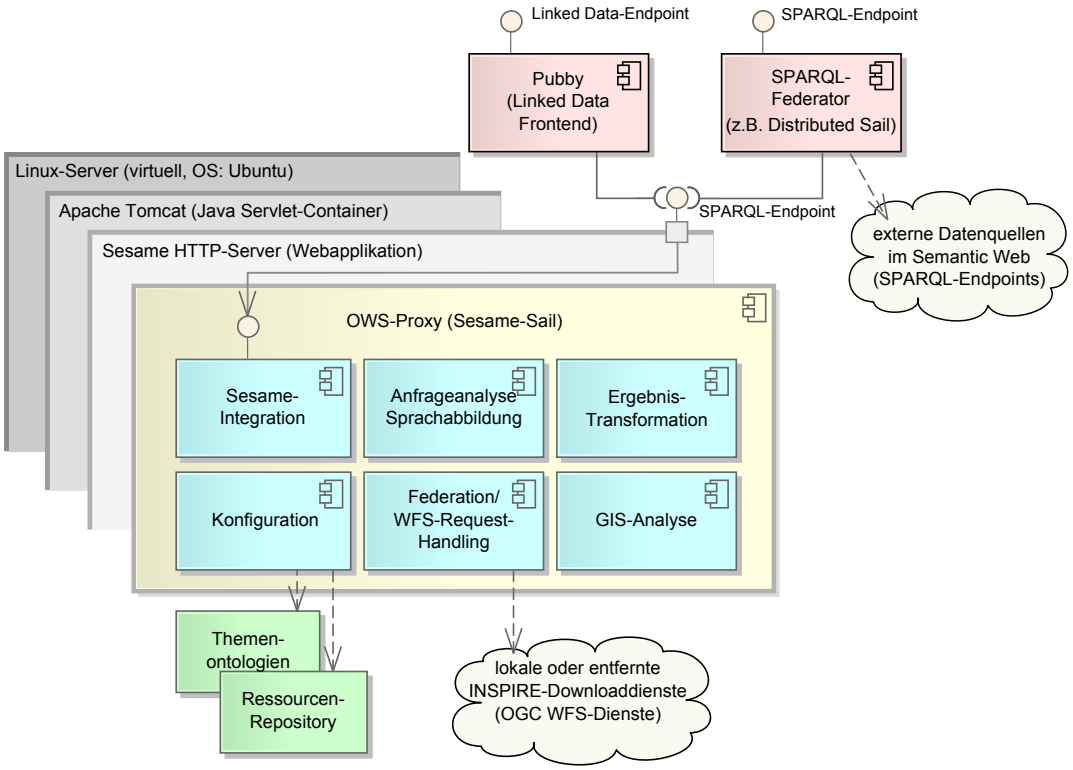


Abbildung 5.17 – Architektur der Proxy-Anwendung, Überblick über die Systemmodule

Die Sesame-Integration wie auch die anderen Proxy-Systemkomponenten (blaue Komponentensymbole in Abbildung 5.17) sind nachfolgend einzeln mit ihrem Software-Einsatz, den Systemlösungen und Bezügen zu den Konzeptinhalten der vorangegangenen Abschnitte 5.2.1 bis 5.2.5 erläutert. In Vorbereitung auf die Testfälle in Abschnitt

[147] verwendete Version: 6.0.33; Projektseite: http://tomcat.apache.org/
[148] verwendete Version 2.3.2; Projektseite: http://www.openrdf.org/
[149] Projektseite: http://jena.sourceforge.net/
[150] Projektseite: http://www.joseki.org/

5.3.4 schließt sich die Beschreibung von zwei weiteren Systembausteinen an. In Abbildung 5.17 rot dargestellt, erweitern sie die SPARQL-Schnittstelle des OWS-Proxy (siehe Interface-Symbol *SPARQL-Endpoint*) um externe Datenzugriffsmechanismen.

Modul: Sesame-Integration

Sesame bietet - allgemein gesprochen - Abfrage- und Speicherfunktionen im Umgang mit RDF-Graphen und damit der prototypischen Entwicklung den primären Zugang zu Semantic Web-Techniken. Die RDF-Strukturelemente sind in Sesame in einem objektorientierten Modell mit Objektklassen und Zugriffsmethoden realisiert. Neben anderen Basisfunktionalitäten, wie z.B. den *RDF-Input/Output*-Methoden (RIO) zum Parsen und Serialisieren von RDF-Formaten (z.B. RDF/XML, Turtle etc.), gibt es zwei bedeutende APIs: 1. die *Repository-API* und 2. die *Storage and Inference Layer-API* (SAIL). Die Repository-API ermöglicht dem Anwender einen gekapselten, logischen Zugriff auf RDF-Graphen, um diese in ein filterbares Repository einzuladen, dort zu manipulieren, terminologische Aspekte zu analysieren und Inferenzen zu bilden. Ein Repository selbst kann in zwei Varianten vorliegen: einem *SAIL-Repository*, das intern Speicherroutinen auf dem gleichen Host-Server nutzt, oder einem *HTTP-Repository*, das den Zugriff auf entfernte Sesame HTTP-Server im Sinne einer ausgelagerten Wissensbasis erlaubt. Für die Masterarbeit relevant ist nur der Typ des SAIL-Repository, das auf einem oder ineinander geschachtelten SAILs aufsetzt. Ein SAIL ist dicht bei der darunterliegenden Persistenzschicht beheimatet (im Akronym namentlich: *Storage*) und führt deshalb sehr speicherschonend die konkreten Anfrageauswertungen und Inferenzbildungen durch (*Inference*) bzw. delegiert diese - sofern SAILs geschachtelt sind - weiter an nachgelagerte SAIL-Instanzen. Abgesehen von den im Framework standardmäßig bereitgestellten SAIL-Klassentypen, die die Ressourcenspeicherung in eine relationale Datenbank (sogenannter *RDBMSStore*), in den Arbeitsspeicher (*MemoryStore*) oder auf Festplatte vornehmen (*NativeStore*), lassen sich mit der SAIL-API auch eigene SAIL-Implementierungen entwickeln. Mit dem Prototypen des OWS-Proxy wurde ein derartiges SAIL realisiert, das die Sesame SAIL-Schnittstelle implementiert und gemäß dem *Java Service Provider Interface* (SPI) in den Sesame HTTP-Server eingebunden ist. Mit dem SPI-Prinzip lässt sich ein übergeordnetes System nur unter der Anwendung von Konfigurationsmitteln um eine individuelle Logik, die die Vorgaben einer Schnittstellenklasse erfüllt, erweitern. Dadurch kann diese Logik mit vergleichbarer Funktionalität beliebig ausgewechselt werden. Der Sesame HTTP-Server bildet in der prototypischen Architektur den SPARQL-Endpoint, der konform zum *SPARQL Protocol for RDF* eine SPARQL-Anfrage entgegennimmt. Diese leitet er an das SAIL, hier an den OWS-Proxy, zur Abfrage der Wissensbasis weiter, welches im Fall des OWS-Proxy einem virtuellen Repository über externe INSPIRE-Downloaddienste gleicht. Nachdem die Resultate zusammengestellt wurden, werden sie an den Sesame HTTP-Server und letztlich an den anfragenden Web-Client ausgeliefert.

Modul: Konfiguration

Beim Anwendungsstart wird der OWS-Proxy mit der Auskunft über angebundene INSPIRE-Quelldienste (*Ressourcen-Repository*, vgl. Abschnitt 5.2.3) und den modellierten Themenontologien initialisiert(*Ontologie-Repository* inklusive INSPIRE-Basistypen, vgl. Abbildung 5.8). Beide Konfigurationsarten liegen in Semantic Web-Datenformaten vor: die Themenontologien sind in einzelnen RDF/XML-Dokumenten serialisiert, während das Ressourcen-Repository gemäß dem Datenmodell aus Abbildung 5.12 in einer einzelnen Datei im Turtle-Format gespeichert ist (Auszüge beider Konfigurationen sind in den Anhängen C und D aufgenommen). Der Vorteil von Semantic Web-Formaten ist ihre unmittelbare Interpretierbarkeit in Semantic Web-Umgebungen. Die Konfigurationen können über die Sesame RIO-Schnittstelle direkt in ein temporäres Repository eingelesen und mittels regulärer SPARQL-Anfragen geparst werden, wodurch das Entwickeln eines maßgeschneiderten Importprozesses leicht zu realisieren ist. Der OWS-Proxy legt die so per SPARQL ausgewerteten Konfigurationseinstellungen in eigens dafür vorgesehene objektorientierte Zugriffsklassen ab, die allen Proxy-Systemmodulen bereitstehen.

Modul: Anfrageanalyse/ Sprachabbildung

Der Sesame HTTP-Server wandelt die vom Benutzer empfangene SPARQL-Anfrage der Query-Form in einen SPARQL Algebra-Ausdruck um. Der Algebra-Ausdruck liegt daraufhin als Objektinstanz vor, die den äußersten Algebra-Operator repräsentiert (z.B. `Project` oder `Filter`; siehe Codebeispiel 18) und weitere Operatoren, Graphen- bzw. Tripelmuster in einem *Anfragebaum* schachtelt (engl. *Query tree*). Für all jene SPARQL-Strukturelemente sind Objektklassen in einem *Query model* verfügbar. Mit Hilfe von Sesame-Funktionsklassen, die dem *Visitor*-Programmiermuster entsprechen, kann der Anfragebaum vom äußersten Operator bzw. Wurzelknoten, bis in die einzelnen Blattknoten, u.a. Tripelmuster und Filterausdrücke, traversiert werden. Die Analyse im OWS-Proxy durchläuft den Anfragebaum genau zweimal. Der erste Durchlauf untersucht die Tripelmuster, hält ihre exakten Positionen im Anfragebaum fest und leitet Prädikatpfade wie auch GML-Elementpfade ab (siehe Theorie im Abschnitt 5.2.1). Die dabei stattfindende Verkettung und Validierung von GML-Elementpfaden ist mit den Modellklassen der XPath-Parserbibliothek *Jaxen*[151] umgesetzt. Anfragemanipulierende Maßnahmen, wie das Query Rewriting oder das Reasoning über Konzepte aus Upper-Ontologien, würden nun sinnvollerweise im Programmablauf folgen. Für die Demonstration der primären Proxy-Funktionalität spielen sie jedoch keine Rolle und sind nicht in die Entwicklung eingeflossen.

Beim erneuten Traversieren des Anfragebaums gilt das Augenmerk der Sprachabbildung von SPARQL nach FE. Dabei wird analog zum Leitpfaden der Sprachabbildung (siehe Abschnitt 5.2.2, Tabellen 5.5 und 5.6) jeder aufgefundene SPARQL-Operator in einen vergleichbaren FE-Operator übersetzt. Die dabei resultierende FE-Operatorenhierarchie ist nachträglich in eine XML-formatierte WFS GetFeature-Anfrage zu serialisieren (Stichwort: *XML-Marshalling*) und kann in dieser Form an einen oder mehrere Downloaddienste gesendet werden. Für eine vollständig automatisierte Sprachabbildung ist auch seitens der FE-Operatoren ein objektorientiertes Query Model, wie es Sesame für SPARQL bereithält, vonnöten, damit eine generische Überführung auf Objektebene von einem in das nächste Query Model stattfinden kann. Ein derartiges Query Model für FE bzw. WFS-Query Objektklassen ist im Deegree-Framework enthalten, aus dem auch der hier eingesetzte INSPIRE-Downloaddienst stammt. Die dazugehörigen Objektklassen wurden in der Masterarbeit einer kleinen Überarbeitung unterzogen, um nicht nur sachattributive FE-Filterausdrücke, sondern auch GML-Vergleichsgeometrien serialisieren zu können und die Nutzung aller WFS-Query-Elemente und -Attributierungen zuzulassen. Die abgeschlossenen Korrekturen und Erweiterungen wurden in die Codebasis des Deegree-Projektes[152] zurückgespielt.

Modul: Federation/ WFS-RequestHandling

Die Ergebnisse der Sprachabbildung sind WFS-Query XML-Fragmente, die als Bestandteile einer WFS GetFeature-Anfrage Filterungen für je einen WFS-Feature Type enthalten. Daraus lässt sich für jeden im Ressourcen-Repository registrierten Downloaddienst mit Berücksichtigung seines Feature Type-Angebotes eine maßgeschneiderte GetFeature-Anfrage zusammenstellen. Die konkreten Dienstaufrufe erfolgen parallelisiert (*Multithreading*) und unter Verwendung der Bibliothek *HttpClient* des *Apache HttpComponents*-Projektes[153] für die Abwicklung einer Client-seitigen HTTP-Kommunikation mit den INSPIRE-Downloaddiensten.

Modul: Ergebnistransformation

Der vom Downloaddienst exportierte GML-Datenstrom wird in der Proxy-Anwendung unmittelbar zur Initialisierung eines XML-Parsers weitergeleitet, um die GML-Inhalte auszulesen. Die Wahl des XML-Parsers fiel auf die Software *VTD-XML*[154], dem nach eigenen Angaben derzeit schnellsten XML-Parser. Es handelt es sich um einen nicht-extrahierenden und XPath 1.0-unterstützenden Parser (siehe Abschnitt 5.2.4). VTD-XML erwies sich in Testläufen als sehr effizient selbst bei umfangreichen GML-Datenmengen (>= 20MB), die in kürzester Zeit (ca. 0.5 s) in ein nur marginal größeres Indexfile - der Grundlage schnellster XPath-Auswertungen - umgewandelt werden konnten.

[151] Projektseite: http://jaxen.codehaus.org/
[152] Projektseite: http://www.deegree.org/
[153] Projektseite: http://hc.apache.org/
[154] Projektseite: http://vtd-xml.sourceforge.net/

Die Auswerteroutine in Codebeispiel 20 wurde über geschachtelte Parseraufrufe realisiert, die GML-Elementpfade als XPath-Ausdrücke unter Beachtung von GML-Namensräumen (*XML-Namespace awareness*) auswerten. VTD-XML stellt dazu eine Funktionsklasse `AutoPilot` bereit, die XPath-Ausdrücke absolut oder relativ zu einer Cursorposition im Indexfile evaluiert und das Iterieren über die damit adressierten Elemente ermöglicht. Eine weitere Besonderheit bietet die Funktionsklasse namens `Navigator`, die sich - sofern angewiesen - eine einmal aufgerufene Cursorposition merkt und mittels einem Sprungbefehl dahin zurückkehren kann. Beide Werkzeuge verhelfen zu einem exakten Navigieren durch den GML-Dokumenteninhalt ohne unnötig zwischengeschaltete Lesevorgänge, die nur der Navigation durch den Elementbaum und nicht dem Abgreifen konkreter Inhalten geschuldet sind.

Ein Manko weist VTD-XML dennoch auf. Es fehlt der Software an der XLink-Unterstützung, die zum Auflösen von GML-Objektreferenzierungen nötig ist. Im Prototypen wurde deshalb ein Notbehelf speziell für Verlinkungen von INSPIRE-Features eingeführt, welcher der umfangreichen Parser-Auswertung einen kurzen Lesevorgang über alle Feature-Elemente voranschaltet. Jedem aufgefundenen Feature wird vorab die individuelle Ressourcen-Id generiert (siehe Abschnitt 5.1.2 *Identifikationsmanagement*) und diese mitsamt dem zugehörigen XLink-Anker, dem GML-Featureattribut `gml:id`, zwischengespeichert. Sobald danach die umfangreiche Parser-Auswertung auf GML-Objektreferenzierungen stößt, werden die registrierten `gml:id`s dazu verglichen. Im Falle eines Treffers ist von der `gml:id` auf die Ressourcen-Id zu schließen, die letztlich als Objekt eines neuen RDF-Tripels dient (siehe hierzu auch die Beschreibung zum Pseudocode in Codebeispiel 20).

Darüber hinaus verzichtet die prototypische Entwicklung auf die Generierung der Ressourcen-Ids für abhängige Objektinstanzen, die den INSPIRE UML-Klassen des Stereotyps `dataType` zugehören. Der Programmieraufwand wäre aller Voraussicht nach zu hoch und für einen Prototypen nicht zu rechtfertigen gewesen. Die sich daraus ergebenden Konsequenzen werden im nächsten Abschnitt 5.3.4 im Rahmen der Testläufe besprochen.

Modul: GIS-Analyse

Die GIS-Analyse ist ein wichtiger Teilaspekt des Prototypen. Zur professionellen GIS-Filterunterstützung innerhalb von SPARQL-Anfragen wurde der Sesame HTTP-Server um GeoSPARQL-Filterfunktionen erweitert. Die Registrierung dieser benutzerdefinierten SPARQL-Filterfunktionen (sogenannte *SPARQL extended filter functions*) ist in Sesame, vergleichbar zu SAIL-Klassentypen, über eine SPI-Schnittstelle gelöst und so einfacherweise konfigurierbar. Zu diesem Zweck wurden im Prototypen alle topologischen GeoSPARQL-Funktionen des Simple Features-Operatorensatzes sowie alle nicht-topologischen GeoSPARQL-Funktionen implementiert (siehe Grundlagenabschnitt 2.4.3). Die interne GIS-Prozessierung ist mit den Klassenbibliotheken des Deegree-Frameworks umgesetzt. Die Deegree-Klassen binden ihrerseits partiell die Bibliothek *Java Topology Suite* (JTS[155]) ein, wobei es sich um ein etabliertes GIS-Toolkit mit hoher Akzeptanz handelt, dessen primäres Ziel die Realisierung und Konformität zur Simple Features-Spezifikation ist.

Die entwickelten GeoSPARQL-Filterfunktionen können als Parameterwerte sowohl GML-Geometrien der Version 3.2.x als auch Simple Features WKT-Geometrien der Version 1.0 aufnehmen und verarbeiten. Die GIS-Auswertung der GeoSPARQL-Funktionen beschränkt sich allerdings allein auf die Verwendung geographischer WGS84-Koordinaten. Einerseits, um sich an der gängigen Praxis des Semantic Web zu orientieren, dessen Projekte fast ausnahmslos die WGS84-Georeferenzierung anwenden. Andererseits erscheint die getroffene Einschränkung unumgänglich, sofern man Client-seitige Koordinatentransformationen während der Anfragebearbeitung im OWS-Proxy und damit einhergende Performanzeinbußen unterbinden möchte. Glücklicherweise verlangt INSPIRE die CRS-Unterstützung für geographische und metrische ETRS89-Koordinaten[156] in den operativen Netzwerkdiensten ([INSPIRE „Network Services“ Drafting Team 2009], Abschnitt 2.6.2 *Coordinate Reference Systems*). Es ist also anzunehmen, dass viele Downloaddienste Datensätze bereitstellen, die originär im ETRS89-Datum vorliegen, und aufgrunddessen Daten-

[155] Projektseite: http://www.vividsolutions.com/jts

[156] das ETRS89-Datum ist mit dem WGS84-Datum nahezu identisch, demzufolge gleichen sich in grober Näherung auch die jeweiligen geographischen Koordinatenangaben

anfragen des OWS-Proxy nach geographischen ETRS89-Koordinaten ohne vorherige Koordinatentransformation, d.h. mit einem maximalen Aufwand einer Koordinatenumformung (Umprojektion) beantworten können.

Erweiterung: Linked Data-Endpoint

Der OWS-Proxy ist dank der Sesame-Unterstützung dazu in der Lage, SPARQL-Filteranfragen zu beantworten (Anwendungsfall: *RDF-Querying*). Der andere Zugang zu Semantic Web-Ressourcen ist derjenige des *RDF-Browsing*. Damit gemeint ist das gezielte Abfragen einzelner RDF-Ressourcen und das Navigieren über die jeweiligen Verlinkungen (HTTP-Hyperlinks) zu anderen RDF-Ressourcen - vergleichbar zum herkömmlichen Internet-Surfen und Navigieren über verknüpfte HTML-Seiten. Für das RDF-Browsing wird ein Linked Data-Endpoint benötigt, der funktional weniger beherrschen muss als ein SPARQL-Endpoint. Abgesehen von der Aushandlung des Rückgabeformates (*Content-Negotiation*) ist ein Linked Data-Endpoint im Wesentlichen für das Auflösen und die Ausgabe der semantischen Aussagen einer einzigen RDF-Ressource zuständig (siehe Abschnitte 2.2.3 und 2.2.4). Ist bereits ein SPARQL-Endpoint operativ, wie im Falle des OWS-Proxy, so kann ein Adapter zum Einsatz kommen, der auf den SPARQL-Endpoint aufsetzt und nach außen wie ein Linked Data-Endpoint funktioniert. Ein derartiges Linked Data-Frontend bietet die Software **Pubby**[157], ebenfalls eine Java-Webapplikation, die auf dem Apache Tomcat gehostet werden kann (siehe Architekturskizze in Abbildung 5.18)

Blickt man auf die Prozesskette einer Anfragebearbeitung, so empfängt Pubby zunächst eine Datenanfrage in Form einer URL-Adresse (z.B. `http://localhost:8080/pubby/DE.BW.PS/001`). Daraus leitet Pubby eine SPARQL `Select`-Filteranfrage ab und richtet sie an den SPARQL-Endpoint im Backend. Die SPARQL-Ergebnisse wandelt Pubby je nach gewünschtem Format z.B. in RDF/XML oder in eine HTML-Ansicht um und sendet sie an den Client zurück (siehe Praxisbeispiel in Abbildung 5.20). Das Konzept und auch die prototypische Implementierung sind so angelegt, dass Ressourcenidentifikatoren eine öffentliche Basisadresse beinhalten; hier die INSPIRE-Domäne `http://inspire-jrc.ec.europa.eu`. Der Aufruf und das Starten einer Datenabfrage an die INSPIRE-Adresse (z.B. zum Schutzgebiet der Id `http://inspire-jrc.ec.europa.eu/data/DE.BW.PS/001`) ist natürlich vergebens, solange dort noch kein Linked Data-Endpoint eingerichtet ist. Die lokale Pubby-Installation erlaubt stattdessen das **Adressen-Mapping** von der lokalen Adresse (angefangen mit `http://localhost:8080/pubby/...`) auf die intern vom SPARQL-Endpoint gepflegte INSPIRE-Adresse (angefangen mit `http://inspire-jrc.ec.europa.eu/data/..`) und umgekehrt. Somit sind alle Vorbedingungen erfüllt, um die Publikation von INSPIRE-Ressourcen sowohl über einen SPARQL- wie auch einen Linked Data-Endpoint zu prüfen.

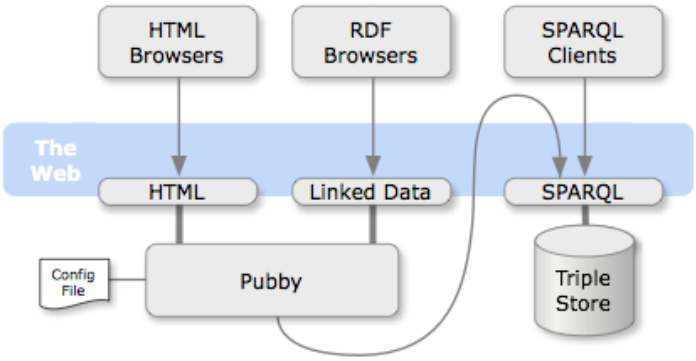


Abbildung 5.18 – Architekturskizze des Linked Data-Frontends Pubby[158]

[157] Projektseite: http://www4.wiwiss.fu-berlin.de/pubby/
[158] Quelle: Pubby-Dokumentation unter: http://www4.wiwiss.fu-berlin.de/pubby/

Erweiterung: SPARQL-Federator

Prinzipiell lässt sich mit der Anfragesprache SPARQL nicht nur eine einzige Wissensbasis analysieren, sondern mehrere Datenquellen gleichzeitig anfragen und innerhalb eines Fragekontextes zueinander in Beziehung setzen. Die Anwendungsszenarien II bis IV aus Kapitel 3 setzen jeweils voraus, dass eine intelligente Software-Komponente, *Dispatcher* oder *Federator* genannt, Anfragebestandteile interpretiert, an datenbereitstellende Quelldienste weiterleitet und deren Ergebnisse bündelt. Der OWS-Proxy ist so konzipiert, dass er die Rolle eines Federators für INSPIRE-Downloaddienste übernimmt, nicht aber eine verteilte Suche über mehrere SPARQL-Endpoints. Letzteres ist auch eher eine allgemeine Aufgabenstellung des Semantic Web, das zu diesem Zweck bereits einige Systemlösungen hervorgebracht hat. Um ein Anwendungsszenario mit verteilten Datenquellen durchzuspielen, wurden zwei Open-Source Komponenten getestet, die systemnah wie auch der prototypische OWS-Proxy auf dem Sesame-Framework basieren: das *Federation Sail*[159] des Projektes *Alibaba*, einem Schwesterprojekt von Sesame, sowie die Eigenentwicklung der Universität Koblenz-Landau namens *Distributed Sail*[160].

5.3.4 Testergebnisse

Die durchgeführten Software-Tests sind eingeteilt in **vier Testszenarien**, die sich in ihrer Datengrundlage und anhand des Software-Einsatzes unterscheiden (siehe Übersicht über die Testszenarien in Tabelle 5.8). Bei den Testläufen bestand die Datengrundlage entweder aus INSPIRE-Daten eines einzigen Annex-Themas (Szenarien I und II), aus INSPIRE-Daten mehrerer Annex-Themen (Szenario III) oder aus gemischten Linked Data-Datentöpfen (INSPIRE und LinkedGeoData; Szenario IV). Die Datenabfragen wurden entweder direkt an den SPARQL-Endpoint des Prototypen gerichtet (Szenarien I, III und IV) oder an die vorgelagerte Schnittstelle eines Linked Data-Endpoints (Szenario II), realisiert mit der Software *Pubby* (siehe vorangegangenen Abschnitt 5.3.3). Neben der Überprüfung auf Anwendbarkeit und einer zweckmäßigen Software-Unterstützung für die jeweiligen Datenanfragen, werden für jedes Testszenario ausgewählte Prüfaspekte betrachtet (siehe Spalte 3 in Tabelle 5.8). Desweiteren weisen zwei Testszenarien einen direkten Bezug zu je einem Anwendungsszenario aus Kapitel 3 auf, unterziehen Letztere also einem Praxistest (siehe Spalte 4).

Tabelle 5.8 – Übersicht der Testszenarien

Nr.	Beschreibung des Testszenarios	Prüfaspekte	Anwendungs szenarien
I	Publikation von INSPIRE-Daten per SPARQL-Endpoint (RDF-Querying)	a) Funktionstüchtigkeit von Filterprozessen b) Inhaltliche Vollständigkeit c) Perfomanz	Szenario I
II	Publikation von INSPIRE-Daten per Linked Data-Endpoint (RDF-Browsing)	a) Transparentes RDF-Browsing b) Perfomanz	–
III	Verteilte Suche über verschiedene INSPIRE-Datenthemen (per SPARQL-Endpoint, RDF-Querying)	a) Suche über statische Ressourcenverlinkungen b) Suche über temporäre Ressourcenverlinkungen (räumliche Filterungen)	–
IV	Verteilte Suche über Linked Data- und INSPIRE-Daten (per SPARQL-Endpoint, RDF-Querying)	a) Suche über statische Ressourcenverlinkungen b) Suche über temporäre Ressourcenverlinkungen (räumliche Filterungen)	Szenario II

[159] Dokumentation: http://www.openrdf.org/doc/alibaba/2.0-beta6/alibaba-sail-federation
[160] Dokumentation: http://www.uni-koblenz-landau.de/koblenz/fb4/AGStaab/Research/systeme/DistributedSPARQL

Testszenario I

Um die **Funktionstüchtigkeit von Filterprozessen** zu überprüfen, wurden die baden-württembergischen Schutzgebietedaten (2. Datensatz; siehe Abschnitt 5.3.1) ausgiebig gefiltert. Dies geschah nach den Kriterien des Anwendungsszenarios I (siehe Kapitel 3). Darin versucht ein Naturfreund herauszufinden, ob die von ihm beobachteten Waldschäden innerhalb von Schutzgebietgrenzen angerichtet wurden. Als SPARQL-Client wurde die Software *Twinkle*[161] verwendet. Twinkle ermöglicht rudimentäre SPARQL-Anfragen an einen standardkonformen SPARQL-Endpoint, hier den prototypischen OWS-Proxy, ohne dabei aufwendige GUI-Elemente anzubieten. Der Screenshot in Abbildung 5.19 zeigt den Gebrauch von Twinkle, dessen zweigeteiltes Fenster SPARQL-Anfragen vom Benutzer entgegennimmt (oberer Fensterbereich) und die zurückgelieferten Ergebnisse zur Anzeige bringt (unterer Bereich). Twinkle wurde außerhalb dieses Dialoges mit der Dienstadresse des OWS-Proxy konfiguriert. Die SPARQL-Anfrage sucht über eine räumliche Einschränkung nach einem Schutzgebiet (`?schutzgebiet`) und gibt im Trefferfalle seinen Namen (`?schutzgebietName`), seine Schutzwidmung (`?designation`) und optional die zuständige Koordinierungsstelle aus (`?agencyName`). Gefunden werden diejenigen Schutzgebiete, die die vom Naturfreund mitgetrackte Fundstelle, hier beispielhaft mit der Punktgeometrie `POINT(8.91 49.39)` wiedergegeben, topologisch enthalten (Funktionsaufruf `geof:contains`).

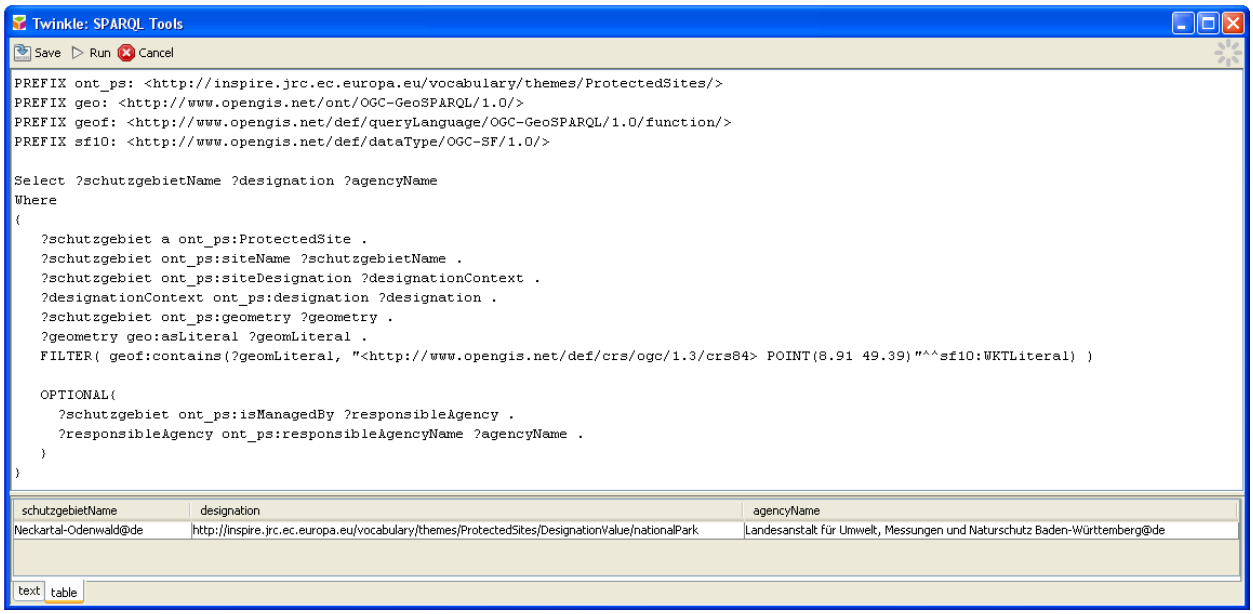


Abbildung 5.19 – Testszenario I, SPARQL-Anfrage und Ergebnisse

Nach Abschicken der Anfrage wird in der Ergebnisanzeige ein Treffer eingeblendet, es handelt sich um das Schutzgebiet *Neckartal-Odenwald*. Die räumliche Filterung über den OWS-Proxy kann demnach erfolgreich durchgeführt werden. Weitere Tests innerhalb der folgenden Testszenarien zeigten darüber hinaus gelungene Filterungen mittels anderer räumlicher Operatoren (z.B. räumliche Verschneidungen, Bufferbildungen) und Filterungen über Sachattribute (z.B. Codelist-Werte und Zeitangaben). Die inhaltliche Richtigkeit der Ergebnisse räumlicher Filterungen ließ sich unabhängig vom OWS-Proxy in einem Desktop-GIS und auf Grundlage der INSPIRE GML-Datensätze nachvollziehen. Neben SPARQL-Objektwertvergleichen werden im Prototypen auch SPARQL-Pfadmustervergleiche berücksichtigt und in Filter Encoding-Ausdrücke der Operatorensequenz `<ogc:Not><ogc:PropertyIsNull>` umgewandelt.[162] Die Übersetzungen in WFS GetFeature-Requests sind für diese wie auch alle folgenden SPARQL-Anfragen in Anhang E eingefügt.

[161] Projektseite: http://www.ldodds.com/projects/twinkle/
[162] die Begriffe SPARQL Objektwert- und Pfadmustervergleiche sind definiert in Abschnitt 5.2.2

Die vom OWS-Proxy zurückgereichten SPARQL-Resultate, wie z.B. der Ergebnistreffer Schutzgebiet *Neckartal-Odenwald*, wurden auf **inhaltliche Vollständigkeit** geprüft. Es ließ sich grundsätzlich feststellen, dass GML-Inhalte, die selektiv über SPARQL-Variablen adressiert und per XPath-Ausdrücke aus den WFS-Ergebnissen gefiltert werden, einwandfrei auf Semantic Web-Ressourcen und -Aussagen abgebildet und im SPARQL-Client (hier *Twinkle*) dargestellt werden können. Sofern die GML-Eingabedaten Elemente beinhalten, die das Fehlen von Informationen umschreiben (charakteristisches Attribut `xsi:nil`; anwendbar für alle INSPIRE-Elemente des Stereotyps `voidable`), werden jene Elementinhalte entsprechend der Konzeptausführungen ignoriert und nicht in Semantic Web-Ressourcen und -Aussagen übertragen (siehe Abschnitt 5.1.1).

Um die Effizienz der Sprachabbildungs- und Datenaufbereitungsprozesse, die innerhalb des OWS-Proxy ablaufen, in Zahlen ausdrücken zu können, fanden geringfügige und dennoch aussagekräftige **Perfomanztests** statt. Durchgeführt wurden sie lokal auf einem Desktop-Rechner, um die variable Netzwerkübertragung auszublenden, die sich in ihrer Geschwindigkeit aufgrund der gegebenen Randbedingungen[163] erheblich unterscheiden kann. Die Leistungseckdaten des lokalen Rechners sind wie folgt: CPU: INTEL® Core™ 2 DUO P7450 2,13GHz; Arbeitsspeicher: 3GB RAM (DDR3). Weil die eingesetzte Software, d.h. der OWS-Proxy Prototyp im Sesame-Container und der Deegree3 inspireNode, als Webapplikationen auf dem gleichen Apache Tomcat-Server liefen, wurde dem Apache Tomcat genügend Arbeitsspeicher für Java-Prozesse zugewiesen (max. 1024MB RAM, zusätzlich 512MB RAM für permanente Speicherungen). Der zugewiesene Speicher wurde im Laufe der Tests von den Webapplikationen nie vollends ausgenutzt. Der markanteste Perfomanztest behandelte die Abfrage aller 433 slovakischen und baden-württembergischen Schutzgebiete (ca. 13,1 MB, hauptsächlich Geometriedaten) inklusive der Selektion (XPath-Auswertung) diverser INSPIRE-Attribute der Applikationsschemata Protected Sites Simple und Full, wie z.B. des Schutzgebietnames (`ps:siteName`), der Gesetzesgrundlage (`ps:legalFoundationDocument`), der zuständigen Stelle (`ps-f:ResponsibleParty`), der Gültigkeitsangaben zum Speicherobjekt (`ps-f:beginLifespanVersion`) usw. Es konnte beobachtet werden, dass die Auswertung der SPARQL-Anfrage sowie ihre Abbildung auf Filter Encoding-Ausdrücke (Prozessschritte 2 und 3; siehe Skizze 5.9) weniger als eine Sekunde, durchschnittlich *ca. 0,5s* bedurfte. Die deutlich aufwendigere Aufbereitung und Umformung der GML-Ergebnisse in Semantic-Web Entsprechungen (inklusive GeoSPARQL-Speicherung in Literalen, Prozessschritt 5) benötigte *ca. 14s*. Insgesamt dauerte eine lokal stattfindende Abfrage aller 433 Schutzgebiete vom Abschicken der Anfrage bis zur Darstellung der Ergebnisse im Twinkle-Client *unter 20s*, was angesichts der angehenden Massendatenverarbeitung (13,1 MB) sehr performant erscheint.

Testszenario II

Im zweiten Testszenario ist die Publikation von INSPIRE-Ressourcen über einen Linked Data-Endpoint (hier der *Pubby*-Adapter) zu untersuchen. Der Bildschirmabzug in Abbildung 5.20 zeigt die Anfrage nach einer Schutzgebiete-Ressource über den URL-Aufruf `localhost:8080/pubby/DE.BW.PS/67_919014000001`. Die Eigenschaften bzw. die semantischen Aussagen zur Ressource werden in einer HTML-Darstellung in einer beliebigen Reihenfolge aufgelistet und vorhandene Ressourcenlinks als HTTP-Hyperlinks zum *RDF-Browsen* aufbereitet. Beispielsweise wird als fünfte Schutzgebieteeigenschaft namens `isInRegion` ein Verknüpfung zu der das Schutzgebiet umgebenden Region aufgeführt (Ressourcenidentifikator `http://localhost:8080/pubby/EU.BGR/1`). Dabei handelt es sich um eine Ressourcenverlinkung, die in den INSPIRE-Datenmodellen enthalten ist und im nächsten Testfall näher betrachtet wird.

Wirft man einen Blick auf die Objekteigenschaften `inspireId`, `legalFoundationDocument` und `siteDesignation`, so findet man diese in Pubby klassifiziert als *anonyme Ressourcen* vor (`[1 anonymous resource]`). Dahinter verbergen sich INSPIRE-Objektinstanzen, deren Identifikatoren von anderen Objektinstanzen abhängen (siehe Abschnitt 5.1.2) und für die der prototypische OWS-Proxy anstelle der konzeptionell vorgesehenen auflösbaren HTTP-URIs (angefangen mit `http://inspire.jrc.ec.europa.eu/data/...`) lediglich Blank Nodes generiert (z.B. `_:node_33a41`).

[163] charakteristisch für die jeweilige Datenübertragungsrate ist der Übermittlungsweg, d.h. durch welche Datennetzwerke (Internet, WAN, (W)LAN) die Informationen führen und wieweit der datenliefernde Netzknoten entfernt ist

Mit dieser Art der Umsetzung ließ sich der Implementierungsaufwand des Prototypen reduzieren. Während SPARQL-Anfragen davon nicht tangiert werden und problemlos funktionieren, wirkt sich die Generierung von Blank Nodes hingegen erschwerend auf das RDF-Browsing aus, dessen **Transparenz** verloren geht, indem das Navigieren zu den von den Blank Nodes repräsentierten RDF-Ressourcen unterbunden wird. Dieser Aspekt ist bei Inbetriebnahme eines operativen OWS-Proxy zu berücksichtigen und, sofern RDF-Browsing unterstützt werden soll, eine durchgehende Ressourcenidentifikation auch für abhängige Objektinstanzen anzustreben.

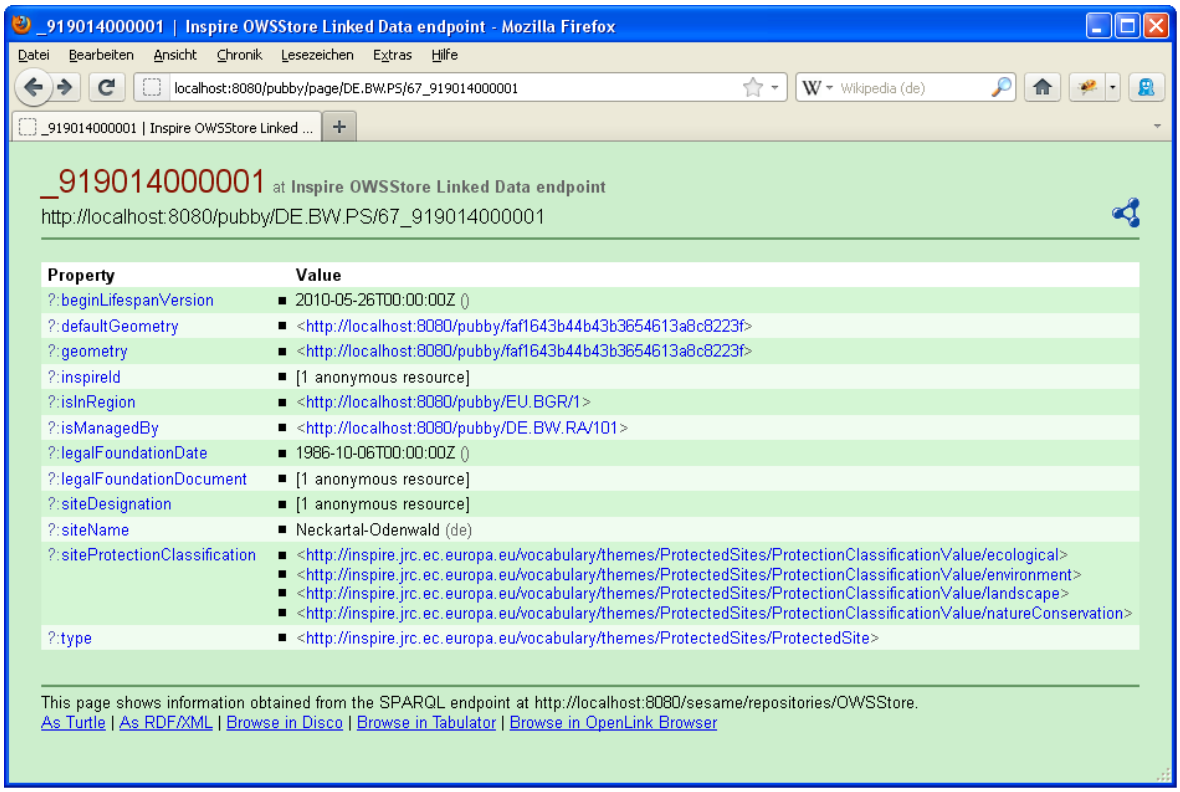


Abbildung 5.20 – Testszenario II, Ergebnisausgabe eines Linked Data-Endpoints

Es stellte sich während des Praxistests heraus, dass das Navigieren über Ressourcenverlinkungen auf eindeutige Adressen, wie z.B. dem Objektwert der Eigenschaft `isManagedBy`, sehr schleppend funktioniert. Dies erscheint logisch, betrachtet man sich die von Pubby an den OWS-Proxy weitergeleiteten SPARQL-Anfragen. So zeigt das Codebeispiel 21 eine typische SPARQL-Anfrage von Pubby, die nach allgemeinsten Tripelmustern sucht (Zeile 4), deren Subjekte oder Objekte der gesuchten Ressource (hier `http://inspire.jrc.ec...1400001`) entsprechen müssen (Filterungen in den Zeilen 5 und 6).

```
1  SELECT ?subject ?predicate ?object
2  WHERE
3  {
4    ?subject ?predicate ?object .
5    FILTER ( SameTerm( ?subject , <http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/67_919014000001>)
6     || SameTerm( ?object , <http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/67_919014000001>) )
7  }
```

Codebeispiel 21 – Anfrageweiterleitung von Pubby an die SPARQL-Datenquelle

Der Prototyp ist darauf ausgerichtet, den INSPIRE-Namensraum (hier `DE.BW.PS`) auf die mit dem Namensraum registrierten INSPIRE-Downloaddienste und ihrer angebotenen Feature Types aufzulösen (siehe Abschnitt 5.1.2

Identifikationsmanagement) und der gesuchten Ressource auf diese Weise potentielle Objekttypen (WFS-Feature Types) zuzuordnen. Zusätzlich zur Objekttypisierung wäre es jedoch ebenso zweckmäßig, den INSPIRE-Identifikator als weiteres Filterkriterium einzusetzen. Berücksichtigt man beide Filterkriterien und führt ein internes Query-Rewriting der SPARQL-Anfrage durch, so enthält die umformulierte SPARQL-Anfrage wie im Codebeispiel 22 dargestellt sowohl eine Objekttypeneinschränkung (Zeile 8) als auch eine Filterung über die INSPIRE-Identifikatorattribute `localId` und `namespace` (Zeile 9 bis 11). Die GetFeature-Antwort(en) ein oder mehrerer Donwloaddienste belaufen sich dann auf maximal ein Schutzgebiet-Feature, dessen Umwandlung in semantische Aussagen gegenüber derjenigen aller Schutzgebiete-Features nur einen Bruchteil der Berechnungszeit benötigt. Das RDF-Browsen mittels Pubby würde durch die beschriebene Maßnahme merklich beschleunigt.

```
1  PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
2  PREFIX ont_base: <http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes/>
3
4  SELECT ?subject ?predicate ?object
5  WHERE
6  {
7    ?subject ?predicate ?object .
8    ?subject a ont_ps:ProtectedSite .
9    ?subject ont_ps:inspireId ?identifier .
10   ?identifier ont_base:localId "67_919014000001" .
11   ?identifier ont_base:namespace "DE.BW.PS" .
12 }
```

Codebeispiel 22 – Per Query-Rewriting modifizierte Anfrage an die SPARQL-Datenquelle

Testszenario III

Gegenstand des dritten Testszenarios sind SPARQL-Anfragen, die ein Verschneiden und gleichzeitiges Auswerten von Ressourcen verschiedener INSPIRE-Thematiken erlauben. Für dieses Testszenario wurden GML-Features des Annex-Themas *Bio-geographische Regionen* neuerstellt (4. Datensatz in Tabelle 5.7; Feature Type: `Bio-geographicalRegion`) und mit einigen Schutzgebieten der Slovakei bzw. Baden-Württembergs verknüpft (Datensätze 1 und 2, Feature Type: `ps-f:ProtectedSite`). Die jeweilige **statische Objektreferenzierung** über die Assoziation `ps-f:isInRegion` (siehe UML-Modell der INSPIRE-Datenspezifikation Schutzgebiete in Abbildung 2.9) wurde der GML-Datengrundlage des Downloaddienstes hinzugefügt. Bei der Ergebnistransformation GML nach RDF/OWL folgt daraus eine ontologische Aussage der Form *<URI des Schutzgebietes A>* `ont_ps:isInRegion` *<URI der Region B>*. Die diesbezügliche SPARQL-Abfrage zur Beantwortung der Fragestellung, welche Schutzgebiete in welcher Region gelegen sind, konnte erfolgreich getestet werden und ist zusammen mit dem Ergebnis im Anhang E dokumentiert. Die wesentlichen Anfragebestandteile sind in der oberen SPARQL `Select`-Anfrage in Abbildung 5.21 aufgeführt. Das zusammenhängende Graphenmuster enthält Tripelmuster zum Auffinden von Schutzgebieten und zugehöriger Eigenschaften (Zeile 3), Tripelmuster zur Suche nach biogeographischen Regionen und zugehöriger Eigenschaften (Zeile 5) und das verknüpfende Tripelmuster in Zeile 7. Konkrete Verknüpfungen sind im abstrahierten Graphen $\mathtt{G_a}$ mit roten Knotenkanten zwischen Schutzgebieteressourcen ($\mathtt{PS_x}$) und Regionenressourcen ($\mathtt{BGR_x}$) dargestellt. Nach Auswertung des Graphenmusters ($\mathtt{BGP(G_a)}$) ergibt sich eine minimale RDF-Ergebnisrelation ($\mathtt{R_1}$) mit passgenauen Zuordnungen von Schutzgebieten (SPARQL-Variable `?schutzgebiet`) und Regionen (Variable `?region`).

Sofern statische Ressourcenverlinkungen fehlen, können verschiedenartige Ressourcen über Filter-Anweisungen auch temporär, d.h. zur Laufzeit der Datenanfrage, miteinander verknüpft werden. Da Objekte häufig einen Raumbezug aufweisen, eignen sich insbesondere **räumliche Filterungen** für diese Art von Datenanfragen. In der herkömmlichen GIS-Analyse ist es sogar üblich, die von einer konkreten Fragestellung betroffenden topologischen Objektbeziehungen nur mit Hilfe **temporärer Auswertungen** zu identifizieren. Statisch gespeicherte Topologiesachverhalte

bilden hingegen eher die Ausnahme (siehe Abschnitt 2.3.3). Übertragen auf eine SPARQL `Select`-Anfrage (siehe Anfragebeispiel b) in Abbildung 5.21) wird anstelle des verknüpfenden Tripelmusters eine Filteranweisung mit einem räumlichen Operator eingefügt (hier der GeoSPARQL-Operator `geof:intersects`; Zeile 7). Die Anfrage wird gegen den Beispielsgraphen G_b gerichtet, der keine Ressourcenverlinkungen (rote Knotenkanten) aufweist, so dass aus der Graphenmustererkennung eine RDF-Relation R_1 als Kreuzprodukt aller Schutzgebiete (PS) mit allen Regionen (BGR) hervorgeht. In einem zweiten Schritt wird der räumliche Filter $\sigma_{intersects}$ auf alle Ergebniszeilen aus R_1 angewandt und es folgt daraus die finale RDF-Relation R_2. Die RDF-Relation R_2 liefert dieselben Ergebnisse, die auch mit einer Suche über statische Ressourcenverlinkungen und unter der Bedingung vorhandener statischer Topologieprädikate erzielt werden kann (siehe Anfragebeispiel a) in Abbildung 5.21).

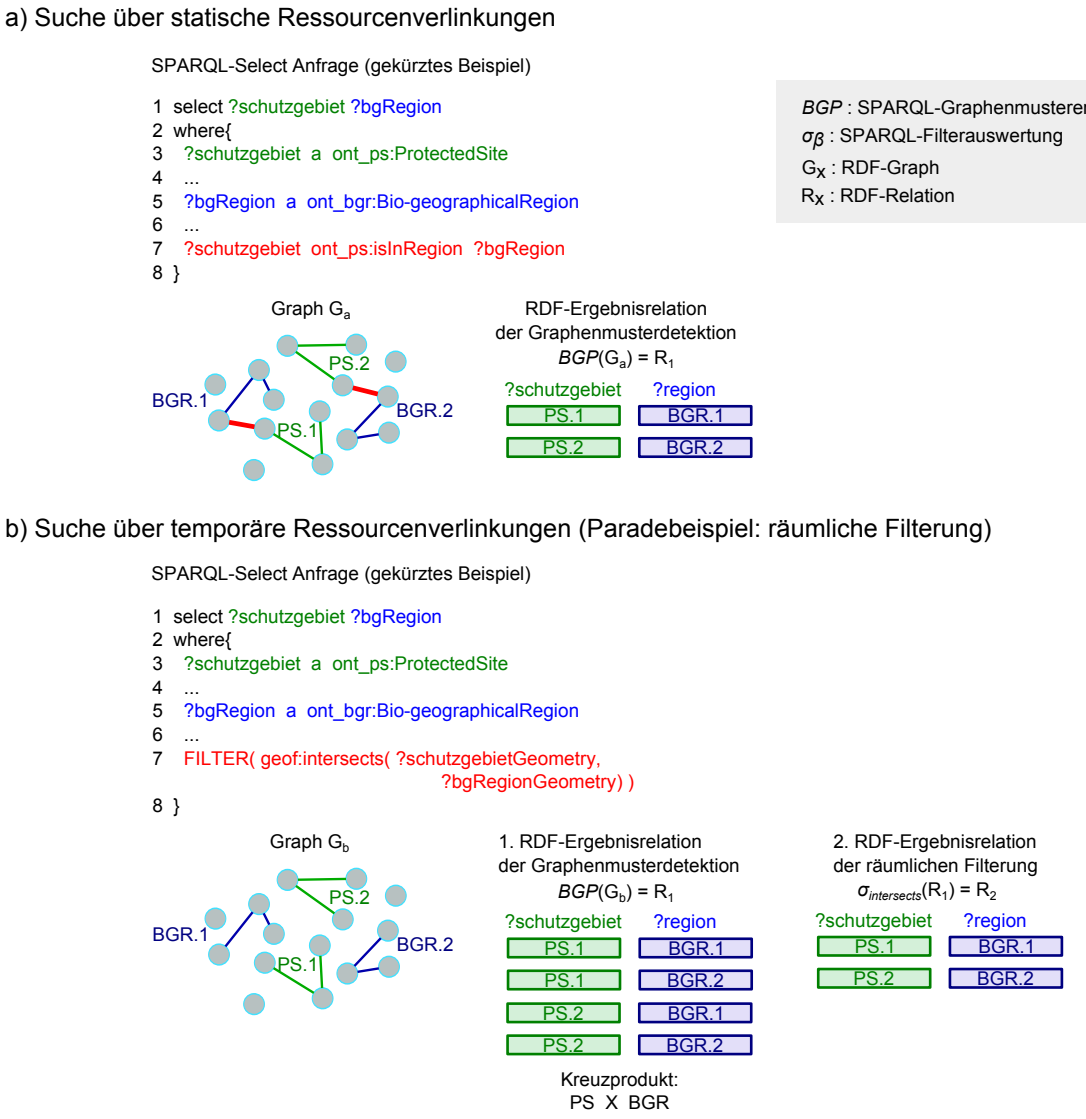


Abbildung 5.21 – Testszenario III, SPARQL-Auswertung über statische und temporäre Ressourcenverlinkungen

Problematisch ist die **Bildung von Kreuzprodukten**, die in Anfragebeispiel b) als Zwischenergebnisse entstehen. Zwar sind die Kreuzprodukte erforderlich, um die voneinander unabhängigen Ressourcen temporär in Beziehung zu setzen, jedoch werden durch den anschließenden Filtervorgang leichthin Tausende von räumlichen Abfragen ausgelöst. So führt die Verschneidung von 100 Schutzgebieten mit 100 Regionen bereits zu 10000 GIS-Operationen. Wie die Testläufe zeigten, verzögerte die Verschneidung von allen 433 Schutzgebiete mit 7 Regionen (Anfragebeispiel b)

die Ergebnisauswertung gegenüber der Abfrage über statischen Ressourcenverlinkungen (Anfragebeispiel a) spürbar. Vermeiden lässt sich dies durch frühzeitiges Filtern der zu verschneidenden Objektmengen, sofern diese zweckmäßig vorselektiert werden können. Alternativ ist eine Art **GIS-Vorprozessierung** anzustreben, die im Abschnitt 5.2.5 diskutiert wurde. Dabei können die Geometrien vor der eigentlichen SPARQL-Auswertung in eine Geometrietabelle eingelesen und dank eines räumlichen Indexes schneller analysiert werden. Obwohl der Index jedesmal neu angelegt bzw. aktualisiert werden muss, sorgt er für eine erhebliche Beschleunigung der GIS-Prozessierung. Die ermittelten topologischen Beziehungen lassen sich sodann der Wissensbasis als statische Ressourcenverlinkung anfügen, für deren Analyse wiederum ein SPARQL-Prozessor prädestiniert ist. Soviel zur Theorie, die praktischen Testläufe wurden hingegen ohne GIS-Vorprozessierung ausgeführt und erfüllten zumindest im kleinen Rahmen, d.h. mit wenigen Objekten (433 Schutzgebiete; 7 Regionen), die gehegten Erwartungen. Die verwendeten SPARQL-Anfragen, je ein Beispiel für temporäre und statische Verlinkungen, sind mitsamt ihren Ergebnisdarstellungen dem Anhang E zu entnehmen.

Testszenario IV

Das Anwendungsszenario II aus Kapitel 3 soll nun den Anwendungskontext vorgeben und Gelegenheit bieten, sich mit den Erfordernissen einer verteilten Suche zu befassen. Im gegebenen Szenario bereitet ein Lehrer einen Klassenausflug vor, dessen primäres Ziel die Naturerkundung in einem stark reglementierten Schutzgebiet ist, wie z.B. einem Nationalpark. Neben dem Schutzgebiet, das vom Schulort eine Mindestentfernung von 50km aufweisen soll, sucht der Lehrer außerdem nach Schullandheimen (Hostels) in der näheren Umgebung des Schutzgebietes (maximale Entfernung: 3km). Für dieses Anwendungsszenario werden drei Datenquellen bzw. Objektkategorien benötigt: 1. Schutzgebiete, 2. die Schule (beschränkt auf die Geometrie des Schulortes oder genauer auf jene des Schulstandortes) und 3. Hostels für die Übernachtung. Die Schutzgebiete können den INSPIRE-Quellen entstammen bzw. über den SPARQL-Endpoint des OWS-Proxy abgerufen werden. Die Informationen und Punktgeometrien der Schule und der Hostels lassen sich dem SPARQL-Endpoint des Projektes *LinkedGeoData* entnehmen.

Das Anwendungsszenario erfordert also eine verteilte Suche (*Federation*), in der zwei oder mehr separate SPARQL-Datenquellen angefragt werden. Da die der verteilten Suche zugrunde liegenden Prinzipien bislang nur am Rande angeklungen sind, soll vor der praktischen Umsetzung zuerst die Theorie besprochen werden. Die Abbildung 5.22 veranschaulicht die Ergebnisverarbeitung zweier Auswertestrategien. Werden entfernte SPARQL-Endpoints mit Anfragen der Query form SPARQL `Select` aufgerufen (Auswertestrategie a), liefern sie RDF-Relationen (R_x) zurück, die in der Ergebnisaufbereitung des Federators miteinander relational verknüpft (`Join`-Operation; R_1 * R_2) oder relational vereinigt werden (`Union`-Operation; R_1 U R_2).

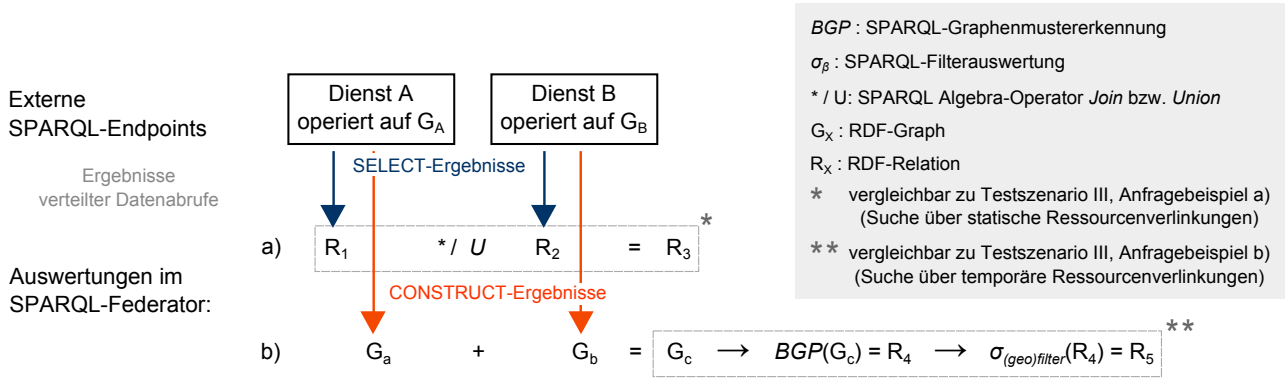


Abbildung 5.22 – Testszenario IV, Verteilte SPARQL-Suche und Ergebniszusammenstellung

Die Auswertung a) entspricht im Falle einer `Join`-Verknüpfung dem Anfragebeispiel a) aus Abbildung 5.21, das im Zuge der Graphenmustererkennung (`BGP(G)`) vergleichbare `Join`-Operationen auf Graphenbestandteile ausführt. Der Datenabruf per SPARQL `Select` eignet sich also für statische Ressourcenverlinkungen. Im gegebenen Szenario müssten demnach die Ressourcen der Schutzgebiete, der Schule und der Hostels mit entsprechenden ontologischen Aussagen verknüpft sein, die entweder der INSPIRE-Wissensbasis (vgl. Datenquelle D in Abbildung 5.13) oder der LinkedGeoData-Wissensbasis entstammen. Die Auswertestrategie a) lässt sich jedoch nicht für räumliche Abfragen anwenden, welche temporäre Ressourcenverlinkungen, entstanden durch ein Kreuzprodukt, analysieren und filtern (vgl. Testszenario III). Hierfür ist die Auswertestrategie b) mit verteilten Anfragen der Query form SPARQL `Construct` erforderlich. Sie liefert Graphenergebnisse (G_a und G_b), die zu einem gemeinsamen Anfragegraphen (G_c) zusammengesetzt werden (*Merging* mehrerer RDF-Graphen). Auf Grundlage des vereinten Graphen G_c können dann die bekannten Auswerteschritte des Anfragebeispiels b) aus Abbildung 5.21 folgen: erstens eine Graphenmustererkennung zur Erstellung eines Kreuzproduktes ($BGP(G_c)$) und zweitens eine abschließende räumliche Filterung ($\sigma_{geofilter}(R_4)$).

Wie aber lässt sich eine verteilte Suche mit der Sprache SPARQL realisieren? Die SPARQL-Syntax enthält genau genommen **zwei SPARQL-Ausdrucksmittel**, um innerhalb einer Anfrage mehrere Datenquellen zu adressieren:

1. *Benannte Graphen*, die kontextbezogene Graphenmuster mit dem `Graph`-Operator einleiten
2. *Dienstaufrufe* mit dem `Service`-Operator

Das Codebeispiel 23, das der Dokumentation der Software *Distributed Sail* entnommen wurde, gibt einen Einblick, wie mit dem `Graph`-Operator der Graphenkontext, identifiziert über die Grahen-URI, vorgegeben wird. Im Beispiel dienen die beiden `Graph`-Anweisungen in den Zeilen 6 und 11 dazu, Publikationen (`?paper`, Zeile 7/8) und Mitverfasser (`?coauthor`, Zeile 8/9) in der wissenschaftlichen Bibliothek *DBLP*[164] ausfindig zu machen und gemeinsam mit den Geburtstagsinformationen der Mitverfasser zu präsentieren (`?birthdate`, Zeile 13), die aus der Wikipedia bzw. DBPedia stammen. Nebenbei weisen die Zeilen 2 und 3 an, keine Daten des Default-Graphen, sondern lediglich jene der beiden benannten Graphen abzurufen. Zwar ist die `Graph`-Notation ursprünglich dazu gedacht, mehrere Grahenkontexte in einer physikalischen Wissensbasis (RDF-Triplestore) zu unterscheiden, es ist jedoch der Implementierung eines Federators überlassen, das Unterscheidungsmerkmal des Graphenkontextes zu nutzen, um entfernte SPARQL-Endpoints anzufragen. Die neuere SPARQL-Spezifikation Version 1.1 [Williams 2012] (noch im Entwurfsstadium; Stand: Januar 2012) führt zwecks Aufruf entfernter Dienste ein dediziertes Ausdrucksmittel ein: den `Service`-Operator. Wollte man diesen Operator verwenden und das Codebeispiel 23 dahingehend anpassen, so entfielen die Zeilen 2 und 3 und anstelle des `Graph`-Operators (Zeile 6 und 11) wäre der `Service`-Operator einzusetzen.

```
1  SELECT ?coauthor ?birthdate
2  FROM NAMED <http://www4.wiwiss.fu-berlin.de/dblp/>
3  FROM NAMED <http://www.dbpedia.org>
4  WHERE
5  {
6    GRAPH <http://www4.wiwiss.fu-berlin.de/dblp/> {
7      ?paper dc:creator <http://www4.wiwiss.fu-berlin.de/dblp/resource/person/100007>.
8      ?paper dc:creator ?coauthor.
9      ?coauthor foaf:name ?name.
10   }
11   GRAPH <http://www.dbpedia.org> {
12     ?person foaf:name ?name.
13     ?person dbpedia:birth ?birthdate.
14   }
15 }
```

Codebeispiel 23 – SPARQL-Anfrage zur verteilten Suche[165]

[164] Dokumentation: http://www.informatik.uni-trier.de/~ley/db/

Beide Ausdrucksmittel, der `Graph`- sowie der `Service`-Operator, sind in der SPARQL-Spezifikation leider dazu vorgesehen, die elementare Abfrage mittels SPARQL `Select`-Anfragen durchzuführen. Dadurch können keine Graphen, sondern nur RDF-Relationen in den gemeinsamen Fragekontext zurückliefert werden, so dass der Benutzer bei Verwendung von `Graph`- und `Service`-Operatoren auf die Auswertestrategie a) festgelegt ist. Im Umkehrschluss erlauben also beide Ausdrucksmittel keine räumlichen Filterungen zur Laufzeit, weil zur Bildung von Kreuzprodukten ein einheitlicher Anfragegraph fehlt (vgl. G_c) in Abbildung 5.21). Die Testläufe mit den erwähnten Federationswerkzeugen *Federation Sail* und *Distributed Sail* waren dementsprechend ernüchternd und für das behandelte Anwendungsszenario unbrauchbar. Das Federation Sail, das scheinbar keine Weiterentwicklung erfährt, entpuppte sich als nicht funktionstüchtig. Sowohl die spärliche Dokumentation als auch der Einblick in den Quellcode verhalfen nicht dazu, das Federation Sail in Betrieb zu nehmen. Hingegen funktionierte das Distributed Sail prinzipiell schon, allerdings auf Basis von SPARQL `Select`-Anfragen, die eine vorhandene Verlinkung der Ressourcen voraussetzen (Auswertestrategie a); siehe Abbildung 5.22).

Aus Mangel an verwendungsfähigen Federatoren wurde das Anwendungsszenario schließlich *lokal* ausgeführt. Dazu wurden die LinkeGeoData-Daten, d.h. Ressourcen zu Schulen und Hostels, vorab gefiltert, als Graph exportiert und in das temporäre Repository des OWS-Proxy eingeladen (vgl. Datenquelle D in Abbildung 5.13). Dadurch waren sowohl INSPIRE-Schutzgebietedaten als auch LinkedGeoData-Daten zum Zeitpunkt der SPARQL-Abfrage auf das temporäre Repository verfügbar und konnten miteinander verschnitten werden (Prozessschritt 6 im Workflow; Abbildung 5.9). Die diesbezügliche SPARQL-Anfrage und die Darstellung des Ergebnisses ist in Anhang E aufgenommen. Damit konnte die Durchführbarkeit des Anwendungsszenarios in einem lokalen Anwendungskontext unter Beweis gestellt werden.

Letztlich ließen aber auch inhaltliche - nicht nur technische - Hürden die Schlussfolgerung zu, dass eine lokale Auswertung einer allumfassenden Anfrage an einen Federator vorgezogen werden sollte. In Bezug auf LinkedGeoData, deren Datenbestand aus dem Crowd Sourcing-Projekt OpenStreetMap hervorgeht, tauchten dabei Fragen zur Datenqualität auf. Zum einen sind die zahlreichen und teils sehr ähnlichen Ressourcenklassifizierungen von LinkedGeoData hinderlich, um eine Identifizierung der benötigten Daten vorzunehmen, z.B. die gleichrangigen Ressourcenklassen `Natural`, `NationalPark`, `ProtectedArea` oder `NatureReserve`. Andererseits ist die Vollständigkeit der gemeinnützig erfassten Daten oder deren Sachattribute schwerlich nachzuvollziehen, was in mancher Hinsicht sicherlich für behördliche Daten spricht, deren Erhebung einem gesetzlichen Auftrag folgt.

Die Diskussion zur Anwendbarkeit der Anwendungsszenarien wird im nächsten Kapitel 6 fortgesetzt, dort mit dem Hauptaugenmerk auf den anfangs aufgestellten Nutzeranforderungen (Kapitel 3), die zu einem Vergleich mit Implementierungsaspekten und den gewonnenen Testerfahrungen herangezogen werden. Rückblickend haben sich alle Testszenarien als praktizierbar herausgestellt. Die zuletzt aufgedeckten und theoretisch untermauerten Schwierigkeiten hinsichtlich einer verteilten Datensuche sind vor allem den hier offensichtlich werdenden begrenzten Ausdrucksmitteln der Anfragesprache SPARQL geschuldet. Diese Defizite konnten hier zwar identifiziert werden, sie berühren aber nicht das primär behandelte Arbeitsthema, das darin besteht, ein tragfähiges Konzept für die Semantic Web-Erschließung von INSPIRE-Daten zu entwickeln. Dessen erfolgreiche Umsetzung und Realisierung in einem Prototypen belegen insbesondere die Ergebnisse der ersten drei Testszenarien.

[165] Quelle: Uni Koblenz-Landau: http://www.uni-koblenz-landau.de/koblenz/fb4/AGStaab/Research/systeme/DistributedSPARQL; die Präfix-Deklarationen wurden zur kürzeren Darstellung ausgespart

6 Ergebnis und Diskussion

Die Arbeitsresultate sollen nun der Reihe nach aus Anwendersicht und aus Sicht eines Diensteadministrators und Datenbereitstellers beurteilt werden. Der erste Abschnitt 6.1 greift dazu die Nutzeranforderungen aus Kapitel 3 auf und kontrolliert aus Anwendersicht, inwieweit die aufgestellten Konzepte ihren Zweck erfüllen. Die Anforderungen eines Datenbereitstellers sind hingegen eng verknüpft mit den eingangs formulierten Hypothesen, weshalb Letztere in Abschnitt 6.2 geprüft und bewertet werden. Ausgehend von den Hypothesen, die Potentiale der INSPIRE-Infrastruktur betonen, soll auch die Kehrseite von INSPIRE betrachtet werden, die in zahlreichen technischen und inhaltlichen Hürden zum Ausdruck kommt. Zum Abschluss des Diskussionskapitels beschäftigt sich der Abschnitt 6.3 mit der Übertragbarkeit der hier erarbeiteten Lösungen und zeigt die Chancen ihrer Verwendung auf.

6.1 Nutzeranforderungen

Die beiden zentralen Nutzeranforderungen funktionaler Art sind im Anforderungskatalog in Kapitel 3 an erster Stelle genannt. Sie beziehen sich auf den **Datenzugriff** (*FA1*) und einer gleichzeitig stattfindenden **Datenselektion**, um die Datenmenge auf die gewünschten Objekte vorfiltern und damit den Datentransfer begrenzen zu können (*FA2*). Die Aufgaben sind von technischen Datenschnittstellen zu leisten und werden, wie in der INSPIRE-Datenbereitstellung vorgesehen, bereits durch INSPIRE-Downloaddienste in der Ausprägung eines OGC WFS-Dienstes abgedeckt. Es ist jedoch fraglich, ob das INSPIRE-Datenangebot tatsächlich über die OGC-Webdienste wahrgenommen und einem großen Benutzerkreis zugeführt werden kann, denn bislang beschränkte sich ihre Nutzung vornehmlich auf Geschäftsprozesse der Länder- und Bundesverwaltungen. Das Semantic Web bietet sicherlich eine breitere Benutzerplattform, weshalb die zusätzliche Verfügbarmachung von INSPIRE-Inhalten im Semantic Web trotz dadurch erzielter redundanter INSPIRE-Zugriffsmethoden durchaus sinnvoll ist. Der OWS-Proxy adaptiert in der Funktion eines SPARQL-Endpoints die Fähigkeiten der Downloaddienste und erlaubt den Zugriff auf INSPIRE-Daten sowie vergleichbarer Filteroptionen. Dabei unterstützt der OWS-Proxy sämtliche der in Anforderung *FA2* aufgeführten Selektionsmethoden, von textuellen bis hin zu geographischen Attributfilterungen. Implizit erfüllen sowohl die INSPIRE-Downloaddienste als auch SPARQL-Endpoints die Nutzeranforderung *FA3*, indem sie als webbasierte Datenzugriffsdienste je nach Informationsbedarf beliebig angefragt und in Rechercheoberflächen eingebunden werden können.

Die spezifischen Vorzüge einer INSPIRE-Datenbereitstellung im Semantic Web werden tatsächlich erst mit Blick auf die Nutzeranforderungen *FA4* bis *FA7* ersichtlich. Ein grundlegendes Prinzip des Semantic Web und im Speziellen der Linked Open Data-Initiative ist die weitestmögliche **Informationsvernetzung**, die auf einfachen Web-Mechanismen, wie der Verlinkung und Nachverfolgung von Inhalten mittels auflösbarer HTTP-Ressourcenadressen, basiert. Dieser Maxime folgend bildet der OWS-Proxy INSPIRE-Objektreferenzierungen verlustfrei in Ressourcen-verknüpfende ontologische Aussagen ab. INSPIRE-Konzepte und -Objektinstanzen erhalten eine eindeutige und auflösbare HTTP-URI, um Verlinkungen zwischen INSPIRE-Ressourcen und Ressourcen externer Datenquellen zu fördern (sogenannte *Links* und *Backlinks*). Darüber hinaus zeigt das Konzept des OWS-Proxy Mittel und Wege auf, die Rechercheergebnisse mit Referenzinformationen, z.B. mit Definitionen zu INSPIRE-Registereinträgen (SKOS-Inhalte) oder INSPIRE-Konzepten (OWL-Annotationen), anzureichern. All diese Ansätze führen zu einer besseren Informationsvernetzung und kommen dadurch der Nutzeranforderung *FA5* nach.

Sind Datenquellen über Ressourcenverlinkungen miteinander vernetzt, eignen sich Semantic Web-Anfragen mit der Sprache SPARQL zur Verschneidung und systemübergreifenden **Integration von Datenquellen** (*FA4*). Fehlen jedoch die Ressourcenverlinkungen, ist zumindest eine verteilte Suche über Online-Datenquellen problematisch und ohne eine integrative Komponente nicht zu bewerkstelligen, was die Testszenarien III und IV ergeben haben. Der OWS-Proxy ist eine integrative Komponente, die speziell auf INSPIRE-Downloaddienste ausgelegt ist, um Bestandteile einer SPARQL-Anfrage auf ausgewählte INSPIRE-Dienstaufrufe zu verteilen und Ergebnisse unabhängig von ihrer Diensteherkunft miteinander zu verschneiden. Darüber hinaus berücksichtigt das Proxy-Konzept eine **Datenintegration auf schematischer Ebene**. Mit dem Einsatz der OWL zur Modellierung der INSPIRE-Themenontologien sind die Grundvoraussetzungen für ein umfassendes Ontologie-Alignment der INSPIRE-Konzeptwelt mit externen Upper- und Domänenontologien geschaffen, die perspektivisch gesehen interessante Anwendungsmöglichkeiten eröffnen. Die Verknüpfung von Konzeptwelten dient einerseits der Rechercheunterstützung in Suchoberflächen, um dort z.B. Synonyme kenntlich zu machen oder Taxonomien als Orientierungshilfe zu visualisieren (siehe *FA7*). Datendienste können ihrerseits Query-Rewriting-Methoden anwenden und die Anfrageauswertung mit Inferenzenbildungen koppeln. Das Ontologie-Alignment erleichtert zudem die Ausführung von Datentransformationen, die im Zuge einer Datenweitergabe und diesbezüglicher Integrationsaufgaben anfallen können. Wichtig ist in diesem Zusammenhang die Verwendung standardisierter Transferformate und Übertragungsprotokolle, um eine **syntaktische Interoperabilität** von Datenexporten zu gewährleisten. Das vom W3C-geprägte Semantic Web harmonisiert den Datentransfer über die maßgebende RDF-Syntax, auf der alle Ontologiesprachen wie OWL oder Datenschnittstellen wie SPARQL- und Linked Data-Endpoints basieren. Der OWS-Proxy macht sich die Semantic Web-Normierungen zunutze und berücksichtigt damit auch die funktionale Nutzeranforderung *FA6* nach geeigneten Reproduktionsmechanismen und Transferformaten.

Die **nicht-funktionalen Anforderungen** betreffen überwiegend Aspekte, die nicht oder nur zu einem geringen Ausmaß durch eine Proxy-Realisierung beeinflusst werden können. Mit den Forderungen nach Datenkonsistenz und Aktualität (*NFA1*), der Reproduzierbarkeit von Ergebnissen (*NFA4*) sowie der Vermeidung von Redundanzen (*NFA6*) werden notwendige Bedingungen für einen qualitativen Datenabruf angesprochen, deren Einhaltung allerdings vornehmlich von den Betreibern der INSPIRE-Downloaddienste zu gewährleisten ist. Schließlich ist die Proxy-Realisierung auf aussagekräftige und konsistente INSPIRE-Ausgangsdaten angewiesen, welche zwar in OWL umformatiert und geringfügig annotiert, jedoch ohne große inhaltliche Veränderungen an einen SPARQL-Client ausgegeben werden. Weitere Faktoren sind gleichermaßen stark von den äußeren Randbedingungen abhängig, wie z.B. die Diensteverfügbarkeit (*NFA2*) und die Performanz von Suchanfragen (*NFA3*). Hinsichtlich der Performanz sollte mit einer effizienten Systementwicklung wenigstens sichergestellt sein, dass die durch Auswerteprozesse im OWS-Proxy hervorgerufenen Verzögerungen möglichst gering ausfallen, was nicht zuletzt mittels der hier vorgeschlagenen sehr performanten Art der GML-Ergebnistransformation erzielt werden kann (siehe Abschnitt 5.2.4). Die Nutzeranforderungen *NF5* (*einfacher Umgang mit Ressourcen*) und *NF6* (*harmonisierte Datenwelten ohne Redundanz*) berühren Aspekte, die sich auf die Benutzerfreundlichkeit und dadurch indirekt auf die Akzeptanz beim Anwender auswirken. Da die Themenontologien mit ihrer Datenstruktur die wesentliche Anwendungsgrundlage bilden, galt es insbesondere bei ihrer Modellierung auf Aspekte der Benutzerfreundlichkeit zu achten. Beispielsweise sind die Themenontologien so angelegt, dass die Modellnähe zu den INSPIRE-Datenspezifikationen erhalten bleibt, wodurch u.a. eine bessere Orientierung in den INSPIRE-Datenmodellen ermöglicht werden soll. Zugleich sorgen Vorkehrungen in den Datentypen aus numerischen, klassifizierenden und geographischen Informationskategorien dafür, dass Semantic Web-Anfragen benutzerfreundlich und transparent durchgeführt werden können. Desweiteren werden Harmonisierungen herbeigeführt, die auch im Geo Web sinnvoll erscheinen, dort aber leider fehlen, wie z.B. einheitliche Speicherformen und -listen für Kodierungen und deren eindeutige Identifizierung (z.B. für CRSs und Maßeinheiten). Alles in allem wurden also die Gestaltungsfreiräume im Spannungsfeld zahlreicher Spezifikationen aus dem Semantic- und dem Geo Web sowie dem INSPIRE-Umfeld und allen davon abstammenden technischen Randbedingungen ausgiebig genutzt und den beschriebenen funktionalen wie nicht-funktionalen Nutzeranforderungen in sehr vielen Belangen nachgekommen.

6.2 Evaluation der Hypothesen, Semantic Web-Potentiale und Hürden

In dieser Arbeit werden Gemeinsamkeiten des Semantic Web und des Geo Web untersucht und das Ziel des interoperablen Datenaustausches zwischen beiden Technikdomänen verfolgt. Dabei wird der Frage nachgegangen, ob und auf welche Weise sich INSPIRE-Geodaten, die originär im Geo Web beheimatet sind und zukünftig über GDI-Dienste bereitstehen, in Formate überführen und mit Methoden des Semantic Web analysieren lassen. Die **zentrale Hypothese** der Arbeit lautet:

> *Es treffen genügend begünstigende Faktoren - ausgelöst durch etablierte Standards und dem allgemein fortgeschrittenen Stand in Wissenschaft, Technik und Anwendung - zusammen, um INSPIRE-Geodaten effizient und interoperabel im Semantic Web nutzbar zu machen.*

Die Hypothese und auch das Arbeitsthema konzentrieren sich bewusst nur auf die INSPIRE-Datenschicht, gewissermaßem dem Herzstück der INSPIRE-GDI, die aus den Rohdaten, sprich den Geometrie- und Sachdaten, besteht. Mit der Datenverwendung und effektiven Zugriffsmechanismen steht und fällt der praktische Nutzen einer Semantic Web-Erschließung von GDI-Strukturen, die allein durch Maßnahmen der Metadatenübertragung oder GDI-Datenzugriffen ohne Filterqualitäten nicht zufriedenstellend gelöst werden kann. Folglich ist die Datenschicht prioritär zu behandeln und der enge Fokus darauf zwingend erforderlich, um angesichts komplexer INSPIRE-Datenmodelle und -Umsetzungskriterien zu fundierten Schlussfolgerungen und anwendbaren Lösungen zu gelangen.

Die Arbeitsergebnisse bestätigen die obige Hypothese von der theoretischen Seite anhand der Konzepte zu den Themenontologien und der Funktionsweise des OWS-Proxy und beweisen zudem die Praktikabilität der Konzepte durch die Umsetzung und die erfolgreichen Testläufe eines Prototypen. Die begünstigenden Faktoren stecken vor allem in diversen **Standardisierungen** beider Technikdomänen, die eine jeweils hohe Verbreitung und Entwicklungsreife erlangt haben. Sie basieren auf gemeinsamen Basistechnologien, wie die für Transferformate relevante und allgegenwärtige Auszeichnungssprache XML und die zur XML-Sprachfamilie gehörenden Analysetechniken zur Inhaltstransformation (XSLT), -Selektion und -Abfrage (XPath und XQuery) und der Informationsvernetzung per XLink. Ebenso harmonisierend wirkt die Verwendung des Internetprotokolls HTTP, das Zugriffe auf Webdienste und entfernte Ressourcen stark vereinheitlicht. Darauf bauen die mittlerweile ausgereiften Semantic Web-Empfehlungen des W3C auf, angefangen bei den ursprünglichen Normierungen des Daten-Layers mit RDF (2004) und OWL 1.0 (2004), der überarbeiteten OWL in der Version 2.0 (2009) und den parallelen Entwicklungen der Anfragesprache SPARQL 1.0 (2008) und ihrer noch unfertigen Weiterentwicklung SPARQL 1.1 (*Working Draft*; Stand: Januar 2012). Die Kontinuität und Popularität der Spezifikationen wird begleitet von einer Fülle an zugänglichen Datentöpfen, die z.B. Linked Data-Prinzipien oder SKOS-Kriterien genügen und wegführen von der anfänglichen modelllastigen Praxis hin zu interessanten Informationssammlungen. Das Geo Web blickt auf eine vergleichbare Entwicklungsgeschichte zurück, die zuletzt darin mündete, dass eine Reihe von OGC-Spezifikationen aktualisiert und als ISO-Standardisierungen übernommen wurden. Darunter befinden sich alle für die INSPIRE-Datenbereitstellung relevanten OGC-Spezifikationen: zum Datentransfer GML 3.2.1 (2007; Review: 2010 abgeschlossen) und für die Datenschnittstelle WFS 2.0 und FE 2.0 (2010).

Die beidseitig hohen Standardisierungsgrade und die Verwandtschaft der Spezifikationen untereinander bilden das Rückrat für eine **syntaktische Interoperabilität**. Betrachtet man die **Datenformate**, so ist die Kompabilität zwischen RDF und GML auffallend hoch, was strukturell bedingt ist durch die anfängliche RDF-Realiserung der damaligen GML-Version 1.0 und sich am deutlichsten im GML Object-Property-Muster widerspiegelt. Aber auch verwandte Mechanismen der Objektreferenzierung bzw. -verlinkung, die XML-Formatierung von GML und RDF/XML sowie der Gebrauch vieler primitiver XSD-Datentypen in GML wie auch in RDF und OWL erleichtern die syntaktische Umformung in die ein oder andere Richtung. Mit der beim OGC betriebenen GeoSPARQL-Spezifizierung leistet die Geo Web-Gemeinde sogar einen Know-how-Transfer, der auf die Einführung fortschrittlicher Geodatenverarbeitungsstrategien und Geometriespeicherungen im Semantic Web abzielt. Umgekehrt übernimmt das Geo Web vorteilhafte

Semantic Web-Methodiken, indem seit kurzem z.B. OGC-Spezifikationselemente und -konzepte mit auflösbaren HTTP-URIs beschrieben, Dienstprofile für das Semantic Web-Enablement entworfen und in Interoperabilitätsexperimenten und anderen Bemühungen Synergien untersucht werden. Das fehlende Puzzlestück der syntaktischen Interoperabilität zwischen RDF/OWL und GML sind möglicherweise die Abbildungsregeln. Bisherige praktische Ansätze beschränken sich auf die Umformung von Datenmodellen (XSD → OWL) und lassen dabei die Instanzdaten außer Acht oder formen sie nur rudimentär um, so dass komplexere Datenthemen, wie die in INSPIRE transferierten Geodatensätze, unberücksichtigt bleiben. Das hier erarbeitete Modellierungskonzept schließt die Lücke, indem es erstens Regeln für eine (teil-) automatisierte Modellabbildung von UML-Klassenmodellen auf OWL-Ontologien einführt und zweitens eine Strategie vorgibt, wie Abbildungsregeln für Instanzdaten basierend auf XPath-Anweisungen direkt in OWL-Ontologien zu integrieren sind. Das wiederum erlaubt einen lückenlosen und transparenten Semantic Web-Zugang zu INSPIRE-Datenmodellen und -Geodaten. Der Ausblick darauf wird leider durch einige wenige **inhaltliche Hindernisse** getrübt, die der Semantic Web-Tauglichkeit von INSPIRE-Daten im Wege stehen. Dabei zeichnet sich ein großes Problem in den Datenspezifikationen zu den Themen aus Annex-II/III ab, deren Feature Types nicht mehr mit verbindlichen, sondern nur noch mit optionalen INSPIRE-Identifikationselementen (*INSPIRE-Ids*) modelliert sind. Ohne eindeutige INSPIRE-Identifikatoren ist allerdings eine Generierung persistenter und auflösbarer HTTP-URIs ausgeschlossen. Ebenso schwindet die Aussicht auf eine Informationsvernetzung der INSPIRE-Ressourcen mit denen externer Linked Data-Datenquellen. Doch auch mit eindeutigen Identifikatoren ist eine Informationsvernetzung aufwendig herbeizuführen und ihr Management schwierig, sieht man der Tatsache ins Auge, dass INSPIRE-Objektinstanzen womöglich mit Anpassungen fortgeführt oder die INSPIRE-Datenmodelle weiterentwickelt werden könnten, deren fortschreitende Versionierung sich u.a. auf die Themenontologien und Ressourcenverlinkungen auswirken würde.

INSPIRE forciert die **syntaktische Interoperabilität über Webdienste**, so dass INSPIRE-Geodaten über die Schnittstelle der INSPIRE-Downloaddienste abgerufen und - mit der Unterstützung einer vermittelnden Komponente - direkt für Semantic Web-Analysen herangezogen werden können. Das Konzept des OWS-Proxy definiert hierzu einzelne Auswerteschritte, u.a. eine Sprachabbildung von SPARQL auf WFS GetFeature-Anfragen und die Ergebnistransformation von GML nach RDF/OWL. Zusammengefügt zu einem Anfrage-Workflow ermöglichen sie die ganzheitliche und webbasierte SPARQL-Auswertung von INSPIRE-Geodaten über den verteilten Aufruf von INSPIRE-Downloaddiensten. Die Funktionsweise eines Proxys, der an beliebiger Stelle administriert und eventuell mehrfach, jeweils bezogen auf ausgewählte Themen eingerichtet werden kann, trägt entscheidend zur Entlastung der INSPIRE-Datenhalter bei. Denn sie können allein durch den Betrieb von INSPIRE-Downloaddiensten und ohne zusätzliche Administrationsaufwände und technische Vorkehrungen leisten zu müssen dazu beitragen, die Informationsvielfalt des Semantic Web zu steigern. Somit stimmen abgesehen von der technischen Machbarkeit grundsätzlich auch die organisatorischen Rahmenbedingungen. Nichtsdestotrotz sind insbesondere die INSPIRE-Downloaddienste mit einigen Fallstricken behaftet, die eine effiziente Recherche und Übertragung von INSPIRE-Daten verhindern können. Zum einen sind anstelle der höherwertigen WFS-Dienste auch Downloaddienste mit fertigen Datenpacketen zulässig (*for pre-defined datasets*), die in der Funktion von HTTP- oder FTP-Servern Filterfunktionalitäten vermissen lassen und denjenigen INSPIRE-Datenhaltern zugedacht sind, für die das Betreiben eines WFS-Dienstes zu aufwendig wäre. Es bleibt zu hoffen, dass die davon betroffenen INSPIRE-Datensätze von größeren Institutionen bzw. zentralen Rechenzentren gehostet und auf diesem Wege über WFS-Schnittstellen zugänglich gemacht werden. Zum anderen ist auch der Umgang mit den WFS-Diensten selbst recht ineffizient, insofern nämlich, dass stets die Geometriedaten in der Ergebnismenge enthalten sind, egal ob sie in einer Anfrage explizit verlangt wurden oder nicht. Darüber hinaus können Koordinatentransformationen oder -umformungen während eines Datenabrufes anfallen, falls Geometrien in einem anderen als dem zur Speicherung verwendeten CRS nachgefragt werden. INSPIRE gibt zwar die verpflichtend zu unterstützenden CRSs vor, legt allerdings nicht fest, in welchem CRS die Geometrien originär zu speichern sind. Somit werden viele WFS-Datenanfragen eine längere Bearbeitungszeit benötigen und die allgemeine Leistungsfähigkeit der INSPIRE-GDI verringern. Ein weiteres Manko ist die mangelnde Transparenz bei WFS-Datenanfragen, die mit dem Auflösen von Objektreferenzierungen einhergeht und nur bedingt mit dem Steuerungsparameter `traverseXlinkDepth`

beeinflusst werden kann. Wie in der Arbeit ausgeführt, erschwert die Ungewissheit darüber, an welchen Stellen die INSPIRE-Daten überall Objektreferenzierungen enthalten, eine Anfrageplanung bzw. automatisierte Verarbeitung der WFS-Ergebnisse (vgl. Abschnitt 5.2.3).

Zusammenfassend lassen sich also einige größere und kleinere Hürden ausmachen, die den effizienten und interoperablen Datenabruf begrenzen. Es ist jedoch ausdrücklich darauf hinzuweisen, dass die erwähnten Schwierigkeiten, hervorgerufen durch unnötig tolerante INSPIRE-Projektfestlegungen und zugleich zu restriktive WFS-Spezifizierungen, sich letztlich auf den allgemeinen Nutzen der INSPIRE-GDI auswirken und keinesfalls auf die hier untersuchte Nachnutzung im Semantic Web beschränkt sind. Gleichsam verhält es sich mit der räumlichen Datenanalyse über Dienstegrenzen hinweg, die sich im Testszenario IV als problematisch erweist. Der dortige Versuchsaufbau strebt die verteilte Suche bei gleichzeitiger Anwendung räumlicher Verschneidungsoperationen an, die im *herkömmlichen* Geo Web nur über Anfragen an einen kaskadierenden, d.h. externe Datenquellen integrierenden WFS-Dienst durchgeführt werden könnte. Jedoch auch nur unter der Prämisse, dass alle Anfrage-relevanten Datenquellen in dem kaskadierenden WFS-Dienst vereint, sprich durch Vorkonfiguration integriert wurden, was wiederum eine freie Themenauswahl und das dynamische Einbeziehen frei verfügbarer Datenquellen ausschließt. Letztlich stößt damit das Konzept und die Realisierung des Prototypen an **Interoperabilitätsgrenzen**, die auch die Geodatenverarbeitung im Geo Web charakterisieren und für die sowohl im Geo Web wie auch im Semantic Web noch keine passgenauen, geschweigedenn eleganten Systemlösungen vorhanden sind.

Zuletzt sei das Augenmerk auf die **semantische Interoperabilität** bei der Übertragung von INSPIRE-Daten in das Semantic Web gerichtet. Die Semantik korrekt zu transportieren und keine begrifflichen Ungenauigkeiten - außer jenen in den INSPIRE-Datenspezifikationen bereits vorhandenen - zuzulassen, ist ein großes Anliegen dieser Arbeit. Damit zusammen hing eine grundlegende Entscheidung für die Ausarbeitung des Modellierungskonzeptes. Gemeint ist die Wahl, entweder auf existierende und etablierte Semantic Web-Vokabulare zurückzugreifen und damit die INSPIRE-Begrifflichkeiten auszudrücken oder aber gänzlich neue INSPIRE-Vokabulare zu schaffen, die eine neue Begriffswelt im Semantic Web eröffnen. Aus mehreren Gründen fiel die Wahl auf die zweite Option, das Erstellen neuer Vokabulare in Form der Themenontologien. Einerseits, um langwierige Konzeptstudien zu vermeiden, die ohnehin besser von Fachexperten durchgeführt und diskutiert werden. Desweiteren wäre im Gegensatz zur jetzigen Übertragbarkeit des Modellierungskonzeptes für jedes INSPIRE Annex-Thema eine neuerliche Untersuchung erforderlich, und zwar mit ungewissem Ausgang auf Erfolg in der praktischen Umsetzung. Vielmehr erlaubt die Einführung klar getrennter Konzeptwelten, zunächst den notwendigen Datenzugriff zu offerieren, wie hier anhand der Zugriffsmechanismen des OWS-Proxy entworfen und praktiziert, und erst in einem zweiten Schritt und mit den dedizierten Semantic Web-Mitteln innerhalb eines Ontologie-Alignments die Verknüpfung zu beliebigen Upper- und Domänenontologien zu suchen. Bei dieser Vorgehensweise werden die originären INSPIRE-Begriffe und Bedeutungen souverän in das Semantic Web überführt, d.h. sie werden mit Identifikatoren, Definitionen und kontextbezogenen Verlinkungen versehen und somit als Auswerteobjekte greifbar bzw. maschinell lesbar. Die semantische Interoperabilität ist also dank der direkten Übersetzung der Datenmodellstrukturen und der wortgleichen Abbildung auf Ontologiekonzepte implizit erreicht. Die in diesem Zusammenhang aufgestellte Hypothese lautet:

> *Die Überführung von INSPIRE-Datenmodellen in semantische Ontologien ist lohnenswert und unkompliziert, weil jede INSPIRE-Datenspezifikation ein bestimmtes Fachgebiet abdeckt und somit Qualitäten einer Domänenontologie aufweist.*

Anstatt problemorientiert und bestimmten Aufgabenstellungen zugedacht zu sein, stellen die INSPIRE-Datenmodelle - soweit es die Abstimmungsprozesse zwischen den INSPIRE-Beteiligten gestatten - eine objektive Sicht auf komplexe Fachgebiete und ihre Begrifflichkeiten dar. Damit erfüllen sie die Bedingungen eines gemeinschaftlich verwendeten Vokabulars bzw. einer *Domänenontologie*. Die obigen Ausführungen weisen zudem darauf hin, dass eine direkte Übersetzung und Etablierung von INSPIRE-Themenontologien viele Vorteile bieten und leicht unter Anwendung der aufgezeigten, neutralen Modellierungsregeln erzielt werden können. Diese Gesichtspunkte berücksichtigend wurde nicht

zuletzt auch die thematische Strukturierung der INSPIRE-Datenmodelle auf die INSPIRE-Themenontologien übertragen, so dass jedes INSPIRE Annex-Thema separat in einer eigenen Domänenontologie behandelt wird.

Fazit

Der initiale Aufwand, der zur Erschließung der INSPIRE-Datenwelt im Semantic Web geleistet werden muss, scheint recht hoch angesichts der zahlreichen Hürden in der INSPIRE-Datenbereitstellung, den vielfältigen Konzepten dieser Arbeit und den diversen technischen Komponenten, die es zur Umsetzung in Betrieb zu nehmen gilt, wie z.B. ein oder mehrere als Broker agierende Proxy-Anwendungen oder Webdienste zwecks Publikation der Themenontologien und der Referenzinformationen. Mit Einrichten einer derartigen Infrastruktur stehen jedoch im Gegenzug sehr innovative Semantic Web-Techniken und Linked Data-Praktiken zur Verfügung, denen sich die vorgeschlagene Systemlösung nicht im Sinne eines minimalistischen, restriktiven Ansatzes verschließt, sondern sie in ihrer offenen Gestaltung entweder direkt unterstützt oder die Fähigkeit zur inhaltlichen Erweiterung aufweist. Dadurch ist ein semantischer Zugriff auf INSPIRE-Daten zu realisieren, der über den reinen Datenabruf hinausreicht und verschiedenste Nachnutzungen fernab der limitierenden INSPIRE-Anwendungsfälle in Aussicht stellt.

6.3 Übertragbarkeit auf weitere Geo Web-Anwendungsfelder

Zu guter Letzt soll die Übertragbarkeit der gefundenen Lösungen in Augenschein genommen und damit ihre Wiederverwendbarkeit im Kontext anderer Geo Web-Projekte abgeschätzt werden. Was INSPIRE anbelangt, wurde die **Übertragbarkeit der ontologischen Modellierungregeln**, die aus Versuchen mit dem Datenmodell *Protected Sites* gewonnen und abstrahiert werden konnten, anhand der zwei Annex-Themen *Administrative Units* und *Biogeographical Regions* sichergestellt. So wie der Untertitel der Arbeit andeutet (*Verteilte Suche und Publikation von GML-Daten im Semantic Web am Beispiel INSPIRE*), zielt das Arbeitsthema sogar darauf ab, nicht nur für die INSPIRE-Nutzung, sondern auch für andere GML-Datenprojekte gleichermaßen Methoden der Semantic Web-Recherche und -Publikation zu erkunden. Dieser vermeintlich hohe Anspruch relativiert sich dadurch, dass GML - u.a. als Standardausgabeformat von WFS- und Sensorwebdiensten - die Rolle des primären Geo Web-Transferformates zugedacht ist und bislang in zahlreichen Anwendungsschemata von Fachdomänen (z.B. O&M oder WaterML 2.0) und applikationsspezifischen Einzellösungen zur Anwendung kommt. Zugleich machen die INSPIRE GML-Anwendungsschemata einen sehr intensiven Gebrauch der GML-Sprachmittel, z.B. unter Verwendung einer großen Bandbreite an GML-Datentypen als auch Mechanismen wie die GML-Objektrefenzierung und -Objektvererbung, aufgrund der Einführung verzweigter anstelle von flachen Objekt-Attributlisten und nicht zuletzt wegen der fortschrittlichen Modellierung mit UML-Werkzeugen. Es ist also davon auszugehen, dass andere Geo Web-Projekte mit Blick auf INSPIRE vergleichbare oder weniger umfängliche GML-Techniken verwenden und ihre zugehörigen UML- bzw. GML-Schemata eine weitestgehend analoge Modellabbildung auf OWL-Ontologien erfahren können. Und schließlich sind die getrennt voneiander untersuchten Aspekte der Ontologiemodellierung auch nur in Ausnahmefällen wirklich INSPIRE-spezifisch (vgl. INSPIRE-Stereotypen, -Identifikatoren) und ansonsten frei auf alle GML-Anwendungsfelder übertragbar. Darunter zählen insbesondere die Analyse geeigneter Semantic-Web Entsprechungen für diverse GML-Datentypen, die Betrachtung eines zweckmäßigen Umgangs mit Referenzinformationen sowie des interoperablen Transfers geographischer Informationen.

Für die ebenso bedeutsame **Datentransformation** (GML-Instanzdaten → RDF/OWL-Ressourcen) führt die Arbeit zum einen eine Methodik zur Konfiguration von Abbildungsregeln ein, die als Annotationen in die OWL-Ontologien aufgenommen werden. Zum anderen wurde ein technischer Lösungsweg entworfen, um die Abbildungsregeln im Rahmen einer performanten Datentransformation basierend auf XPath-Auswerteschritten anzuwenden. Beide Lösungen haben einen allgemeingültigen Charakter. Das heißt sie sind nicht zwangsweise auf OWL-Ontologien beschränkt, die gemäß

des vorgeschlagenen Modellierungskonzeptes erstellt wurden. Sie lassen sich generell auch im Rahmen bereits existenter und aus GML-Datenmodellen entstammender OWL-Ontologien nutzen, um neben dem schematischen Wissen auch konforme GML-Instanzdaten zu überführen und effizient in das Semantic Web zu replizieren. Werden diese Gedanken konsequent zu Ende gedacht, so kann auch je nach Datenhaltung und Anforderungslage der Einsatz eines OWS-Proxy unterbleiben und die erwähnten Transformationslösungen isoliert angewendet und mit der im Lösungsansatz besprochenen Systemarchitektur A kombiniert werden (siehe Abschnitt 4.2, *Separates Semantic Web-Repository*). Entgegen der im OWS-Proxy stattfindenden Datentransformation zur Laufzeit würde in diesem alternativen Szenario eine Vorabtransformation ausgeführt, deren Ergebnisse in ein statisches und per SPARQL-Schnittstelle anfragebereites Semantic Web-Repository (RDF-Triplestore) hineinlaufen. Die Abwägungskriterien, die für die Proxy-Lösung im Kontext von INSPIRE sprechen, z.B. das Vermeiden einer redundanten Datenhaltung unbekannter Größe und damit verbundener Administrationsaufwände und Aktualisierungsmaßnahmen, treffen nicht notwendigerweise auf andere Datenprojekte und ihre webbasierte Datenbereitstellung zu. Letztere sind möglicherweise kleiner dimensioniert, kommen ohne die Filterunterstützung fremder Prozessierdienste aus oder unterliegen nur seltenen oder keinen Aktualisierungen. Letztlich bedarf es immer einer projektinternen Kosten/ Nutzenabschätzung und der entsprechenden Wahl der Mittel. Von dieser Warte aus gesehen, liefert die vorliegende Arbeit viele nützliche Ansätze, die sich unabhängig voneinander oder auch aneinandergereiht in der Funktionsweise des hier realisierten OWS-Proxy wiederverwenden lassen.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

Die INSPIRE-Richtlinie bringt in absehbarer Zukunft europaweit abgestimmte Datenbestände hervor, die über standardisierte Webdienste zugänglich gemacht werden. Ihre Nutzung beschränkt sich jedoch auf Geo Web-Dienstprotokolle und -Transferformate, die eine weiterführende, semantische Datenanalyse sowie die Verschneidung mit den im Internet frei verfügbaren Beständen behördlicher und Nutzer-generierten Daten erschwert. Das Semantic Web ist aufgrund seiner semantischen Auswertetechniken prädestiniert für derartige Integrationsaufgaben und enger mit der heutigen Welt des gemeinnützigen Internets verzahnt, weshalb das Semantic Web eine ideale technische Plattform zur breiteren Nutzung von INSPIRE-Daten darstellt. Die vorliegende Arbeit liefert einen gewichtigen Beitrag, den interoperablen Datenaustausch zwischen dem Semantic und dem Geo Web zu fördern und konzentriert sich dabei auf den Zugriff und die gleichzeitige Filterung von INSPIRE-Daten und ähnlicher Geo Web-Inhalte mit originären Mitteln des Semantic Web. Die Resultate der Arbeit sind hier noch einmal zusammengefasst:

- Die anfängliche Untersuchung weist auf viele Gemeinsamkeiten beider Technikdomänen hin, die einen interoperablen Datenaustausch auf hohem Niveau versprechen, zeigt jedoch auch mögliche projektbedingte Fallstricke auf. Die jeweiligen strukturellen Defizite lassen sich ausgleichen, indem das Semantic Web professionelle Geodatenverarbeitungsstrategien adaptiert und das Geo Web seinerseits die einfachen wie effektiven Semantic Web-Prinzipien der Datenidentifikation und -verlinkung berücksichtigt.
- Das erstellte Modellierungskonzept erlaubt die Überführung von INSPIRE UML-Datenmodellen in OWL-Ontologien, mittels der die semantische Interoperabilität beim INSPIRE-Datenzugriff aus dem Semantic Web sichergestellt und zugleich eine schematische Analyse im Kontext anderer Ontologien ermöglicht wird.
- Die vorgeschlagene Systemarchitektur beinhaltet eine Proxy-Komponente, die auf INSPIRE-Downloaddiensten aufsetzt und den selektiven Datenzugriff auf INSPIRE-Inhalte mit der verbreiteten Semantic Web-Anfragesprache SPARQL unterstützt. Die zwei wesentlichen, hier entworfenen Systemlösungen der Proxy-Komponente sind a) die Sprachabbildung von SPARQL- auf WFS GetFeature-Anfragen und b) die Abbildungsregeln und die Transformation der WFS-Ergebnisse in zielführende Semantic Web-Speicherformen (RDF/OWL) konform zu den erstellten INSPIRE-Ontologien.
- Da INSPIRE-Daten ein hohen Maß an Raumbezug aufweisen, widmet sich die Arbeit im Speziellen der Geodatenspeicherung und -verarbeitung mit dem Ziel, die aus dem Geo Web bekannten GIS-Operationen im Semantic Web zu etablieren. In diesem Zusammenhang wurden die aktuellen Entwicklungen am OGC-Standardisierungsentwurf GeoSPARQL begleitet und begutachtet sowie im Rahmen des GeoSPARQL-Reviews Korrekturvorschläge eingereicht.

Die Praktikabilität der Konzepte konnte durch den Aufbau einer INSPIRE-Testplattform und eines funktionierenden Proxy-Prototypen unter Beweis gestellt werden. Diverse Testläufe dienten zur Prüfung auf Performanz, verschiedener Software-Konstellationen und Anwendungsfälle. Die daraus gewonnenen Erfahrungen sind umfassend dokumentiert und zeigen die grundsätzliche Machbarkeit der hier entwickelten Konzepte und Systemlösungen, die nicht auf die Anwendung von INSPIRE-Daten beschränkt bleiben müssen, sondern sich auch auf andere Geo Web-Infrastrukturen

und -Projekte übertragen lassen. Sofern die technischen und konzeptionellen Hürden der zugrunde liegenden Datenbereitstellung ausgeräumt sind, bietet der transparente Semantic Web-Zugriff auf die essentielle Ebene der INSPIRE-Rohdaten ein großes Nutzungspotential und zudem einen beachtlichen Mehrwert unter Anwendung semantischer Auswertetechniken.

7.2 Ausblick

Gegen Ende einer Machbarkeitsstudie und bei erfolgreichem technischem Nachweis ist es natürlich wünschenswert, dass die erarbeiteten Konzepte eine praktische Umsetzung erfahren. Dafür sind jedoch unabhängig von funktionstüchtigen und leistungsfähigen INSPIRE-Downloaddiensten richtige Rahmenbedingungen für das Zusammenspiel von Informationen im Semantic Web zu schaffen. So sind in erster Linie die hier entworfenen oder ähnlich gestaltete INSPIRE-Ontologien, verschiedenste ISO- und OGC-Basiskonzepte sowie universelle Referenzinformationen (z.B. Koordinatenreferenzsysteme und Maßeinheiten) zwecks einer Fortschreibung administrativ zu pflegen und hinsichtlich einer größeren INSPIRE-Themenabdeckung zu koordinieren. Desweiteren ist die vielschichtige Wissensbasis online zugänglich zu machen, so dass alle zugehörigen Ressourcen unter individueller URI angesprochen und aufgelöst werden können. Für die Stabilität von URIs sorgen gegebenenfalls Resolverdienste, die - wie in der Arbeit dargelegt - sich auch vorteilhaft zur Nachverfolgung von INSPIRE-Objektidentifikatoren verwenden lassen. Als weitere Infrastrukturknoten könnten sich Semantic Web-Repositories anschließen, die Verknüpfungsaussagen zu externen Ontologien beinhalten und über Webschnittstellen veröffentlichen. Es sind demzufolge eine Reihe an technischen und organisatorischen Maßnahmen zu treffen, die entweder INSPIRE-spezifisch oder projektungebunden die Voraussetzungen für die zweckmäßige Inbetriebnahme der entwickelten Systemlösungen schaffen. Der entscheidende Knackpunkt ist letztlich aber die INSPIRE-Implementierung und damit das zukünftige Datenangebot selbst. Inwieweit die INSPIRE-Umsetzung von Erfolg gekrönt sein wird, ob filterbare Datenquelldienste in ausreichender Zahl und nötiger Funktionalität oder eher vorprozessierte Datensätze feilgeboten werden, ob Dateninhalte reichhaltig verknüpft und in entsprechender Datenqualität vorliegen werden und vieles mehr steht leider noch in den Sternen und wird sich erst in den kommenden Jahren herausstellen.

Ganz zum Schluss sollen auch kurz potentielle Anschlussthemen umrissen werden. Wie in der Einleitung erläutert, nimmt die Arbeit eine thematische Abgrenzung zur komplexen Thematik der Metadaten vor, die u.a. damit begründet wird, dass INSPIRE-Datenspezifikationen viele datenharmonisierende Effekte haben, wodurch insbesondere Metainformationen zur Datenqualität an Bedeutung verlieren. Nichtsdestotrotz können aber Aussagen zur datenhaltenden Stelle oder auch zu Nutzungsbedingungen, die derzeit im INSPIRE-Kontext auf jeweils nationaler Ebene und bezogen auf den Privatgebrauch lebhaft diskutiert und schriftlich festgehalten werden, durchaus sinnvoll sein. Sollte also das Thema der INSPIRE-Metadaten aufgegriffen und eine Metadatenpublikation in Einklang mit den hier behandelten Mechanismen des Datenzugriffs gebracht werden, so besteht die Möglichkeit, sich an vorhandenen Arbeiten zu orientieren. So streben gleich mehrere Ansätze eine Semantic Web-Profilierung des OGC-Katalogdienstes an [Stock 2009], [Janowicz et al. 2010]. Darüber hinaus gibt es für Metadaten einschlägige ontologische Ausdrucksformen, z.B. das weitverbreitete Vokabular *Dublin Core* oder das neue und umfängliche *Resource Description & Access* (RDA[166]), sowie Abbildungsdefinitionen z.B. von Dublin Core-Metadaten auf ISO 19115/19-normierte Metadaten des Geo Web.[167] Im *Meer der Ideen und Möglichkeiten* ist jedoch darauf zu achten, zur automatisierten Ableitung und Publikation von INSPIRE-Metadaten einen vernünftigen Lösungsweg einzuschlagen, der auch den organisatorischen Herausforderungen gewachsen ist. Zum einen müssen die Metadaten auf einem praktikablen Level (Ontologie- oder Konzept-, Intanzenebene) im Semantic Web etabliert werden. Das erscheint allerdings gar nicht so einfach angesichts der vorliegenden

[166] Vokabular zur aktuellsten Fassung des amerikanischen Katalogisierungsregelwerkes; siehe: http://rdvocab.info/

[167] Abbildungen u.a. zu finden auf den Seiten des Projektes *Semantic Web: Development and Transmission* unter: http://www.iwi-iuk.org/cashmere/htdocs/html/newsletter/data/geoinf.shtml oder im Spezifikationsdokument des ISO-Applikationsprofils für den OGC CS-W [Voges & Senkler 2007], siehe Tabelle 6

Diskrepanzen. Denn während das Geo Web bevorzugt Geodatensätze mit Metadaten beschreibt, also Objektmengen gleicher Objekttypen, verknüpft/ annotiert das Semantic Web Metadaten eher auf der Ebene der Ontologien bzw. Wissensbasen, die üblicherweise multiple Ressourcen- bzw. Objekttypen und ihre Instanzen umfassen. Zum anderen ist natürlich eine unkomplizierte Metadatensuche zu fördern und Aspekte des Zugriffs und des Wirkungsgrades zu untersuchen. Problematisch dabei ist sicherlich die diffuse Nutzung vernetzter Inhalte im Semantic Web, wodurch kaum beeinflusst werden kann, dass beispielsweise Nutzungsbedingungen vor der eigentlichen Datenverwendung eingesehen oder nötige Hinweise zur Informationszuverlässigkeit beachtet werden.

Ein anderes, noch offenes Forschungsfeld ist das Schlussfolgern geographischer und vor allem topologischer Objekteigenschaften. Die GeoSPARQL-Spezifizierung, die in dieser Hinsicht das W3C-geprägte Semantic Web einschneidend verändern könnte, weist zur Zeit noch einige Defizite auf, die sowohl auf dem jetzigen GeoSPARQL-Entwurf als auch der allgemeinen SPARQL-Auswertelogik beruhen und u.a. die Rückwärstkompabilität zu älteren Geometriespeicherformen oder die räumliche Analyse verteilter Datenquellen betreffen. Nach Verabschiedung der GeoSPARQL-Spezifikation sollte spätestens der Praxistest die angesprochenen Problematiken stärker ins Bewusstsein der Semantic Web-Gemeinde rücken und - so bleibt zu hoffen - solide Lösungen nach sich ziehen. Sie könnten in Zusammenhang mit externen GIS-Prozessierdiensten/ geographischen Reasonern stehen, die - wie im Geo Web neuerdings auch - GIS-Operationen im Sinne des Cloud Computing auslagern würden. Es ist sogar denkbar, derartige Berechnungsaufrufe unmittelbar mit Semantic Web-Ressourcen zu verknüpfen, sie also einzubetten in eine Wissensbasis und als Querverweise auf Zusatzquellen zu gebrauchen. Hierfür qualifizeren sich auch herkömmliche OGC-Dienstaufrufe, beispielsweise zum Abruf von Kartenbildern (WMS) oder zur Geokodierung von geographischem Namensgut (WFS-Gazetteer). Ein weiteres Betätigungsfeld ist das OGC-Sensor Web, um dessen Semantic Enablement sich bereits diverse Forschungsarbeiten ranken. Sie beziehen sich auf OGC Sensor Observation Services (SOS) zur Veröffentlichung von Beobachtungsmesswerten und Sensordaten im Semantic Web. Dies geschieht jedoch bislang ohne Spezifizieren einer SPARQL-Schnittstelle, die eine freie Datenrecherche gestatten würde, sondern nur über proprietäre HTTP-Aufrufschemata, die Platzhalter für Suchkriterien enthalten, wie z.B. den Zeitpunkt, die Örtlichkeit, das beobachtete Phänomen, das Aggregationsniveau der Daten etc. [Page et al. 2009]. Hier wäre zu prüfen, ob sich SOS-Dienste für Anfragen nach Messwerten (GetObservation-Operation) auf ähnliche Weise mit einer SPARQL-Schnittstelle erweitern lassen wie WFS-Dienste über ihre GetFeature-Operation. Wie schon mit Blick auf die INSPIRE-GDI, steht auch hier die Frage im Raum, wohin letztlich die Reise führt und sich die beiden Technikwelten entwickeln werden. Die bisherigen Bemühungen zur Annäherung und des Datentransfers gingen größtenteils von der Geo Web-Gemeinde aus und bereicherten das Semantic Web. Vielleicht werden bald Mittel und Wege gefunden, die Kommunikationsbrücken und Datenströme auch in die entgegengesetzte Richtung zu lenken, um Semantic Web-Inhalte in Geoportalen oder kaskadierenden OWS zu integrieren. Eine weitergehende Harmonisierung oder gar Verschmelzung beider Technikwelten ist zu begrüßen, ergänzen sie sich doch außerordentlich gut in der Wahl der Mittel, um den Trends des heutigen Internets nach mehr Verlinkung und Verortung von Netzinhalten zu begegnen und sie mit praktischen Anwendungen zu unterfüttern.

Literaturverzeichnis

Auer, S., Lehmann, J. & Hellmann, S. [2009]. LinkedGeoData: Adding a Spatial Dimension to the Web of Data. In: A. Bernstein, D. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta & K. Thirunarayan, eds, *8th International Semantic Web Conference (ISWC 2009)*. Vol. 5823 of *Lecture Notes in Computer Science*. Springer Verlag. Berlin/Heidelberg. pp. 731–746. Chantilly, VA, USA. 25-29. Oktober 2009.

Bedini, I., Gardarin, G. & Nguyen, B. [2008]. Deriving Ontologies from XML Schema. In: *4émes Journées francophones sur les Entrepôts de Données et l 'Analyse en ligne (EDA 2008)*. Vol. B-4 of *Revue des Nouvelles Technologies de l'Information (RNTI)*. Cépaduès-Éditions. Toulouse. pp. 3–17. Toulouse, Frankreich. 5-6. Juni 2008.

Bernard, L., Fitzke, J. & Wagner, R. [2005]. *Geodateninfrastrukturen*. Herbert Wichmann Verlag. Heidelberg. ISBN: 9783879073955.

Berners-Lee, T., Hendler, J. & Lassila, O. [2001]. The Semantic Web: a new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. In: *Scientific American*. Vol. 284(5). pp. 34–43.

Boag, S., Chamberlin, D., Fernández, M., Florescu, D., Robie, J. & Siméon, J. [2010]. *XQuery 1.0: An XML Query Language (Second Edition)*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/xquery/ [Zugriff 28. Januar 2012]

Bohring, H. & Auer, S. [2005]. Mapping XML to OWL Ontologies. In: K. Jantke, K. Fähnrich & W. Wittig, eds, *13. Leipziger Informatik-Tage*. Vol. 72 of *Lecture Notes in Informatics*. Köllen Verlag. Bonn. pp. 147–156. Leipzig. 21-23. September 2005.

Brickley, D. [2003]. *W3C semantic web interest group: Basic geo (WGS84 lat/long) vocabulary*. [Online] Url: http://www.w3.org/2003/01/geo/ [Zugriff 28. Januar 2012]

Brickley, D. & Guha, R. [2004]. *RDF Vocabulary Description Language 1.0: RDF Schema*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/2004/REC-rdf-schema-20040210/ [Zugriff 28. Januar 2012]

Clark, J. & DeRose, S. [1999]. *XML Path Language (XPath) Version 1.0*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/xpath/ [Zugriff 28. Januar 2012]

Clementini, E. & Di Felice, P. [1995]. A Comparison of Methods for Representing Topological Relationships. In: *Information Sciences*. Vol. 3(3). pp. 149–178.

Cox, S. [2010]. *OGC Identifiers - the case for http URIs*. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 10-124r1. Url: http://portal.opengeospatial.org/files/?artifact_id=39467 [Zugriff 28. Januar 2012]

Cyganiak, R. [2005]. *A relational algebra for SPARQL, HPL-2005-170*. Technischer Bericht. HP-Labs. Palo Alto, CA, USA. Url: http://www.hpl.hp.com/techreports/2005/HPL-2005-170.pdf [Zugriff 28. Januar 2012]

de la Beaujardiere, J. [2006]. *OpenGIS Web Map Server Implementation Specification*. Version 1.3.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 06-042, aka International Standard ISO 19128:2005. Url: http://portal.opengeospatial.org/files/?artifact_id=14416 [Zugriff 28. Januar 2012]

Djuric, D., Gaševic, D., Devedžic, V. & Damjanovic, V. [2004]. A UML Profile for OWL Ontologies. In: U. Aßmann, M. Aksit & A. Rensink, eds, *Model Driven Architecture, European MDA Workshops*. Vol. 3599 of *Lecture Notes in Computer Science*. Springer Verlag. Berlin/Heidelberg. pp. 204–219. Linköping, Schweden. 10-11. Juni 2004.

Duerst, M. & Suignard, M. [2005]. *Internationalized Resource Identifiers (IRIs)*. International Engineering Task Force (IETF) - Network Working Group. RFC 3987. Url: http://www.ietf.org/rfc/rfc3987.txt [Zugriff 28. Januar 2012]

Egenhofer, M. & Herring, J. [1992]. *Categorizing binary topological relationships between regions, lines and points in geographic databases*. Technischer Bericht. University of Maine. Orono, ME, USA.

Elmasri, R. & Navathe, S. B. [2002]. *Grundlagen von Datenbanksystemen*. Pearson Studium. München. ISBN: 9783827370211.

Europäisches Parlament und Europäischer Rat [2007]. *Richtlinie 2007/2/EG des Europäischen Parlaments und des Rates vom 14. März 2007 zur Schaffung einer Geodateninfrastruktur in der Europäischen Gemeinschaft*. EU-Richtlinie, Amtsblatt der Europäaischen Union Nr. L 108/1, EU. Url: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32007L0002:DE:NOT [Zugriff 28. Januar 2012]

Fallside, D. C. & Walmsley, P. [2004]. *XML Schema Part 0: Primer Second Edition*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/xmlschema-0/ [Zugriff 28. Januar 2012]

Gomes Jr, L. C. & Medeiros, C. B. [2007]. Ecologically-aware Queries for Biodiversity Research. In: *9th Brazilian Symposium on GeoInformatics*. INPE - SBC.. pp. 73–84. Campos do Jordão, Brasilien. 25-28. November 2007.

Gruber, T. R. [1993]. A Translation Approach to Portable Ontology Specifications. In: *Knowledge Acquisition*. Vol. 5(2). pp. 199–220.

Harris, S. & Seaborne, A. [2012]. *SPARQL 1.1 Query Language*. World Wide Web Consortium (W3C). W3C Working Draft. Url: http://www.w3.org/TR/sparql11-query/ [Zugriff 28. Januar 2012]

Hartig, O. & Zhao, J. [2010]. Publishing and Consuming Provenance Metadata on the Web of Linked Data. In: D. McGuinness, J. Michaelis & L. Moreau, eds, *Provenance and Annotation of Data and Processes*. Vol. 6378 of *Lecture Notes in Computer Science*. pp. 78–90.

Hayes, P. [2004]. *RDF Semantics*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/ [Zugriff 28. Januar 2012]

Hebeler, J., Fisher, M., Blace, R. & Perez-Lopez, A. [2009]. *Semantic Web Programming*. John Wiley & Sons. Indianapolis, IN, USA. ISBN: 047041801X.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. & Rudolph, S. [2009]. *OWL 2 Web Ontology Language: Primer*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/owl2-primer [Zugriff 28. Januar 2012]

Hitzler, P., Krötzsch, M., Rudolph, S. & Sure, Y. [2008]. *Semantic Web : Grundlagen*. Springer Verlag. Berlin/Heidelberg. ISBN: 9783540339939.

Hitzler, P. & van Harmelen, F. [2010]. A reasonable Semantic Web. In: *Information Technology, Artificial Intelligence and Theory of Computation*. Vol. 1(1-2). pp. 39–44.

INSPIRE Drafting Team „Data Specifications“ [2008*a*]. *D2.3: Definition of Annex Themes and Scope*. Version 3.0. INSPIRE. Ref.-No. D2.3_v3.0 doc. Url: http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/DataSpecifications/D2.3_Definition_of_Annex_Themes_and_scope_v3.0.pdf [Zugriff 28. Januar 2012]

INSPIRE Drafting Team „Data Specifications“ [2008*b*]. *D2.6: Methodology for the development of data specifications*. Version 3.0. INSPIRE. Ref.-No. D2.6_v3.0.pdf. Url: http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/DataSpecifications/D2.6_v3.0.pdf [Zugriff 28. Januar 2012]

INSPIRE Drafting Team „Data Specifications“ [2010*a*]. *D2.5: Generic Conceptual Model*. Version 3.3. INSPIRE. Ref.-No. D2.5_v3.3.doc. Url: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.5_v3_3.pdf [Zugriff 28. Januar 2012]

INSPIRE Drafting Team „Data Specifications“ [2010*b*]. *D2.7: Guidelines for the encoding of spatial data*. Version 3.2. INSPIRE. Ref.-No. D2.7_v3.2.doc. Url: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.7_v3.2.pdf [Zugriff 28. Januar 2012]

INSPIRE „Network Services“ Drafting Team [2008]. *Network Services Architecture*. Version 3.0. INSPIRE. Ref.-No. D3_5_INSPIRE_NS_Architecture_v3-0.pdf. Url: http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/D3_5_INSPIRE_NS_Architecture_v3-0.pdf [Zugriff 28. Januar 2012]

INSPIRE „Network Services“ Drafting Team [2009]. *Draft Technical Guidance for INSPIRE Download Services*. Version 2.0. INSPIRE. Ref.-No. Draft_Technical_Guidance_Download_Services v2.0.doc. Url: http://inspire.jrc.ec.europa.eu/documents/Network_Services/INSPIREDraftTechnical%20GuidanceDownload(Version2.0).pdf [Zugriff 28. Januar 2012]

INSPIRE Thematic Working Group „Coordinate Reference Systems and Geographical Grid Systems“ [2010]. *D2.8.I.1 INSPIRE Specification on Coordinate Reference Systems - Guidelines*. Version 3.1. INSPIRE. Ref.-No. INSPIRE_Specification_CRS_v3.1.pdf. Url: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/INSPIRE_Specification_CRS_v3.1.pdf [Zugriff 28. Januar 2012]

INSPIRE Thematic Working Group „Protected sites“ [2010*a*]. *D2.8.I.3 INSPIRE Data Specification on Geographical Names - Guidelines*. Version 3.0.1. INSPIRE. Ref.-No. INSPIRE_DataSpecification_GN_v3.0.1.pdf. Url: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/INSPIRE_DataSpecification_GN_v3.0.1.pdf [Zugriff 28. Januar 2012]

INSPIRE Thematic Working Group „Protected sites“ [2010*b*]. *D2.8.I.9 INSPIRE Data Specification on Protected Sites - Guidelines*. Version 3.1. INSPIRE. Ref.-No. INSPIRE_DataSpecification_PS_v3.1.pdf. Url: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/INSPIRE_DataSpecification_PS_v3.1.pdf [Zugriff 28. Januar 2012]

ISO/TC 211 [2002*a*]. *ISO 19101:2002 Geographic information – Reference model*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2002*b*]. *ISO 19108:2002 Geographic information – Temporal schema*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2003*a*]. *ISO 19107:2003 Geographic information – Spatial schema*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2003*b*]. *ISO 19115:2003 Geographic information – Metadata*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2004*a*]. *ISO 19125-1:2004 Geographic information – Simple feature access – Part 1: Common architecture*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2004*b*]. *ISO 19125-1:2004 Geographic information – Simple feature access – Part 2: SQL option*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2005*a*]. *ISO 19109:2005 Geographic information – Rules for application schema*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2005*b*]. *ISO 19110:2005 Geographic information – Methodology for feature cataloguing*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2005*c*]. *ISO 19135:2005 Geographic information – Procedures for item registration*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2007*a*]. *ISO 19131:2007 Geographic information – Data product specifications*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2007*b*]. *ISO/TS 19139:2007 Geographic information – Metadata – XML schema implementation*. International Organization for Standardization (ISO). Genf.

ISO/TC 211 [2009]. *ISO 19126:2009 Geographic information – Feature concept dictionaries and registers*. International Organization for Standardization (ISO). Genf.

Jain, P., Hitzler, P., Yeh, P., Verma, K. & Sheth, A. [2010]. Linked data is merely more data. In: D. Brickley, V. Chaudhri, H. Halpin & D. McGuinness, eds, *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence*. AAAI Press. Menlo Park, CA, USA. pp. 82–86. Menlo Park, CA, USA. 22-24. März 2010.

Jain, P., Yeh, P., Verma, K., Henson, C. & Sheth, A. [2009]. SPARQL Query Re-writing Using Partonomy Based Transformation Rules. In: *GeoSpatial Semantics*. Vol. 5892 of *Lecture Notes in Computer Science*. pp. 140–158.

Janowicz, K., Schade, S., Bröring, A., Keßler, C., Maué, P. & Stasch, C. [2010]. Semantic Enablement for Spatial Data Infrastructures. In: *Transactions in GIS*. Vol. 14(2). pp. 111–129.

Kiko, K. & Atkinson, C. [2008]. *A Detailed Comparison of UML and OWL, TR-2008-004*. Technischer Bericht. Department for Mathematics and Computer Science, University of Mannheim. Mannheim.

Klien, E. [2008]. Semantic Annotation of Geographic Information. Ph. D. Thesis. Institute for Geoinformatics, University of Münster. Münster. Url: http://ifgi.uni-muenster.de/~klien/publications/Klien_PhDThesis_full.pdf [Zugriff 28. Januar 2012]

Knauff, M., Renz, J. & Rauh, R. [1998]. Empirische Ergebnisse zur konzeptionellen Adäquatheit topologischer Relationensysteme. In: U. Kotkamp & W. Krause, eds, *Intelligente Informationsverarbeitung*. Deutscher Universitätsverlag. Wiesbaden. pp. 1–8.

Kolas, D. [2008]. Supporting Spatial Semantics with SPARQL. In: *Transactions in GIS*. Vol. 12. pp. 5–18.

Lieberman, J. [2006]. *Geospatial Semantic Web Interoperability Experiment Report*. Version 0.5. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 06-002r1. Url: http://portal.opengeospatial.org/files/?artifact_id=15198 [Zugriff 28. Januar 2012]

Linking Open Data Community Project [2007]. *How to publish linked data on the web*. [Online] Url: http://sites.wiwiss.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/ [Zugriff 28. Januar 2012]

Manola, F. & Miller, E. [2004]. *RDF Primer*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/ [Zugriff 28. Januar 2012]

Miles, A. & Bechhofer, S. [2009]. *SKOS Simple Knowledge Organization System Reference*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/2009/REC-skos-reference-20090818/ [Zugriff 28. Januar 2012]

Nebert, D., Whiteside, A. & Vretanos, P. [2007]. *OGC Catalogue Services Specification*. Version 2.0.2, Corrigendum 2 Release. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 07-006r1. Url: http://portal.opengeospatial.org/files/?artifact_id=20555 [Zugriff 28. Januar 2012]

Open Geospatial Consortium Inc. [2008]. *OGC Reference Model*. Version 2.0. OGC Ref.-No. 08-062r4. Url: http://portal.opengeospatial.org/files/?artifact_id=31112 [Zugriff 28. Januar 2012]

Page, K., Roure, D. D., Martinez, K., Sadler, J. & Kit, O. [2009]. Linked Sensor Data: RESTfully serving RDF and GML. In: K. Taylor, A. Ayyagari & D. Roure, eds, *2nd International Workshop on Semantic Sensor Networks Workshop (SSN2009), in conjunction with the 8th International Semantic Web Conference (ISWC2009)*. Vol. 522. CEUR. pp. 49–63. Washington DC, USA. 26. Oktober 2009.

Patni, H., Henson, C. & Sheth, A. [2010]. Linked sensor data. In: *2010 International Symposium on Collaborative Technologies and Systems (CTS)*. IEEE. pp. 362–370. Chicago, IL, USA. 17-21. Mai 2010.

Perry, M. & Herring, J. [2011]. *GeoSPARQL - A geographic query language for RDF data*. Version 1.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 11-052r3. Url: http://portal.opengeospatial.org/files/?artifact_id=44722 [Zugriff 28. Januar 2012]

Portele, C. [2007]. *OpenGIS Geography Markup Language (GML) Encoding Standard*. Version 3.2.1. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 07-036, aka International Standard ISO 19136:2007. Url: http://portal.opengeospatial.org/files/?artifact_id=20509 [Zugriff 28. Januar 2012]

Prud'hommeaux, E. & Seaborne, A. [2008]. *SPARQL Query Language for RDF*. World Wide Web Consortium (W3C). Url: http://www.w3.org/TR/rdf-sparql-query/ [Zugriff 28. Januar 2012]

Randell, D., Cui, Z. & Cohn, A. [1992]. A spatial logic based on regions and connection. In: B. Nebel, W. Swarthout & C. Rich, eds, *Proceedings of the Third Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann. Waltham, MA, USA. pp. 165–176. Cambridge, MA, USA. 25-29. Oktober 1992.

Reed, C., Singh, R., Lake, R., Lieberman, J. & Maron, M. [2006]. *OGC White Paper - An Introduction to GeoRSS: A Standards Based Approach for Geo-enabling RSS feeds*. Version 1.0.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 06-050r3. Url: http://portal.opengeospatial.org/files/?artifact_id=15755 [Zugriff 28. Januar 2012]

Roug, S. [2009]. *Plans for SEIS and reportnet 2010, where are we headed?* [Online] Url: http://www.eionet.europa.eu/events/NRC_IS_2009/Plans%20for%20SEIS%20and%20Reportnet%202010.pdf [Zugriff 28. Januar 2012]

Schade, S., Cox, S., Panho, M., Santos, M. & Pundt, H. [2010]. Linked Data in SDI or How GML is not about Trees. In: *13th AGILE International Conference on Geographic Information Science - Geospatial Thinking*. Guimarães, Portugal. 11-14. Mai 2010. Url: http://agile2010.dsi.uminho.pt/pen/ShortPapers_PDF/73_DOC.pdf [Zugriff 28. Januar 2012]

Staub, P. [2009]. Über das Potenzial und die Grenzen der semantischen Interoperabilität von Geodaten - Ein operationelles Verfahren zur Nutzung verteilter Systeme in Geodaten-Infrastrukturen. Ph. D. Thesis. ETH Zürich. Zürich. Url: http://www.igp-data.ethz.ch/berichte/Blaue_Berichte_PDF/102.pdf [Zugriff 28. Januar 2012]

Stock, K. [2009]. *OGC Catalogue Services - OWL Application Profile of CSW*. Version 0.3.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 09-010. Url: http://portal.opengeospatial.org/files/?artifact_id=32620 [Zugriff 28. Januar 2012]

Stolze, K. [2003]. The Standard to Manage Spatial Data in Relational Database Systems. In: *10th Conference on Database Systems for Business, Technology and Web*. Vol. 26 of *Lecture Notes in Informatics*. Köllen Verlag. Bonn. pp. 247–264. Leipzig. 26-28. Februar 2003. Url: http://doesen0.informatik.uni-leipzig.de/proceedings/paper/68.pdf [Zugriff 28. Januar 2012]

Voges, U. & Senkler, K. [2007]. *OpenGIS Catalogue Services Specification 2.0.2 - ISO Metadata Application Profile*. version 1.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 07-045. Url: http://portal.opengeospatial.org/files/?artifact_id=21460 [Zugriff 28. Januar 2012]

Vretanos, P. [2005*a*]. *OGC Filter Encoding Implementation Specification*. Version 1.1.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 04-095. Url: http://portal.opengeospatial.org/files/?artifact_id=8340 [Zugriff 28. Januar 2012]

Vretanos, P. [2005*b*]. *Web Feature Service Implementation Specification*. Version 1.1.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 04-094. Url: http://portal.opengeospatial.org/files/?artifact_id=8339 [Zugriff 28. Januar 2012]

Vretanos, P. [2007]. *OWS-4 GeoDDS Mass Market (formerly GeoRSS) Interoperability Program Report*. Version 0.0.1. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 07-004. Url: http://portal.opengeospatial.org/files/?artifact_id=20431 [Zugriff 28. Januar 2012]

Vretanos, P. [2010*a*]. *OGC Filter Encoding 2.0 Encoding Standard*. Version 2.0.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 09-026r1, aka International Standard ISO 19143:2010. Url: http://portal.opengeospatial.org/files/?artifact_id=39968 [Zugriff 28. Januar 2012]

Vretanos, P. [2010*b*]. *OpenGIS Web Feature Service 2.0 Interface Standard*. Version 2.0.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 09-025r1, aka International Standard ISO 19142:2010. Url: http://portal.opengeospatial.org/files/?artifact_id=39967 [Zugriff 28. Januar 2012]

Webber, J., Parastatidis, S. & Robinson, I. [2010]. *REST in Practice: Hypermedia and Systems Architecture*. O'Reilly Media, Inc.. Sebastopol, CA, USA. ISBN: 9780596805821.

Weiser, A. & Zipf, A. [2007]. Web Service Orchestration of OGC Web Services for Disaster Management. In: J. Li, S. Zlatanova & A. G. Fabbri, eds, *Geomatics Solutions for Disaster Management*. Lecture Notes in Geoinformation and Cartography. pp. 239–254.

Whiteside, A. & Greenwood, J. [2010]. *OGC Web Service Common Standard*. Version 2.0.0. Open Geospatial Consortium Inc. (OGC). OGC Ref.-No. 06-121r9. Url: http://portal.opengeospatial.org/files/?artifact_id=38867 [Zugriff 28. Januar 2012]

Williams, G. [2012]. *SPARQL 1.1 Service Description*. World Wide Web Consortium (W3C). W3C Working Draft. Url: http://www.w3.org/TR/sparql11-service-description/ [Zugriff 28. Januar 2012]

Worboys, M. & Duckham, M. [2004]. *GIS: A Computing Perspective*. CRC Press. Boca Raton, FL, USA. ISBN: 9780415283755.

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 *Lecture Notes in Computer Science*. pp. 370–392.

Abkürzungsverzeichnis

ABox	Assertional Box
API	Application Programming Interface
AS	Applikationsschema
BBox	Bounding Box
BLOB	Binary Large Object
CQL	OGC Common Query Language
CRS	Coordinate Reference System
CRUD	Create, Read, Update, Delete
CS-W	Catalogue Service-Web
CT	Consolidation Team
CWA	Closed World Assumption
DHDN	Deutsches Hauptdreiecksnetz
DL	Description Logic (dt. Beschreibungslogik)
DOI	Digital Object Identifier
DQL	Data Query Language
DS	Datenspezifikation
DT	Drafting Team
DWG	Domain Working Group
EEA	European Environmental Agency
ESDI	European Spatial Data Infrastructure
F-Logic	Frame-Logic
FE	Filter Encoding
FGDC	Federal Geographic Data Committee
FME	Feature Manipulation Engine
FOAF	Friend of a Friend
FOL	First Order Logic (dt. Prädikatenlogik erster Stufe)
GDI	Geodateninfrastruktur (engl. SDI)
GEOSS	Global Earth Observing System of Systems
GFM	General Feature Model
GIS	Geographical Information System
GMES	Global Monitoring for Environment and Security
GML	Geography Markup Language
GSW	Geospatial Semantic Web
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IE	Interoperabilitätsexperiment
IETF	Internet Engineering Task Force

INSPIRE	Infrastructure for Spatial Information in Europe
IR	Implementing Rule
IRI	Internationalized Resource Identifier
ISO	International Organization for Standardization
KB	Knowledge Base
KML	Keyhole Markup Language
LBS	Location-based Services
LGD	LinkedGeoData
LMO	Legally Mandated Organisation
LoD	Level of Detail
LOD	Linked Open Data
LSG	Landschaftsschutzgebiet
MDA	Model-driven Architecture
MSCP	Member State Contact Point
NGDI	Nationale Geodateninfrastruktur (engl. NSDI)
NSDI	National Spatial Data Infrastructure
NUNA	Non-Unique Name Assumption
O&M	Observation & Measurement
OCL	Object Constraint Language
ODM	Ontology Definition Metamodel
OGC	Open Geospatial Consortium
OGP	International Association of Oil and Gas Producers
OpenLS	Open Location Services
OSM	OpenStreetMap
OWA	Open World Assumption
OWL	Web Ontology Language
OWS	OGC Web Service
PSI	Public Sector Information
PURL	Persistent Uniform Resource Locator
RDF	Resource Description Framework
REST	Representational State Transfer
RCC	Region Connection Calculus
RfC	Request for Comments
RIF	Rule Interchange Format
ROA	Resource-Orientated Architecture
RSS	Really Simple Syndication
SAIL	Storage and Inference Layer
SDI	Spatial Data Infrastructure
SDIC	Spatial Data Interest Community

SEL	Semantic Enablement Layer
SFA	Simple Feature Access
SKOS	Simple Knowledge Organization System
SLD	Styled Layer Descriptor
SOA	Service-Oriented Architecture
SOS	Sensor Observation Service
SPARQL	SPARQL Protocol And RDF Query Language
SPI	Service Provider Interface
SQL	Structured Query Language
TBox	Terminological Box
TC	Technical Committee
Turtle	Terse RDF Triple Language
TWG	Thematic Working Group
UCUM	Unified Code for Units of Measure
UML	Unified Modeling Language
UNA	Unique Name Assumption
UNECE	United Nations Economic Commission for Europe
UoM	Unit of Measure (dt. Maßeinheit)
URI	Uniform Resource Identifier
VGI	Volunteered Geographical Information
W3C	World Wide Web-Consortium
WaterML	Water Markup Language
WCPS	Web Coverage Processing Service
WCTS	Web Coordinate Transformation Service
WFS	Web Feature Service
WKB	Well-Known Binary
WKT	Well-Known Text
WMS	Web Map(ping) Service
WOS	Web Ontology Service
WPS	Web Processing Service
WRS	Web Reasoning Service
WS-I	Web Services-Interoperability
WWW	World Wide Web
XML	Extensible Markup Language
XSD	XML Schema Language
XTM	XML Topic Maps

Anhang A
Beispiel einer SPARQL-Auswertung mit SPARQL Algrebra-Operatoren

Demonstriert wird eine SPARQL-Auswertung anhand eines verkürzten Beispiels aus dem Grundlagenbuch *Semantic Web* [Hitzler 2008, pp. 228-232]. Die RDF-Datenbasis, die abgefragt werden soll, enthält Bücherressourcen mit Autoren-, Preis-, Titel- und Angaben zur Verkaufsstelle. Die RDF-Tripel sind in der unten stehenden Ausgangstabelle nach Subjekten, Prädikaten und Objekten in einzelnen Tabellenspalten separiert. Je Tabellenzeile ist also ein RDF-Tripel bzw. SPO-Tupel abgelegt.

RDF-Datenbasis

Subjekt	Prädikat	Objekt
ex:Hamlet	ex:Autor	ex:Shakespeare
ex:Hamlet	ex:Preis	„10.50“^^xsd:decimal
ex:Hamlet	ex:Verkaufsstelle	ex:Green_BookStore
ex:Macbeth	ex:Autor	ex:Shakespeare
ex:Macbeth	ex:Verkaufsstelle	ex:Green_BookStore
ex:Tamburlaine	ex:Autor	ex:Marlowe
ex:Tamburlaine	ex:Preis	„17“^^xsd:integer
ex:Tamburlaine	ex:Verkaufsstelle	ex:YellowPress_BookStore
ex:DoctorFaustus	ex:Autor	ex:Marlowe
ex:DoctorFaustus	ex:Preis	„12“^^xsd:integer
ex:DoctorFaustus	ex:Titel	„The Tragical History of Doctor Faustus“
ex:DoctorFaustus	ex:Verkaufsstelle	ex:Green_BookStore
ex:RomeoJulia	ex:Autor	ex:Brooke
ex:RomeoJulia	ex:Preis	„9“^^xsd:integer
ex:RomeoJulia	ex:Verkaufsstelle	ex:Green_BookStore

Die syntaktische SPARQL-Form, die der SPARQL-Benutzeranfrage entspricht, ist zunächst in die SPARQL-Algebra zu übersetzen. Mit den SPARQL Algebra-Operatoren ergibt sich ein exaktes Berechnungsschema für Vorgänge der Graphenmustererkennung und Filterung.

Aus der syntaktischen SPARQL-Anfrage:

```
1  @prefix ex: <http://example.org/> .
2
3  Select ?buch ?preis ?titel
4  {
5    ?buch ex:Preis ?preis .
6    ?buch ex:Verkaufsstelle ex:Green_BookStore .
7    OPTIONAL{
8      ?buch ex:Titel ?titel .
9    }
10   FILTER(?preis <= 10)
11 }
```

folgt der SPARQL Algebra-Ausdruck:

```
1  Filter((?preis <= 10),
2    LeftJoin(
3      BGP( ?buch <http://example.org/Preis> ?preis .
4           ?buch <http://example.org/Verkaufsstelle>  <http://example.org/Green_BookStore> . ),
5      BGP( ?buch <http://example.org/Titel> ?titel . )
6    )
7  )
```

Der SPARQL Algebra-Operator `BGP` (*Basic Graph Pattern*) führt die eigentliche Graphenmustererkennung durch. Im obigen Algebra-Ausdruck kontrolliert beispielsweise der BGP-Operator in den Zeilen 3/4, ob Bücherressourcen bepreist sind (Zeile 3) und von der Verkaufsstelle *Green_BookStore* angeboten werden. Die SPARQL Algebra-Operatoren `Filter` und `LeftJoin` gleichen Operationen der Relationalen Algebra. `Filter` ist eine attributive Selektion und `LeftJoin` gleicht dem relationalen `Left Outer Join`.

Es werden nun die einzelnen Zwischenergebnisse in tabellarischer Form wiedergegeben. Cyganiak [2005] spricht bei diesen Tabellen von *RDF-Relationen*. Auffällig sind ihre Spalten, die nicht mehr dem Subjekt-Prädikat-Objekt-Schema folgen, sondern die Variablen des letzten Auswerteschrittes abbilden. Der jeweilige, für eine RDF-Relation prägende Auswerteschritt wird im Tabellenkopf durch den Algebra-Operatornamen mitsamt der Zeilenangabe aus dem obigen SPARQL Algebra-Ausdruck genannt.

1. RDF-Relation: Ergebnis von BGP (3/4)

buch	preis
ex:Hamlet	„10.50“^^xsd:decimal
ex:DoctorFaustus	„12“^^xsd:integer
ex:RomeoJulia	„9“^^xsd:integer

2. RDF-Relation: Ergebnis von BGP (5)

buch	titel
ex:DoctorFaustus	„The Tragical History of Doctor Faustus“

3. RDF-Relation: Ergebnis von LeftJoin (2)

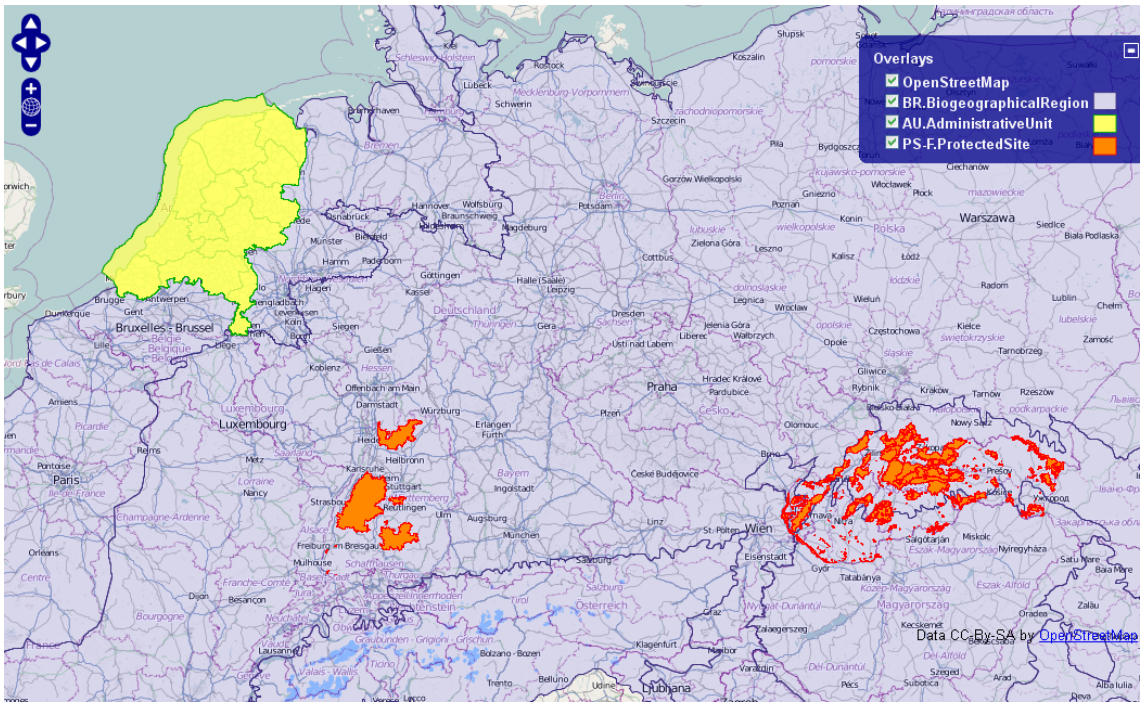
buch	preis	titel
ex:Hamlet	„10.50“^^xsd:decimal	
ex:DoctorFaustus	„12“^^xsd:integer	„The Tragical History of Doctor Faustus“
ex:RomeoJulia	„9“^^xsd:integer	

4. Endergebnis: Ergebnis von Filter (1)

buch	preis	titel
ex:RomeoJulia	„9“^^xsd:integer	

Anhang B
Auszüge aus den Testdaten

Die Testdatensätze aus Abschnitt 5.3.1 (siehe Tabelle 5.7) werden hier mit exemplarischen Feature-Instanzen wiedergegeben. Die Georeferenzierung der Geometriedaten ist EPSG:4326 (WGS84). Deren Koordinatenreihenfolge ist Long/Lat und entspricht damit nicht der OGP-Definition für das WGS84 (Lat/Long)[168], jedoch ist die hier gewählte *falsche* Koordinatenreihenfolge gängige Praxis und wurde auch vom verwendeten WFS-Dienst *Deegree3 inspireNode* vorausgesetzt. Unten ist die graphische Übersicht über die Testdaten abgebildet.



Beispiel Testdatensatz I: Schutzgebiete (SAZP-Testdaten)

Im Gegensatz zu den weiteren Testdatensätzen berücksichtigt der slovakische Schutzgebiete-Datensatz die in INSPIRE definierte Schreibweise, um Feature-Instanzen zu gruppieren (`SpatialDataSet`). Anstatt Feature-Instanzen (hier `ps-f:ProtectedSite`) direkt als `gml:featureMember` einer `gml:FeatureCollection` zu speichern, sind die Feature-Instanzen als `base:member` eines `base:SpatialDataSet`-Elementes eingegliedert. Dadurch können die Instanzdaten mit einem Identifikator versehen (GML-Property `base:identifier`) und einem inkludierten Metadatensatz (GML-Property `base:metadata`) beschrieben werden. Die Vorgehensweise ist insbesondere für fertige GML-Instanzdaten gedacht, die über einen INSPIRE-Downloaddienst für *pre-defined data sets/ pre-defined parts of data sets* angeboten werden [INSPIRE Drafting Team „Data Specifications“ 2010*a*].

[168] siehe im EPSG-Register unter: http://www.epsg-registry.org/

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0"
    xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
    xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
    xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
    xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:gmd="http://www.isotc211.org/2005/gmd"
    xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" gml:id="FC_PS_SV1"
    xsi:schemaLocation="urn:x-inspire:specification:gmlas:BaseTypes:3.2 BaseTypes.xsd
      urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0 ProtectedSitesFull.xsd">
  <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4258" srsDimension="2">
      <gml:lowerCorner>16.831479617 47.739173114</gml:lowerCorner>
      <gml:upperCorner>22.565671359 49.613793007</gml:upperCorner>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <base:SpatialDataSet gml:id="SDS1">
      <base:identifier>
        <base:Identifier>
          <base:localId>D2.8.1.9</base:localId>
          <base:namespace>SK_ProtectedSites</base:namespace>
        </base:Identifier>
      </base:identifier>
      <base:metadata>
        <gmd:MD_Metadata>
          <gmd:fileIdentifier>
            <gco:CharacterString>ProtectedSitesMetadata_20090124_ver_001</gco:CharacterString>
          </gmd:fileIdentifier>
          <gmd:language>
            <gco:CharacterString>eng</gco:CharacterString>
          </gmd:language>
          <gmd:hierarchyLevel>
            <gmd:MD_ScopeCode codeList="5" codeListValue="dataset"/>
          </gmd:hierarchyLevel>
          <gmd:contact>
            <gmd:CI_ResponsibleParty>
              <gmd:organisationName>
                <gco:CharacterString>Slovak Environmental Agency</gco:CharacterString>
              </gmd:organisationName>
              ...

        </gmd:MD_Metadata>
      </base:metadata>
      <base:member>
        <ps-f:ProtectedSite gml:id="idProtectedSitex2x7989">
          <ps:geometry>
            <gml:MultiSurface srsName="EPSG:4258" gml:id="idMultiSurfacex2x7991">
              <gml:surfaceMember>
                <gml:Polygon gml:id="idPolygonx2x7994">
                  <gml:exterior><gml:LinearRing><gml:coordinates>
                    19.50360729,49.43036375 19.503734446,49.43087722 ... 19.50360729,49.43036375
                  </gml:coordinates></gml:LinearRing></gml:exterior>
                </gml:Polygon>
              </gml:surfaceMember>
            </gml:MultiSurface>
          </ps:geometry>
          <ps:inspireID>
            <base:Identifier>
```

```
            <base:localId>10</base:localId>
            <base:namespace>SK_PS_10</base:namespace>
          </base:Identifier>
        </ps:inspireID>
        <ps:legalFoundationDate>1998-02-17T00:00:00.000</ps:legalFoundationDate>
        <ps:legalFoundationDocument>
          <gmd:CI_Citation>
            <gmd:title>
              <gco:CharacterString>Oznámenie Fedrálneho ministerstva zahranicných vecí c. 396/1990 Z. z.
              </gco:CharacterString>
            </gmd:title>
            <gmd:date>
              <gmd:CI_Date>
                <gmd:date>
                  <gco:DateTime>1998-02-17T00:00:00.000</gco:DateTime>
                </gmd:date>
                <gmd:dateType>
                  <gmd:CI_DateTypeCode codeList="2" codeListValue="publication"/>
                </gmd:dateType>
              </gmd:CI_Date>
            </gmd:date>
          </gmd:CI_Citation>
        </ps:legalFoundationDocument>
        <ps:siteDesignation>
          <ps:DesignationType>
            <ps:designationScheme>RamsarDesignationValue</ps:designationScheme>
            <ps:designation>ramsar</ps:designation>
          </ps:DesignationType>
        </ps:siteDesignation>
        <ps:siteName>
          <gn:GeographicalName>
            <gn:language>slk</gn:language>
            <gn:nativeness xsi:nil="true"/>
            <gn:nameStatus xsi:nil="true"/>
            <gn:sourceOfName xsi:nil="true"/>
            <gn:pronunciation xsi:nil="true"/>
            <gn:spelling>
              <gn:SpellingOfName>
                <gn:text>Mokrade Oravskej kotliny</gn:text>
                <gn:script xsi:nil="true"/>
              </gn:SpellingOfName>
            </gn:spelling>
          </gn:GeographicalName>
        </ps:siteName>
        <ps:siteProtectionClassification>natureConservation</ps:siteProtectionClassification>
        <ps-f:beginLifespanVersion>2009-01-23T21:55:40.000</ps-f:beginLifespanVersion>
        <ps-f:isInRegion xlink:href="#EU.BGR.3"/>
      </ps-f:ProtectedSite>
    </base:member>
    <base:member>
      ...
    </base:member>
  </base:SpatialDataSet>
 </gml:featureMember>
</gml:FeatureCollection>
```

Beispiel Testdatensatz II: Schutzgebiete (BW-Testdaten)

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
    xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
    xmlns:bgr="http://inspire.jrc.ec.europa.eu/schemas/br/2.0"
    xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0"
    xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
    xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:gmd="http://www.isotc211.org/2005/gmd"
    xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" gml:id="FC_IPS1"
    xsi:schemaLocation="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0 ProtectedSitesFull.xsd">
  <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4326">
      <gml:lowerCorner>7.59 47.57</gml:lowerCorner>
      <gml:upperCorner>9.77 49.67</gml:upperCorner>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <ps-f:ProtectedSite gml:id="DE.BW.PS.132_909027000118">
      <ps:geometry>
        <gml:Surface gml:id="id-cccceaf3-f6e2-41b8-bab9-edbc1070ea19-0" srsName="EPSG:4326" srsDimension="2">
          <gml:patches>
            <gml:PolygonPatch>
              <gml:exterior><gml:LinearRing><gml:coordinates>7.630123438445065,47.904923670520759
                7.632114047342613,47.905935024536312 ... 7.630123438445065,47.904923670520759
              </gml:coordinates></gml:LinearRing></gml:exterior>
            </gml:PolygonPatch>
          </gml:patches>
        </gml:Surface>
      </ps:geometry>
      <ps:inspireID>
        <base:Identifier>
          <base:localId>132_909027000118</base:localId>
          <base:namespace>DE.BW.PS</base:namespace>
        </base:Identifier>
      </ps:inspireID>
      <ps:legalFoundationDate>2007-11-20T00:00:00Z</ps:legalFoundationDate>
      <ps:legalFoundationDocument>
        <gmd:CI_Citation>
          <gmd:title>
            <gco:CharacterString>Landesanstalt für Umwelt, Messungen und Naturschutz vom 20.11.07
            </gco:CharacterString>
          </gmd:title>
          <gmd:date>
            <gmd:CI_Date>
              <gmd:date>
                <gco:Date>2007-11-20</gco:Date>
              </gmd:date>
              <gmd:dateType/>
            </gmd:CI_Date>
          </gmd:date>
        </gmd:CI_Citation>
      </ps:legalFoundationDocument>
      <ps:siteDesignation>
        <ps:DesignationType>
          <ps:designationScheme>natura2000</ps:designationScheme>
          <ps:designation>specialAreaOfConservation</ps:designation>
          <ps:percentageUnderDesignation>100</ps:percentageUnderDesignation>
```

```
        </ps:DesignationType>
      </ps:siteDesignation>
      <ps:siteName>
        <gn:GeographicalName>
          <gn:language>DEU</gn:language>
          <gn:nativeness>endonym</gn:nativeness>
          <gn:nameStatus xsi:nil="true"/>
          <gn:sourceOfName xsi:nil="true"/>
          <gn:pronunciation xsi:nil="true"/>
          <gn:spelling>
            <gn:SpellingOfName>
              <gn:text>Bremgarten</gn:text>
              <gn:script xsi:nil="true"/>
            </gn:SpellingOfName>
          </gn:spelling>
        </gn:GeographicalName>
      </ps:siteName>
      <ps:siteProtectionClassification>natureConservation</ps:siteProtectionClassification>
      <ps:siteProtectionClassification>ecological</ps:siteProtectionClassification>
      <ps:siteProtectionClassification>landscape</ps:siteProtectionClassification>
      <ps:siteProtectionClassification>environment</ps:siteProtectionClassification>
      <ps-f:beginLifespanVersion>2010-05-26T00:00:00Z</ps-f:beginLifespanVersion>
      <ps-f:isInRegion xlink:href="#EU.BGR.1"/>
    </ps-f:ProtectedSite>
  </gml:featureMember>
  <gml:featureMember>
        ...
  </gml:featureMember>
</gml:FeatureCollection>
```

Beispiel Testdatensatz III: Administrative Einheiten

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:gmd="http://www.isotc211.org/2005/gmd"
    xmlns:au="urn:x-inspire:specification:gmlas:AdministrativeUnits:3.0"
    xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0"
    xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
    xsi:schemaLocation="urn:x-inspire:specification:gmlas:AdministrativeUnits:3.0 AdministrativeUnits.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" gml:id="FC_AU1">
  <gml:featureMember>
    <au:AdministrativeUnit gml:id="NL.KAD.AU.GEM.0003">
      <au:geometry>
        <gml:MultiSurface gml:id="MultiSurface_NL.KAD.AU.GEM.0003" srsName="EPSG:4258">
          <gml:surfaceMember>
            <gml:Surface gml:id="Surface_NL.KAD.AU.GEM.0003.1" srsName="EPSG:4258">
              <gml:patches>
                <gml:PolygonPatch>
                  <gml:exterior><gml:LinearRing><gml:posList>6.80333796 53.31592001
                    6.80349917 53.31596690 ... 6.80333796 53.31592001
                  </gml:posList></gml:LinearRing></gml:exterior>
                </gml:PolygonPatch>
              </gml:patches>
            </gml:Surface>
          </gml:surfaceMember>
        </gml:MultiSurface>
      </au:geometry>
      <au:nationalCode>0003</au:nationalCode>
      <au:inspireId xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2">
```

```
        <base:Identifier>
          <base:localId>0003</base:localId>
          <base:namespace>NL.KAD.AU.GEM</base:namespace>
        </base:Identifier>
      </au:inspireId>
      <au:nationalLevel>3rdOrder</au:nationalLevel>
        <au:nationalLevelName xmlns:gmd="http://www.isotc211.org/2005/gmd">
        <gmd:LocalisedCharacterString locale="nl-NL">gemeente</gmd:LocalisedCharacterString>
      </au:nationalLevelName>
      <au:country xmlns:gmd="http://www.isotc211.org/2005/gmd">
        <gmd:Country codeList="http://schemas.kademo.nl/inspire/codelist-1004/CountryCode.xml"
          codeListValue="NL">NL</gmd:Country>
      </au:country>
      <au:name xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0">
        <gn:GeographicalName>
          <gn:language xsi:nil="true"></gn:language>
          <gn:nativeness>endonym</gn:nativeness>
          <gn:nameStatus>official</gn:nameStatus>
          <gn:sourceOfName>Het Kadaster</gn:sourceOfName>
          <gn:pronunciation xsi:nil="true" nilReason="UNPOPULATED"/>
          <gn:spelling>
            <gn:SpellingOfName>
              <gn:text>Appingedam</gn:text>
              <gn:script>Latn</gn:script>
            </gn:SpellingOfName>
          </gn:spelling>
          <gn:grammaticalGender xsi:nil="true"/>
          <gn:grammaticalNumber xsi:nil="true"/>
        </gn:GeographicalName>
      </au:name>
      <au:residenceOfAuthority xsi:nil="true"/>
      <au:beginLifespanVersion xsi:nil="true"/>
      <au:endLifespanVersion xsi:nil="true"/>
      <au:NUTS xsi:nil="true"/>
      <au:condominium xsi:nil="true"/>
      <au:lowerLevelUnit xsi:nil="true"/>
      <au:upperLevelUnit xsi:nil="true"/>
      <au:administeredBy xsi:nil="true"/>
      <au:coAdminister xsi:nil="true"/>
      <au:boundary xsi:nil="true"/>
    </au:AdministrativeUnit>
  </gml:featureMember>
  <gml:featureMember>
    ...
  </gml:featureMember>
</gml:FeatureCollection>
```

Beispiel Testdatensatz IV: Biogeographische Regionen

```
<?xml version="1.0" encoding="utf-8"?>
<gml:FeatureCollection xmlns:bgr="http://inspire.jrc.ec.europa.eu/schemas/br/2.0"
    xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" gml:id="FC_BGR1"
    xsi:schemaLocation="http://inspire.jrc.ec.europa.eu/schemas/br/2.0 BiogeographicalRegions.xsd">
  <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4326">
      <gml:lowerCorner>-9.290690422058106 40.73057556152344</gml:lowerCorner>
      <gml:upperCorner>57.78695678710938 57.61869049072266</gml:upperCorner>
```

```
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <bgr:Bio-geographicalRegion gml:id="EU.BGR.1">
      <bgr:beginLifespanVersion>2010-05-26T00:00:00Z</bgr:beginLifespanVersion>
      <bgr:inspireId>
        <base:Identifier>
          <base:localId>1</base:localId>
          <base:namespace>EU.BGR</base:namespace>
        </base:Identifier>
      </bgr:inspireId>
      <bgr:regionClassification>continental</bgr:regionClassification>
      <bgr:regionClassificationLevel>international</bgr:regionClassificationLevel>
      <bgr:regionClassificationScheme>natura2000AndEmeraldBio-geographicalRegion</bgr:regionClassificationScheme>
      <bgr:shape>
        <gml:MultiSurface srsName="EPSG:4326" gml:id="MultiSurface1">
          <gml:surfaceMember>
            <gml:Polygon gml:id="Polygon">
              <gml:exterior><gml:LinearRing><gml:coordinates>57.687252044677727,56.128696441650391
                57.623912811279297,56.086315155029297 ... 57.687252044677727,56.128696441650391
              </gml:coordinates></gml:LinearRing></gml:exterior>
              <gml:interior><gml:LinearRing><gml:coordinates>23.584499359130856,43.070060729980469
                23.614446640014648,43.0760498046875 ... 23.584499359130856,43.070060729980469
              </gml:coordinates></gml:LinearRing></gml:interior>
            </gml:Polygon>
          </gml:surfaceMember>
        </gml:MultiSurface>
      </bgr:shape>
    </bgr:Bio-geographicalRegion>
  </gml:featureMember>
  <gml:featureMember>
        ...
  </gml:featureMember>
</gml:FeatureCollection>
```

Anhang C
Modellierungsbeispiele, Themenontologie Protected Sites

Beispiel einer Schutzgebiete-Instanz mit Attributierungen

Das unten stehende Beispiel einer Schutzgebiete-Instanz entspricht dem GML Feature-Instanzenbeispiel des baden-württembergischen Testdatensatzes aus Anhang B (Testdatensatz II). Das Beispiel ist in der RDF-Formatierung Turtle gehalten. Die hier angewendeten Konzepte aus den Themenontologien sind im Anschluß auszugsweise dargestellt.

```
@prefix cf: <http://inspire.west.uni-koblenz.de/configuration/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/> .
@prefix natura2000DesignationValue: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                     Natura2000DesignationValue/> .
@prefix designationSchemeValue: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                  DesignationSchemeValue/> .
@prefix protectionClassificationValue: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                        ProtectionClassificationValue/> .
@prefix br: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions/> .
@prefix base: <http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes/> .
@prefix geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/> .
@prefix gml32: <http://www.opengis.net/def/dataType/OGC-GML/3.2/> .
@prefix geoext: <http://inspire.west.uni-koblenz.de/configuration/GeoSPARQLReviewExtension/> .


<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> a ps:ProtectedSite ;
                                                                 ps:geometry _:nodeGeometry1 ;
                                                                 geo:hasDefaultGeometry _:nodeGeometry1 .

_:nodeGeometry1 a geo:Geometry ,
             geo:SpatialObject ;
             geoext:asGML
             """<gml:Surface gml:id=\"id-cccceaf3-f6e2-41b8-bab9-edbc1070ea19-0\"
                    srsName=\"EPSG:4326\" srsDimension=\"2\" xmlns:gml=\"http://www.opengis.net/gml/3.2\">
                  <gml:patches>
                    <gml:PolygonPatch>
                      <gml:exterior><gml:LinearRing><gml:coordinates>7.630123438445065,47.904923670520759
                          7.632114047342613,47.905935024536312 ... 7.630123438445065,47.904923670520759
                      </gml:coordinates></gml:LinearRing></gml:exterior>
                    </gml:PolygonPatch>
                  </gml:patches>
                </gml:Surface>"""^^gml32:GMLLiteral ;
             geo:dimension "2"^^xsd:integer ;
             geo:coordinateDimension "2"^^xsd:integer ;
             geoext:refLocation_lat "48.62"^^xsd:decimal ;
             geoext:refLocation_long "8.68"^^xsd:decimal ;
```

```
            geoext:bbox_minLat "47.57"^^xsd:decimal ;
            geoext:bbox_minLong "7.59"^^xsd:decimal ;
            geoext:bbox_maxLat "49.67"^^xsd:decimal ;
            geoext:bbox_maxLong "9.77"^^xsd:decimal .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:inspireID _:nodeId1 .
_:nodeId1 a base:Identifier ;
          base:namespace "DE.BW.PS" ;
          base:localId "132_909027000118" .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:legalFoundationDate
                                                                 "2007-11-20T00:00:00Z"^^xsd:dateTime .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:legalFoundationDocument _:nodeCitation1 .
_:nodeCitation1 a base:CI_Citation ;
                base:title "Landesanstalt für Umwelt, Messungen und Naturschutz vom 20.11.07" ;
                base:date "2007-11-20T00:00:00Z"^^xsd:dateTime .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:siteDesignation _:nodeDesignation1 .
_:nodeDesignation1 ps:designation natura2000DesignationValue:specialAreaOfConservation ;
                   ps:designationScheme designationSchemeValue:natura2000 ;
                   ps:percentageUnderDesignation "100"^^xsd:float .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:siteName "Bremgarten"@de .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:siteProtectionClassification
                                                                 protectionClassificationValue:natureConservation ,
                                                                 protectionClassificationValue:ecological ,
                                                                 protectionClassificationValue:landscape ,
                                                                 protectionClassificationValue:environment .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:beginLifespanVersion
                                                                 "2010-05-26T00:00:00Z"^^xsd:dateTime .

<http://inspire.jrc.ec.europa.eu/data/DE.BW.PS/132_909027000118> ps:isInRegion
                                                                 <http://inspire.jrc.ec.europa.eu/data/EU.BGR/1> .
<http://inspire.jrc.ec.europa.eu/data/EU.BGR/1> a br:Bio-geographicalRegion .
```

Auszüge aus den Konzepten der Themenontologie Protected Sites

Zur leichteren Orientierung listet die unten stehende Tabelle alle Konzepte auf, die im Anschluß an die Tabelle mit Deklarationen im RDF-Format RDF/XML folgen. Die Präfixe der Konzepte (z.B. `ps` von `ps:ProtectedSite`) deuten auf die jeweiligen Vokabulare hin, die in die Themenontologie *Protected Sites* importiert sind: der Präfix `geo` verweist auf das GeoSPARQL-Vokabular (aus der RfC-Version), `geoext` auf die eigenen GeoSPARQL-Erweiterungsvorschläge und `base` auf die Ontologie der INSPIRE/ISO-Basistypen. Die Tabelle benennt in der zweiten Spalte den OWL-Konzepttyp und in der dritten Spalte das jeweilige Pendant in den INSPIRE/ISO GML-Applikationsschemata.

Konzeptname	Konzepttyp	GML-Schemaname
ps:ProtectedSite	owl:Class, geo:Feature	ps:ProtectedSite, ps-f:ProtectedSite
ps:DesignationType	owl:Class	ps:DesignationType
ps:geometry	owl:ObjectProperty	ps:geometry
ps:inspireID	owl:ObjectProperty	ps:inspireID

ps:legalFoundationDate	owl:DatatypeProperty	ps:legalFoundationDate
ps:legalFoundationDocument	owl:ObjectProperty	ps:legalFoundationDocument
ps:siteDesignation	owl:ObjectProperty	ps:siteDesignation
ps:designation	owl:ObjectProperty	ps:designation
ps:designationSchema	owl:ObjectProperty	ps:designationSchema
ps:percentageUnderDesignation	owl:DatatypeProperty	ps:percentageUnderDesignation
ps:siteName	owl:DatatypeProperty	ps:siteName
ps:siteProtectionClassification	owl:ObjectProperty	ps:siteProtectionClassification
ps:officialsiteArea	owl:ObjectProperty	ps-f:officialsiteArea
ps:beginLifespanVersion	owl:DatatypeProperty	ps-f:beginLifespanVersion
ps:isInRegion	owl:ObjectProperty	ps-f:isInRegion
geo:SpatialObject	owl:Class	gml:AbstractGMLType
geo:Feature	owl:Class	gml:AbstractFeatureType
geo:Geometry	owl:Class	gml:AbstractGeometryType
geo:hasGeometry	owl:ObjectProperty	–
geo:hasDefaultGeometry	owl:ObjectProperty	–
geoext:hasOriginalGeometry	owl:ObjectProperty	–
geoext:generalizes	owl:ObjectProperty	–
geoext:generalizedBy	owl:ObjectProperty	–
geoext:contains	owl:ObjectProperty	–
geoext:intersects	owl:ObjectProperty	–
base:CodelistRegister	owl:NamedIndividual, skos:ConceptScheme	INSPIRE-Register Codelists/Enumerations
ps:DesignationSchemeValue	owl:Class	INSPIRE-Codelist
DesignationSchemeValue:natura2000	owl:NamedIndividual, skos:Concept	INSPIRE-Codelist-Wert
DesignationSchemeValue: UNESCOWorldHeritage	owl:NamedIndividual, skos:Concept	INSPIRE-Codelist-Wert
ps:DesignationValue	owl:Class	abstrakte INSPIRE-Codelist
ps:Natura2000DesignationValue	owl:Class, ps:DesignationValue	INSPIRE-Codelist
Natura2000DesignationValue: specialAreaOfConservation	owl:NamedIndividual, skos:Concept	INSPIRE-Codelist-Wert
Natura2000DesignationValue: proposedSpecialProtectionArea	owl:NamedIndividual, skos:Concept	INSPIRE-Codelist-Wert
ps:ProtectionClassificationValue	owl:Class	INSPIRE-Codelist
ProtectionClassificationValue:landscape	owl:NamedIndividual, skos:Concept	INSPIRE-Codelist-Wert

base:CrsUomRegister	owl:NamedIndividual, skos:ConceptScheme	INSPIRE-Register CRSs/UoMs
base:CrsUomRegister/uom/area/hectare	owl:NamedIndividual, skos:Concept, base:UnitOfMeasure	–
base:Measure	owl:Class, muo:QualityValue	gml:MeasureType
base:Area	owl:Class, base:Measure	gml:AreaType
base:unitOfMeasure	owl:ObjectProperty, muo:measuredIn	Attribut „uom“ von Element gml:MeasureType
base:UnitOfMeasure	owl:Class, muo:UnitOfMeasure	–
base:numericalValue	owl:DatatypeProperty, muo:numericalValue	Elementinhalt von gml:MeasureType
base:Identifier	owl:Class	base:Identifier
base:namespace	owl:DatatypeProperty	base:namespace
base:localId	owl:DatatypeProperty	base:localId
base:versionId	owl:DatatypeProperty	base:versionId
br:Bio-geographicalRegion	owl:Class, geo:Feature	br:Bio-geographicalRegion

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY dc "http://purl.org/dc/elements/1.1/" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY skos "http://www.w3.org/2004/02/skos/core#" >

  <!ENTITY cf "http://inspire.west.uni-koblenz.de/configuration/" >
  <!ENTITY ps "http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/" >
  <!ENTITY br "http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions/" >
  <!ENTITY base "http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes/" >
  <!ENTITY geo "http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/" >
  <!ENTITY geoext "http://inspire.west.uni-koblenz.de/configuration/GeoSPARQLReviewExtension/" >
  <!ENTITY DesignationValue "http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/DesignationValue/" >
  <!ENTITY Natura2000DesignationValue "http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                       Natura2000DesignationValue/" >
  <!ENTITY DesignationSchemeValue "http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                       DesignationSchemeValue/" >
  <!ENTITY ProtectionClassificationValue "http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                       ProtectionClassificationValue/" >
  <!ENTITY muo "http://purl.oclc.org/NET/muo/muo#" >
  <!ENTITY physical-quality "http://purl.oclc.org/NET/muo/ucum/physical-quality/" >
  <!ENTITY ucumPrefix "http://purl.oclc.org/NET/muo/ucum/prefix/" >
  <!ENTITY UNECE "urn:ogc:def:uom:UNECE::" > ]>

<rdf:RDF xmlns="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/"
    xml:base="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
```

```
  xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#" xmlns:skos="http://www.w3.org/2004/02/skos/core#"
  xmlns:cf="http://inspire.west.uni-koblenz.de/configuration/"
  xmlns:ps="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/"
  xmlns:br="http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions/"
  xmlns:base="http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes/"
  xmlns:geo="http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/" xmlns:muo="http://purl.oclc.org/NET/muo/muo#"
  xmlns:DesignationValue="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/DesignationValue/"
  xmlns:Natura2000DesignationValue="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                    Natura2000DesignationValue/"
  xmlns:DesignationSchemeValue="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                    DesignationSchemeValue/"
  xmlns:ProtectionClassificationValue="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/
                                    ProtectionClassificationValue/">

  <owl:Ontology rdf:about="http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites">
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:ps=&quot;urn:x-inspire:specification:gmlas:ProtectedSites:3.0&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:ps-f=&quot;urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:base=&quot;urn:x-inspire:specification:gmlas:BaseTypes:3.2&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:br=&quot;http://inspire.jrc.ec.europa.eu/schemas/br/2.0&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:gn=&quot;urn:x-inspire:specification:gmlas:GeographicalNames:3.0&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:gmd=&quot;http://www.isotc211.org/2005/gmd&quot;</cf:xmlNamespace>
      <cf:xmlNamespace rdf:datatype="&rdf;PlainLiteral">
        xmlns:gml=&quot;http://www.opengis.net/gml/3.2&quot;</cf:xmlNamespace>
      <owl:imports rdf:resource="http://inspire.jrc.ec.europa.eu/vocabulary/baseTypes"/>
      <owl:imports rdf:resource="http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions"/>
      <owl:imports rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
  </owl:Ontology>

<!-- OWL:CLASS ps:ProtectedSite -->

  <owl:Class rdf:about="&ps;ProtectedSite">
      <rdfs:label>protected site</rdfs:label>
      <dc:description>An area designated or managed within a framework of international, Community and Member
        States&#39; legislation to achieve specific conservation objectives. section 5.2.2.1.1 and 5.2.3.1.1
      </dc:description>
      <cf:entityType>featureType</cf:entityType>
      <cf:uriSyntax>http://inspire.jrc.ec.europa.eu/data/{ps:inspireID/base:Identifier/base:namespace}/
        {ps:inspireID/base:Identifier/base:localId}[/{ps:inspireID/base:Identifier/base:versionId}]</cf:uriSyntax>
      <cf:xmlName>ps-f:ProtectedSite</cf:xmlName>
      <cf:xmlName>ps:ProtectedSite</cf:xmlName>
      <rdfs:subClassOf rdf:resource="&geo;Feature"/>
      <rdfs:subClassOf>
          <owl:Restriction>
              <owl:onProperty rdf:resource="&ps;geometry"/>
              <owl:allValuesFrom rdf:resource="&geo;Geometry"/>
          </owl:Restriction>
      </rdfs:subClassOf>
      <rdfs:subClassOf>
          <owl:Restriction>
              <owl:onProperty rdf:resource="&ps;inspireID"/>
              <owl:allValuesFrom rdf:resource="&base;Identifier"/>
          </owl:Restriction>
```

```
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;legalFoundationDate"/>
                <owl:allValuesFrom rdf:resource="&xsd;dateTime"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;legalFoundationDocument"/>
                <owl:allValuesFrom rdf:resource="&base;CI_Citation"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;siteDesignation"/>
                <owl:allValuesFrom rdf:resource="&ps;DesignationType"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;siteName"/>
                <owl:allValuesFrom rdf:resource="&rdf;PlainLiteral"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;siteProtectionClassification"/>
                <owl:allValuesFrom rdf:resource="&ps;ProtectionClassificationValue"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;officialsiteArea"/>
                <owl:allValuesFrom rdf:resource="&base;Measure"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;beginLifespanVersion"/>
                <owl:allValuesFrom rdf:resource="&xsd;dateTime"/>
            </owl:Restriction>
        </rdfs:subClassOf>
          ...
    </owl:Class>

<!-- OWL:CLASS ps:DesignationType -->

    <owl:Class rdf:about="&ps;DesignationType">
        <rdfs:label>designation type</rdfs:label>
        <dc:description>A data type designed to contain a designation for the Protected Site, including the
          designation scheme used and the value within that scheme. INSPIRE Data Specification
          Protected Sites section 5.2.2.2.1</dc:description>
        <cf:entityType>dataType</cf:entityType>
        <cf:xmlName>ps:DesignationType</cf:xmlName>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;designation"/>
                <owl:allValuesFrom rdf:resource="&ps;DesignationValue"/>
```

```
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;designationScheme"/>
                <owl:allValuesFrom rdf:resource="&ps;DesignationSchemeValue"/>
            </owl:Restriction>
        </rdfs:subClassOf>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&ps;percentageUnderDesignation"/>
                <owl:allValuesFrom rdf:resource="&xsd;float"/>
            </owl:Restriction>
        </rdfs:subClassOf>
    </owl:Class>

<!-- OWL:OBJECTPROPERTY ps:geometry -->

    <owl:ObjectProperty rdf:about="&ps;geometry">
        <rdfs:label>geometry</rdfs:label>
        <dc:description>The geometry defining the boundary ... section 5.2.2.1.1</dc:description>
        <cf:xpath>ps:ProtectedSite/ps:geometry</cf:xpath>
        <cf:xpath>ps-f:ProtectedSite/ps:geometry</cf:xpath>
        <rdfs:subPropertyOf rdf:resource="&geo;hasGeometry"/>
    </owl:ObjectProperty>
    <owl:Axiom>
        <owl:annotatedSource rdf:resource="&ps;geometry"/>
        <owl:annotatedProperty rdf:resource="&cf;xpath"/>
        <owl:annotatedTarget>ps:ProtectedSite/ps:geometry</owl:annotatedTarget>
        <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
        <cf:rdfsRange rdf:resource="&geo;Geometry"/>
    </owl:Axiom>
    <owl:Axiom>
        <owl:annotatedSource rdf:resource="&ps;geometry"/>
        <owl:annotatedProperty rdf:resource="&cf;xpath"/>
        <owl:annotatedTarget>ps-f:ProtectedSite/ps:geometry</owl:annotatedTarget>
        <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
        <cf:rdfsRange rdf:resource="&geo;Geometry"/>
    </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:inspireID -->

    <owl:ObjectProperty rdf:about="&ps;inspireID">
        <rdfs:label>inspireID</rdfs:label>
        <cf:xpath>ps:ProtectedSite/ps:inspireID/base:Identifier</cf:xpath>
        <cf:xpath>ps-f:ProtectedSite/ps:inspireID/base:Identifier</cf:xpath>
    </owl:ObjectProperty>
    <owl:Axiom>
        <owl:annotatedSource rdf:resource="&ps;inspireID"/>
        <owl:annotatedProperty rdf:resource="&cf;xpath"/>
        <owl:annotatedTarget>ps:ProtectedSite/ps:inspireID/base:Identifier</owl:annotatedTarget>
        <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
        <cf:rdfsRange rdf:resource="&base;Identifier"/>
    </owl:Axiom>
    <owl:Axiom>
        <owl:annotatedSource rdf:resource="&ps;inspireID"/>
        <owl:annotatedProperty rdf:resource="&cf;xpath"/>
        <owl:annotatedTarget>ps-f:ProtectedSite/ps:inspireID/base:Identifier</owl:annotatedTarget>
        <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
```

```
      <cf:rdfsRange rdf:resource="&base;Identifier"/>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY ps:legalFoundationDate -->

  <owl:DatatypeProperty rdf:about="&ps;legalFoundationDate">
      <rdfs:label>legal foundation date</rdfs:label>
      <dc:description>The date that the protected ... section 5.2.2.1.1</dc:description>
      <cf:xpath>ps-f:ProtectedSite/ps:legalFoundationDate</cf:xpath>
      <cf:xpath>ps:ProtectedSite/ps:legalFoundationDate</cf:xpath>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&xsd;dateTime"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:legalFoundationDocument -->

  <owl:ObjectProperty rdf:about="&ps;legalFoundationDocument">
      <rdfs:label>legal foundation document</rdfs:label>
      <dc:description>A URL or text citation referencing the legal act ... section 5.2.2.1.1</dc:description>
      <cf:xpath>ps:ProtectedSite/ps:legalFoundationDocument/gmd:CI_Citation</cf:xpath>
      <cf:xpath>ps-f:ProtectedSite/ps:legalFoundationDocument/gmd:CI_Citation</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&base;CI_Citation"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:siteDesignation -->

  <owl:ObjectProperty rdf:about="&ps;siteDesignation">
      <rdfs:label>site designation</rdfs:label>
      <dc:description>The designation (type) of Protected Site ... section 5.2.2.1.1</dc:description>
      <cf:xpath>ps:ProtectedSite/ps:siteDesignation/ps:DesignationType</cf:xpath>
      <cf:xpath>ps-f:ProtectedSite/ps:siteDesignation/ps:DesignationType</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&ps;DesignationType"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:designation -->

  <owl:ObjectProperty rdf:about="&ps;designation">
      <rdfs:label>designation</rdfs:label>
      <dc:description>The actual Site designation ... section 5.2.2.2.1</dc:description>
      <cf:xpath>ps:DesignationType/ps:designation</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;DesignationType"/>
      <cf:rdfsRange rdf:resource="&ps;DesignationValue"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:designationScheme -->
```

```
  <owl:ObjectProperty rdf:about="&ps;designationScheme">
      <rdfs:label>designationScheme</rdfs:label>
      <dc:description>The scheme from which the designation code comes ... section 5.2.2.2.1</dc:description>
      <cf:xpath>ps:DesignationType/ps:designationScheme</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;DesignationType"/>
      <cf:rdfsRange rdf:resource="&ps;DesignationSchemeValue"/>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY ps:percentageUnderDesignation -->

  <owl:DatatypeProperty rdf:about="&ps;percentageUnderDesignation">
      <rdfs:label>percentage under designation</rdfs:label>
      <dc:description>The percentage of the site that falls under the designation. This is
        used in particular for the IUCN categorisation. If a value is not provided for this
        attribute, it is assumed to be 100% ... section 5.2.2.2.1</dc:description>
      <cf:xpath>ps:DesignationType/ps:percentageUnderDesignation</cf:xpath>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;DesignationType"/>
      <cf:rdfsRange rdf:resource="&xsd;float"/>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY ps:siteName -->

  <owl:DatatypeProperty rdf:about="&ps;siteName">
      <rdfs:label>site name</rdfs:label>
      <dc:description>The name of the Protected Site ... section 5.2.2.1.1</dc:description>
      <cf:xpath>ps:ProtectedSite/ps:siteName/gn:GeographicalName/gn:spelling/gn:SpellingOfName/gn:text</cf:xpath>
      <cf:xpath>ps-f:ProtectedSite/ps:siteName/gn:GeographicalName/gn:spelling/gn:SpellingOfName/gn:text</cf:xpath>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&rdf;PlainLiteral"/>
      <cf:langXPath>../../../gn:language</cf:langXPath>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:siteProtectionClassification -->

  <owl:ObjectProperty rdf:about="&ps;siteProtectionClassification">
      <rdfs:label>site protection classification</rdfs:label>
      <dc:description>The classification of the protected site based ... section 5.2.2.1.1</dc:description>
      <cf:xpath>ps:ProtectedSite/ps:siteProtectionClassification</cf:xpath>
      <cf:xpath>ps-f:ProtectedSite/ps:siteProtectionClassification</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&ps;ProtectionClassificationValue"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:officialsiteArea -->

  <owl:ObjectProperty rdf:about="&ps;officialsiteArea">
```

```
      <rdfs:label>official site area</rdfs:label>
      <dc:description>The official area of the site in hectares ... section 5.2.3.1.1</dc:description>
      <cf:xpath>ps-f:ProtectedSite/ps-f:officialsiteArea</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
      <owl:annotatedSource rdf:resource="&ps;officialsiteArea"/>
      <owl:annotatedProperty rdf:resource="&cf;xpath"/>
      <owl:annotatedTarget>ps-f:ProtectedSite/ps-f:officialsiteArea</owl:annotatedTarget>
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&base;Measure"/>
      <cf:uomXPath>@uom</cf:uomXPath>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY ps:beginLifespanVersion -->

  <owl:DatatypeProperty rdf:about="&ps;beginLifespanVersion">
      <rdfs:label>begin lifespan version</rdfs:label>
      <dc:description>Date and time at which this version of the spatial object was inserted or changed in the
          spatial data set. INSPIRE Data Specification Protected Sites section 5.2.3.1.1 and 5.2.3.1.2</dc:description>
      <cf:xpath>ps-f:ProtectedSite/ps-f:beginLifespanVersion</cf:xpath>
      <cf:xpath>ps-f:ResponsibleAgency/ps-f:beginLifespanVersion</cf:xpath>
      <owl:propertyDisjointWith rdf:resource="&ps;endLifespanVersion"/>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ResponsibleAgency"/>
      <cf:rdfsRange rdf:resource="&xsd;dateTime"/>
  </owl:Axiom>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&xsd;dateTime"/>
  </owl:Axiom>

<!-- OWL:OBJECTPROPERTY ps:isInRegion -->

  <owl:ObjectProperty rdf:about="&ps;isInRegion">
      <rdfs:label>isInRegion</rdfs:label>
      <dc:description> section 5.2.3.1.1 and 5.2.3.1.2</dc:description>
      <cf:xpath>ps-f:ProtectedSite/ps-f:isInRegion/br:Bio-geographicalRegion</cf:xpath>
  </owl:ObjectProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&ps;ProtectedSite"/>
      <cf:rdfsRange rdf:resource="&br;Bio-geographicalRegion"/>
  </owl:Axiom>

<!-- OWL:CLASS geo:SpatialObject -->

  <owl:Class rdf:about="&geo;SpatialObject">
      <rdfs:label>SpatialObject</rdfs:label>
      <rdfs:comment>The class SpatialObject, superclass of everything feature
        or geometry that can have a spatial representation.</rdfs:comment>
      <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
  </owl:Class>

<!-- OWL:CLASS geo:Feature -->

  <owl:Class rdf:about="&geo;Feature">
```

```
        <rdfs:label>Feature</rdfs:label>
        <rdfs:comment>The class Feature, superclass of everything feature.</rdfs:comment>
        <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
        <rdfs:subClassOf rdf:resource="&geo;SpatialObject"/>
        <owl:equivalentClass>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&geo;hasGeometry"/>
                <owl:someValuesFrom rdf:resource="&geo;Geometry"/>
            </owl:Restriction>
        </owl:equivalentClass>
    </owl:Class>

<!-- OWL:CLASS geo:Geometry -->

    <owl:Class rdf:about="&geo;Geometry">
        <rdfs:label>geometry</rdfs:label>
        <rdfs:comment>The class Geometry, superclass of everything geometry.</rdfs:comment>
        <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
        <rdfs:subClassOf rdf:resource="&geo;SpatialObject"/>
        <cf:entityType>geometry</cf:entityType>
    </owl:Class>

<!-- OWL:OBJECTPROPERTY geo:hasGeometry -->

    <owl:ObjectProperty rdf:about="&geo;hasGeometry">
        <rdfs:label>hasGeometry</rdfs:label>
        <rdfs:comment>A spatial representation for a given feature.</rdfs:comment>
        <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
        <rdfs:domain rdf:resource="&geo;Feature"/>
        <rdfs:range rdf:resource="&geo;Geometry"/>
    </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geo:hasDefaultGeometry -->

    <owl:ObjectProperty rdf:about="&geo;hasDefaultGeometry">
        <rdfs:label>hasDefaultGeometry</rdfs:label>
        <rdfs:comment>The default geometry to be used in spatial calculations</rdfs:comment>
        <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
        <rdfs:subPropertyOf rdf:resource="&geo;hasGeometry"/>
        <rdfs:domain rdf:resource="&geo;Feature"/>
        <rdfs:range rdf:resource="&geo;Geometry"/>
    </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geoext:hasOriginalGeometry -->

    <owl:ObjectProperty rdf:about="&geoext;hasOriginalGeometry">
        <rdfs:label>hasOriginalGeometry</rdfs:label>
        <rdfs:comment>The original or most detailled spatial representation for a given feature</rdfs:comment>
        <rdfs:isDefinedBy rdf:resource="http://www.opengis.net/spec/GeoSPARQL/1.0"/>
        <rdfs:subPropertyOf rdf:resource="&geo;hasGeometry"/>
        <rdfs:domain rdf:resource="&geo;Feature"/>
        <rdfs:range rdf:resource="&geo;Geometry"/>
    </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geoext:generalizes -->

    <owl:ObjectProperty rdf:about="&geoext;generalizes">
        <rdfs:label>generalizes</rdfs:label>
        <rdfs:comment>The geometry which is the predicate-subject generalizes
```

```
        another geometry as the predicate-object</rdfs:comment>
      <rdfs:domain rdf:resource="&geo;Geometry"/>
      <rdfs:range rdf:resource="&geo;Geometry"/>
      <owl:inverseOf rdf:resource="&geoext;generalizedBy"/>
  </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geoext:generalizedBy -->

  <owl:ObjectProperty rdf:about="&geoext;generalizedBy">
      <rdfs:label>generalizedBy</rdfs:label>
      <rdfs:comment>The geometry which is the predicate-subject is generalized by
        another geometry as the predicate-object</rdfs:comment>
      <rdfs:domain rdf:resource="&geo;Geometry"/>
      <rdfs:range rdf:resource="&geo;Geometry"/>
      <owl:inverseOf rdf:resource="&geoext;generalizes"/>
  </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geoext:contains -->

  <owl:ObjectProperty rdf:about="&geoext;contains">
      <rdfs:label>contains</rdfs:label>
      <rdfs:comment>Simple Features topological relation 'contains' between two Spatial objects</rdfs:comment>
      <rdfs:domain rdf:resource="&geo;SpatialObject"/>
      <rdfs:range rdf:resource="&geo;SpatialObject"/>
  </owl:ObjectProperty>

<!-- OWL:OBJECTPROPERTY geoext:intersects -->

  <owl:ObjectProperty rdf:about="&geoext;intersects">
      <rdfs:label>intersects</rdfs:label>
      <rdfs:comment>Simple Features topological relation 'intersects' between two Spatial objects</rdfs:comment>
      <rdfs:domain rdf:resource="&geo;SpatialObject"/>
      <rdfs:range rdf:resource="&geo;SpatialObject"/>
  </owl:ObjectProperty>

<!-- OWL:NAMEDINDIVIDUAL base:CodelistRegister -->

  <skos:ConceptScheme rdf:about="&base;CodelistRegister">
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdfs:label>Codelist Register</rdfs:label>
      <skos:prefLabel>Codelist Register</skos:prefLabel>
      <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/
          Data_Specifications/D2.5_v3_3.pdf"/>
  </skos:ConceptScheme>

<!-- OWL:CLASS ps:DesignationSchemeValue -->

  <owl:Class rdf:about="&ps;DesignationSchemeValue">
      <rdfs:label>designation scheme value</rdfs:label>
      <dc:description>The scheme used to assign a designation to ... section 5.2.2.3.2</dc:description>
      <skos:inScheme rdf:resource="&base;CodelistRegister"/>
      <cf:entityType>codeList</cf:entityType>
      <rdfs:subClassOf>
          <owl:Class>
              <owl:oneOf rdf:parseType="Collection">
                  <rdf:Description rdf:about="&DesignationSchemeValue;natura2000"/>
                  <rdf:Description rdf:about="&DesignationSchemeValue;UNESCOWorldHeritage"/>
                    ...
              </owl:oneOf>
```

```
            </owl:Class>
        </rdfs:subClassOf>
    </owl:Class>

<!-- OWL:NAMEDINDIVIDUAL DesignationSchemeValue:natura2000 -->

    <skos:Concept rdf:about="&DesignationSchemeValue;natura2000">
        <rdf:type rdf:resource="&ps;DesignationSchemeValue"/>
        <rdf:type rdf:resource="&owl;NamedIndividual"/>
        <rdfs:label>Natura 2000</rdfs:label>
        <dc:description>The Protected Site has a designation under either the Habitat
          Directive (92/43/EEC) or the Birds Directive (79/409/EEC) ... section 5.2.2.3.2</dc:description>
        <skos:inScheme rdf:resource="&base;CodelistRegister"/>
        <skos:broader rdf:resource="&ps;DesignationSchemeValue"/>
    </skos:Concept>

<!-- OWL:NAMEDINDIVIDUAL DesignationSchemeValue:UNESCOWorldHeritage -->

    <skos:Concept rdf:about="&DesignationSchemeValue;UNESCOWorldHeritage">
        <rdf:type rdf:resource="&ps;DesignationSchemeValue"/>
        <rdf:type rdf:resource="&owl;NamedIndividual"/>
        <rdfs:label>UNESCO world heritage</rdfs:label>
        <dc:description>The Protected Site has a designation under UNESCO World Heritage
           ... section 5.2.2.3.2</dc:description>
        <skos:inScheme rdf:resource="&base;CodelistRegister"/>
        <skos:broader rdf:resource="&ps;DesignationSchemeValue"/>
    </skos:Concept>

<!-- OWL:CLASS ps:DesignationValue -->

    <owl:Class rdf:about="&ps;DesignationValue">
        <rdfs:label>designation value</rdfs:label>
        <dc:description>Abstract base type for code lists containing the classificaiton and desigation types
          under different schemes ... INSPIRE Data Specification Protected Sites section 5.2.2.3.3</dc:description>
        <skos:inScheme rdf:resource="&base;CodelistRegister"/>
        <cf:entityType>codeList</cf:entityType>
    </owl:Class>

<!-- OWL:CLASS ps:Natura2000DesignationValue -->

    <owl:Class rdf:about="&ps;Natura2000DesignationValue">
        <rdfs:label>Natura 2000 designation value</rdfs:label>
        <dc:description>A code list for the Natura2000 designation scheme ... section 5.2.2.3.6</dc:description>
        <skos:inScheme rdf:resource="&base;CodelistRegister"/>
        <skos:broader rdf:resource="&ps;DesignationValue"/>
        <cf:entityType>codeList</cf:entityType>
        <rdfs:subClassOf rdf:resource="&ps;DesignationValue"/>
        <rdfs:subClassOf>
            <owl:Class>
                <owl:oneOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="&Natura2000DesignationValue;specialAreaOfConservation"/>
                    <rdf:Description rdf:about="&Natura2000DesignationValue;proposedSpecialProtectionArea"/>
                      ...
                </owl:oneOf>
            </owl:Class>
        </rdfs:subClassOf>
    </owl:Class>

<!-- OWL:NAMEDINDIVIDUAL Natura2000DesignationValue:specialAreaOfConservation -->
```

```
  <skos:Concept rdf:about="&Natura2000DesignationValue;specialAreaOfConservation">
      <rdf:type rdf:resource="&ps;Natura2000DesignationValue"/>
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdfs:label>special area of conservation</rdfs:label>
      <dc:description>... designated as a Special Area of Conservation (SAC) under Natura2000
         ... section 5.2.2.3.6</dc:description>
      <skos:inScheme rdf:resource="&base;CodelistRegister"/>
      <skos:broader rdf:resource="&ps;Natura2000DesignationValue"/>
  </skos:Concept>

<!-- OWL:NAMEDINDIVIDUAL Natura2000DesignationValue:proposedSpecialProtectionArea -->

  <skos:Concept rdf:about="&Natura2000DesignationValue;proposedSpecialProtectionArea">
      <rdf:type rdf:resource="&ps;Natura2000DesignationValue"/>
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdfs:label>proposed special protection area</rdfs:label>
      <dc:description>... proposed as a Special Protection Area (SPA) under Natura2000
        ... section 5.2.2.3.6</dc:description>
      <skos:inScheme rdf:resource="&base;CodelistRegister"/>
      <skos:broader rdf:resource="&ps;Natura2000DesignationValue"/>
  </skos:Concept>

<!-- OWL:CLASS ps:ProtectionClassificationValue -->

  <owl:Class rdf:about="&ps;ProtectionClassificationValue">
      <rdfs:label>protection classfication value</rdfs:label>
      <dc:description>The protected site classification based on the purpose ... section 5.2.2.3.1</dc:description>
      <skos:inScheme rdf:resource="&base;CodelistRegister"/>
      <cf:entityType>enumeration</cf:entityType>
      <rdfs:subClassOf>
          <owl:Class>
              <owl:oneOf rdf:parseType="Collection">
                  <rdf:Description rdf:about="&ProtectionClassificationValue;landscape"/>
                    ...
              </owl:oneOf>
          </owl:Class>
      </rdfs:subClassOf>
  </owl:Class>

<!-- OWL:NAMEDINDIVIDUAL ProtectionClassificationValue:landscape -->

  <skos:Concept rdf:about="&ProtectionClassificationValue;landscape">
      <rdf:type rdf:resource="&ps;ProtectionClassificationValue"/>
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdfs:label>landscape</rdfs:label>
      <dc:description>... protected for the maintenance of landscape characteristics
        ... section 5.2.2.3.1</dc:description>
      <skos:inScheme rdf:resource="&base;CodelistRegister"/>
      <skos:broader rdf:resource="&ps;ProtectionClassificationValue"/>
  </skos:Concept>

<!-- OWL:NAMEDINDIVIDUAL base:CrsUomRegister -->

  <skos:ConceptScheme rdf:about="&base;CrsUomRegister">
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdfs:label>CRS/ UoM Register</rdfs:label>
      <skos:prefLabel>CRS/ UoM Register</skos:prefLabel>
      <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.5_v3_3.pdf"/>
```

```
  </skos:ConceptScheme>

<!-- OWL:NAMEDINDIVIDUAL base:CrsUomRegister/uom/area/hectare -->

  <skos:Concept rdf:about="&base;CrsUomRegister/uom/area/hectare">
      <rdf:type rdf:resource="&muo;SimpleDerivedUnit"/>
      <rdf:type rdf:resource="&owl;NamedIndividual"/>
      <rdf:type rdf:resource="&base;UnitOfMeasurement"/>
      <rdfs:label>hectare</rdfs:label>
      <skos:prefLabel>hectare</skos:prefLabel>
      <skos:inScheme rdf:resource="&base;CrsUomRegister"/>
      <muo:dimensionalSize rdf:datatype="&xsd;float">2</muo:dimensionalSize>
      <muo:derivesFrom rdf:resource="&base;CrsUomRegister/uom/length/meter"/>
      <muo:measuresQuality rdf:resource="&physical-quality;area"/>
      <muo:modifierPrefix rdf:resource="&ucumPrefix;hecto"/>
      <owl:sameAs rdf:resource="&UNECE;HAR"/>
  </skos:Concept>

<!-- OWL:CLASS base:Measure -->

  <owl:Class rdf:about="&base;Measure">
      <rdfs:label>measure</rdfs:label>
      <dc:description>The value of a physical quantity, together with its unit. See ISO 19136 17.3.
        gml:MeasureType is defined in the basicTypes schema. The measure types defined here correspond with
        a set of convenience measure types described in ISO/TS 19103</dc:description>
      <cf:entityType>measure</cf:entityType>
      <rdfs:subClassOf rdf:resource="&muo;QualityValue"/>
      <rdfs:subClassOf>
          <owl:Class>
              <owl:intersectionOf rdf:parseType="Collection">
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;unitOfMeasure"/>
                      <owl:allValuesFrom rdf:resource="&base;UnitOfMeasure"/>
                  </owl:Restriction>
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;unitOfMeasure"/>
                      <owl:onDataRange rdf:resource="&base;UnitOfMeasure"/>
                      <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
                  </owl:Restriction>
              </owl:intersectionOf>
          </owl:Class>
      </rdfs:subClassOf>
      <rdfs:subClassOf>
          <owl:Class>
              <owl:intersectionOf rdf:parseType="Collection">
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;numericalValue"/>
                      <owl:allValuesFrom rdf:resource="&xsd;decimal"/>
                  </owl:Restriction>
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;numericalValue"/>
                      <owl:onDataRange rdf:resource="&xsd;decimal"/>
                      <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
                  </owl:Restriction>
              </owl:intersectionOf>
          </owl:Class>
      </rdfs:subClassOf>
  </owl:Class>
```

```
<!-- OWL:CLASS base:Area -->

  <owl:Class rdf:about="&base;Area">
      <rdfs:label>area</rdfs:label>
      <dc:description>gml:AreaType as a subclass of gml:MeasureType, see ISO 19136 17.3.</dc:description>
      <cf:entityType>measure</cf:entityType>
      <rdfs:subClassOf rdf:resource="&base;Measure"/>
  </owl:Class>

<!-- OWL:OBJECTPROPERTY base:unitOfMeasure -->

  <owl:ObjectProperty rdf:about="&base;unitOfMeasure">
      <rdfs:label>unit of measure</rdfs:label>
      <dc:description>unit of measure identifier. ISO 19136 for xml serialization: BasicTypes.xsd,
        &quot;uom&quot;-attribute of complex-element &quot;MeasureType&quot;</dc:description>
      <rdfs:domain rdf:resource="&base;Measure"/>
      <rdfs:range rdf:resource="&base;UnitOfMeasure"/>
      <rdfs:subPropertyOf rdf:resource="&muo;measuredIn"/>
  </owl:ObjectProperty>

<!-- OWL:CLASS base:UnitOfMeasure -->

  <owl:Class rdf:about="&base;UnitOfMeasure">
      <rdfs:label>UnitOfMeasure</rdfs:label>
      <rdfs:subClassOf rdf:resource="&muo;UnitOfMeasure"/>
  </owl:Class>

<!-- OWL:DATATYPEPROPERTY base:numericalValue -->

  <owl:DatatypeProperty rdf:about="&base;numericalValue">
      <rdfs:label>numerical value</rdfs:label>
      <dc:description>represents the numerical value of a &quot;Measure&quot;-datatype</dc:description>
      <rdfs:domain rdf:resource="&base;Measure"/>
      <rdfs:range rdf:resource="&xsd;decimal"/>
      <rdfs:subPropertyOf rdf:resource="&muo;numericalValue"/>
  </owl:DatatypeProperty>

<!-- OWL:CLASS base:Identifier -->

  <owl:Class rdf:about="&base;Identifier">
      <rdfs:label>INSPIRE identifier</rdfs:label>
      <dc:description>External unique object identifier published by the responsible body,
        which may be used by external applications to reference the spatial object.
        INSPIRE Generic Conceptual Model section 9.8.3.1</dc:description>
      <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/
        Data_Specifications/D2.5_v3_3.pdf"/>
      <cf:entityType>dataType</cf:entityType>
      <cf:xmlName>base:Identifier</cf:xmlName>
      <rdfs:subClassOf>
          <owl:Class>
              <owl:intersectionOf rdf:parseType="Collection">
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;localId"/>
                      <owl:allValuesFrom rdf:resource="&rdfs;Literal"/>
                  </owl:Restriction>
                  <owl:Restriction>
                      <owl:onProperty rdf:resource="&base;localId"/>
                      <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
                  </owl:Restriction>
```

```
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&base;namespace"/>
                    <owl:allValuesFrom rdf:resource="&rdfs;Literal"/>
                </owl:Restriction>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="&base;namespace"/>
                    <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="&base;versionId"/>
            <owl:allValuesFrom rdf:resource="&rdfs;Literal"/>
        </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

<!-- OWL:DATATYPEPROPERTY base:namespace -->

  <owl:DatatypeProperty rdf:about="&base;namespace">
      <rdfs:label>namespace</rdfs:label>
      <dc:description> ... uniquely identifying the data source of the spatial object
        ... section 9.8.3.1</dc:description>
      <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/
        Data_Specifications/D2.5_v3_3.pdf"/>
      <cf:xpath>base:Identifier/base:namespace</cf:xpath>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&base;Identifier"/>
      <cf:rdfsRange rdf:resource="&rdf;PlainLiteral"/>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY base:localId -->

  <owl:DatatypeProperty rdf:about="&base;localId">
      <rdfs:label>local id</rdfs:label>
      <dc:description>A local identifier, assigned by the data provider ... section 9.8.3.1</dc:description>
      <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/
        Data_Specifications/D2.5_v3_3.pdf"/>
      <cf:xpath>base:Identifier/base:localId</cf:xpath>
  </owl:DatatypeProperty>
  <owl:Axiom>
        ...
      <cf:rdfsDomain rdf:resource="&base;Identifier"/>
      <cf:rdfsRange rdf:resource="&rdf;PlainLiteral"/>
  </owl:Axiom>

<!-- OWL:DATATYPEPROPERTY base:versionId -->

  <owl:DatatypeProperty rdf:about="&base;versionId">
```

```
        <rdfs:label>version id</rdfs:label>
        <dc:description>The identifier of the particular version of the spatial object
          ... section 9.8.3.1</dc:description>
        <rdfs:isDefinedBy rdf:resource="http://inspire.jrc.ec.europa.eu/documents/
          Data_Specifications/D2.5_v3_3.pdf"/>
        <cf:xpath>base:Identifier/base:versionId</cf:xpath>
    </owl:DatatypeProperty>
    <owl:Axiom>
          ...
        <cf:rdfsDomain rdf:resource="&base;Identifier"/>
        <cf:rdfsRange rdf:resource="&rdf;PlainLiteral"/>
    </owl:Axiom>

  <!-- OWL:CLASS br:Bio-geographicalRegion -->

    <owl:Class rdf:about="&br;Bio-geographicalRegion">
        <rdfs:label>Bio-geographicalRegion</rdfs:label>
        <dc:description>An area of relatively homogeneous ecological conditions
          ... section 5.2.2.1.1</dc:description>
        <cf:entityType>featureType</cf:entityType>
        <cf:uriSyntax>http://inspire.jrc.ec.europa.eu/data/{br:inspireId/base:Identifier/base:namespace}/
          {br:inspireId/base:Identifier/base:localId}[/{br:inspireId/base:Identifier/base:versionId}]</cf:uriSyntax>
        <cf:xpath>br:Bio-geographicalRegion</cf:xpath>
        <rdfs:subClassOf rdf:resource="&geo;Feature"/>
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&br;regionClassification"/>
                <owl:allValuesFrom rdf:resource="&br;RegionClassificationValue"/>
            </owl:Restriction>
        </rdfs:subClassOf>
          ...
    </owl:Class>

</rdf:RDF>
```

Anhang D
Konfiguration eines Ressourcen-Repository

Dargestellt ist eine beispielhafte Konfiguration in der Turtle-Syntax. Das Ressourcen-Repository deklariert einen Downloaddienst (Zeile 5-10), der auf zwei Annex-Themen operiert (Zeilen 12-26 bzw. 28-40). Die Konfiguration des Dienstes und des ersten Themas entsprechen den Instanzeninformationen des UML-Diagramms in Abbildung 5.12.

```
1  @prefix cf: <http://inspire.west.uni-koblenz.de/configuration/> .
2  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3  @prefix inspirethemes: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/> .
4
5  _:repository1 rdf:type cf:DownloadService ;
6    cf:name "Testserver INSPIRE-Downloaddienst" ;
7    cf:serviceUrl "http://inspire.west.uni-koblenz.de:8080/deegree/services" ;
8    cf:serviceType "WFS_1.1" ;
9    cf:theme _:theme1 ;
10   cf:theme _:theme2 .
11
12 _:theme1 rdf:type cf:AnnexTheme ;
13   cf:name "Internationale slovakische Schutzgebiete" ;
14   cf:description "Internationale slovakische Schutzgebiete der Klassifikationen NDA,
15     Natura2000 SCI und Ramsar" ;
16   cf:inspireName "ProtectedSites" ;
17   cf:inspireNamespace "SK_PS_NDA" ;
18   cf:inspireNamespace "SK_PS_SCI" ;
19   cf:inspireNamespace "SK_PS_Ramsar" ;
20   cf:ontologyNamespace inspirethemes:ProtectedSites ;
21   cf:hasSpatialExtent [
22     cf:minLat  47.45 ;
23     cf:minLong  16.45 ;
24     cf:maxLat  49.5 ;
25     cf:maxLong  22.5
26   ] .
27
28 _:theme2 rdf:type cf:AnnexTheme ;
29   cf:name "Bio-geographische Regionen" ;
30   cf:description "Testdatensatz zu ausgewählten europäischen Regionen,
31     Quelldatensatz 'Biogeographische Regionen, Europa 2001' der EEA" ;
32   cf:inspireName "Bio-geographicalRegions" ;
33   cf:inspireNamespace "EU.BGR" ;
34   cf:ontologyNamespace inspirethemes:Bio-geographicalRegions ;
35   cf:hasSpatialExtent [
36     cf:minLat  40.73 ;
37     cf:minLong  -9.29 ;
38     cf:maxLat  57.62 ;
39     cf:maxLong  57.79
40   ] .
```

Anhang E
Anfragen und Resultate aus den Testszenarien

Die Georeferenzierung der Geometriedaten in den OGC GetFeature-Anfragen ist EPSG:4326 (WGS84). Deren Koordinatenreihenfolge ist Long/Lat und entspricht damit nicht der OGP-Definition für das WGS84 (Lat/Long)[169], jedoch ist die hier gewählte *falsche* Koordinatenreihenfolge gängige Praxis und wurde auch vom verwendeten WFS-Dienst *Deegree3 inspireNode* vorausgesetzt.

Testszenario I

Originäre SPARQL-Anfrage:

```
PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/>
PREFIX geof: <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/>
PREFIX sf10: <http://www.opengis.net/def/dataType/OGC-SF/1.0/>

SELECT ?schutzgebietName ?designation ?agencyName
WHERE
{
  ?schutzgebiet a ont_ps:ProtectedSite .
  ?schutzgebiet ont_ps:siteName ?schutzgebietName .
  ?schutzgebiet ont_ps:siteDesignation ?designationContext .
  ?designationContext ont_ps:designation ?designation .
  ?schutzgebiet ont_ps:geometry ?geometry .
  ?geometry geo:asLiteral ?geomLiteral .
  FILTER( geof:contains( ?geomLiteral, "<http://www.opengis.net/def/crs/epsg/0/4326>
    POINT(8.91 49.39)"^^sf10:WKTLiteral) )

  OPTIONAL{
    ?schutzgebiet ont_ps:isManagedBy ?responsibleAgency .
    ?responsibleAgency ont_ps:responsibleAgencyName ?agencyName .
  }
}
```

Aus der SPARQL-Anfrage generierte WFS GetFeature-Anfrage:

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
  xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0" xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
  xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0" version="1.1.0" resultType="results"
  outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">

  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
```

[169] siehe im EPSG-Register unter: http://www.epsg-registry.org/

```
      typeName="ps-f:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:And>
        <ogc:Contains xmlns:gml="http://www.opengis.net/gml">
          <ogc:PropertyName>ps:geometry</ogc:PropertyName>
          <gml:Point srsName="EPSG:4326">
            <gml:pos>8.91 49.39</gml:pos>
          </gml:Point>
        </ogc:Contains>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:siteName/gn:GeographicalName/gn:spelling/gn:SpellingOfName/gn:text
            </ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:siteDesignation/ps:DesignationType</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:siteDesignation/ps:DesignationType/ps:designation</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:geometry</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
      </ogc:And>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
      typeName="ps:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">

      <!-- Filterung entspricht der des Feature Types "ps-f:ProtectedSite" -->

    </ogc:Filter>
  </wfs:Query>

</wfs:GetFeature>
```

SPARQL-Anfragefenster und Ergebnisdarstellung im SPARQL-Client Twinkle:

siehe Abbildung 5.19, eingefügt im Abschnitt 5.3.4

Testszenario II

Originäre SPARQL-Anfrage:

siehe Codebeispiel 22, eingefügt im Abschnitt 5.3.4

Aus der SPARQL-Anfrage generierte WFS GetFeature-Anfrage:

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
  xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0" xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2"
  xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0" version="1.1.0" resultType="results"
  outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">


  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
      typeName="ps-f:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:And>
        <ogc:PropertyIsEqualTo matchCase="false">
          <ogc:PropertyName>ps:inspireID/base:Identifier/base:localId</ogc:PropertyName>
          <ogc:Literal>67_919014000001</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo matchCase="false">
          <ogc:PropertyName>ps:inspireID/base:Identifier/base:namespace</ogc:PropertyName>
          <ogc:Literal>DE.BW.PS</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:inspireID/base:Identifier</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
      </ogc:And>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
      typeName="ps:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">

      <!-- Filterung entspricht der des Feature Types "ps-f:ProtectedSite" -->

    </ogc:Filter>
  </wfs:Query>

</wfs:GetFeature>
```

SPARQL-Anfragefenster und Ergebnisdarstellung im SPARQL-Client Twinkle:

siehe Abbildung 5.20, eingefügt im Abschnitt 5.3.4

Testszenario III a) - Suche über statische Ressourcenverlinkungen

Originäre SPARQL-Anfrage:

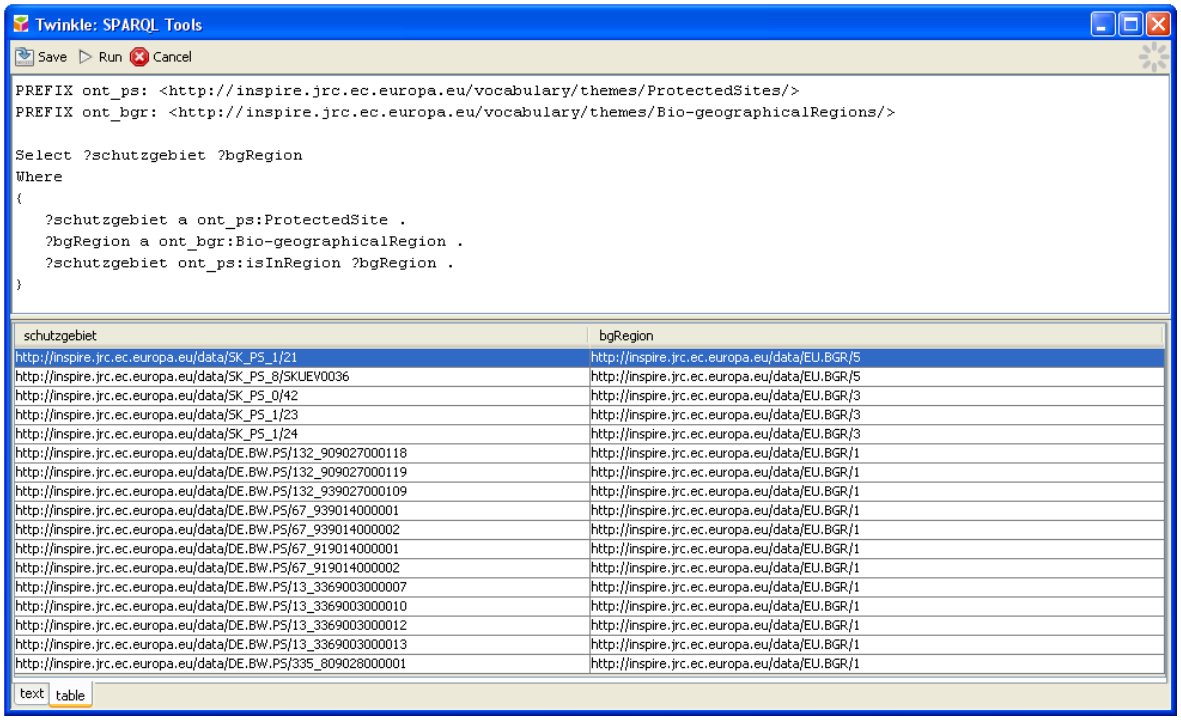
```
PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
PREFIX ont_bgr: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions/>

SELECT ?schutzgebiet ?bgRegion
WHERE
{
```

```
  ?schutzgebiet a ont_ps:ProtectedSite .
  ?bgRegion a ont_bgr:Bio-geographicalRegion .
  ?schutzgebiet ont_ps:isInRegion ?bgRegion .
}
```

Aus der SPARQL-Anfrage generierte WFS GetFeature-Anfrage:

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
  xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2" xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
  xmlns:br="http://inspire.jrc.ec.europa.eu/schemas/br/2.0" version="1.1.0" resultType="results"
  outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">

  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
      typeName="ps-f:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>ps-f:isInRegion/br:Bio-geographicalRegion</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:br="http://inspire.jrc.ec.europa.eu/schemas/br/2.0"
      typeName="br:Bio-geographicalRegion" srsName="EPSG:4326"/>

</wfs:GetFeature>
```

SPARQL-Anfragefenster und Ergebnisdarstellung im SPARQL-Client Twinkle:



Testszenario III b) - Suche über temporäre Ressourcenverlinkungen

Originäre SPARQL-Anfrage:

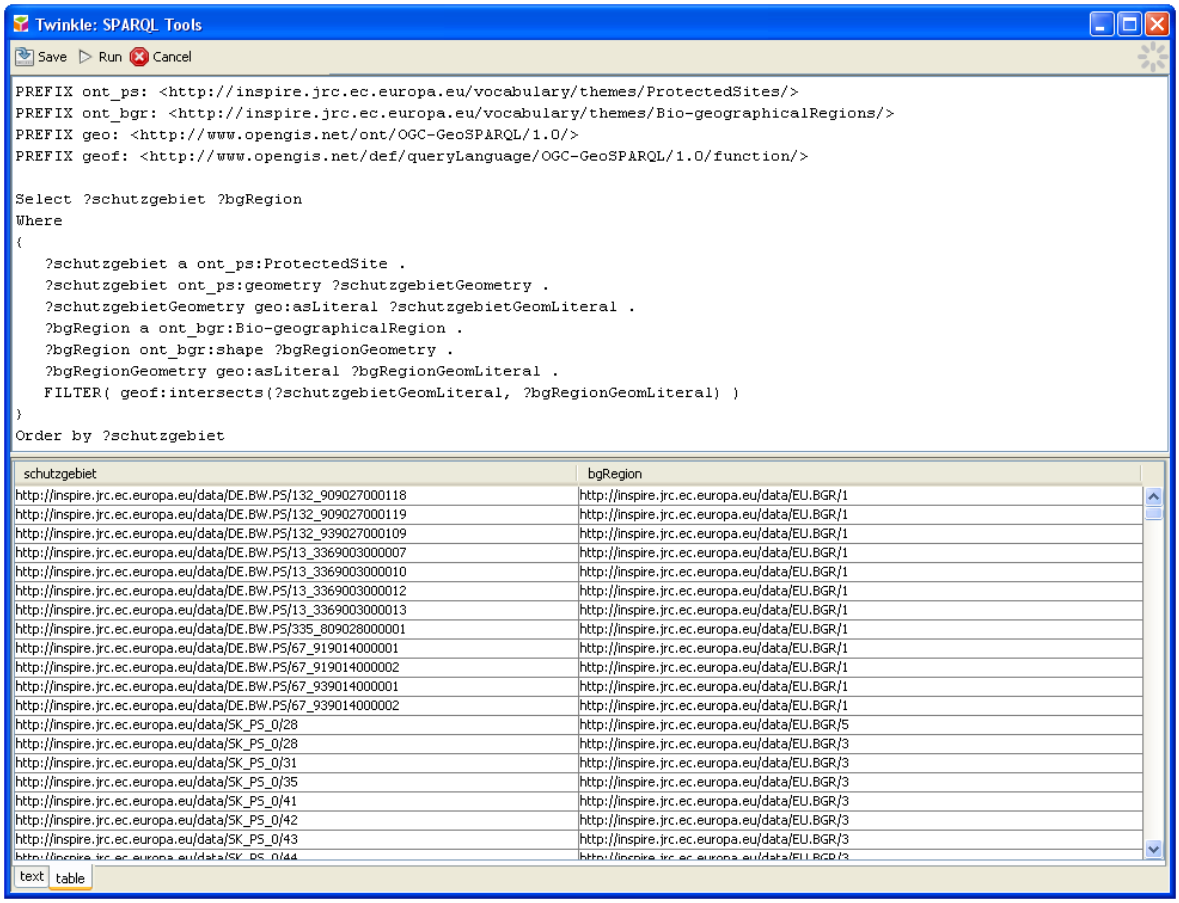
```
PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
PREFIX ont_bgr: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/Bio-geographicalRegions/>
PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/>
PREFIX geof: <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/>

SELECT ?schutzgebiet ?bgRegion
WHERE
{
  ?schutzgebiet a ont_ps:ProtectedSite .
  ?schutzgebiet ont_ps:geometry ?schutzgebietGeometry .
  ?schutzgebietGeometry geo:asLiteral ?schutzgebietGeomLiteral .
  ?bgRegion a ont_bgr:Bio-geographicalRegion .
  ?bgRegion ont_bgr:shape ?bgRegionGeometry .
  ?bgRegionGeometry geo:asLiteral ?bgRegionGeomLiteral .

  FILTER( geof:intersects( ?schutzgebietGeomLiteral, ?bgRegionGeomLiteral) )
}
ORDER BY ?schutzgebiet
```

Aus der SPARQL-Anfrage generierte WFS GetFeature-Anfrage:

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
  xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2" xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
  xmlns:br="http://inspire.jrc.ec.europa.eu/schemas/br/2.0" version="1.1.0" resultType="results"
  outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">

  <wfs:Query xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
      typeName="ps:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>ps:geometry</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
      typeName="ps-f:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>ps:geometry</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:br="http://inspire.jrc.ec.europa.eu/schemas/br/2.0"
      typeName="br:Bio-geographicalRegion" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
```

```
      <ogc:Not>
        <ogc:PropertyIsNull>
          <ogc:PropertyName>br:shape</ogc:PropertyName>
        </ogc:PropertyIsNull>
      </ogc:Not>
    </ogc:Filter>
  </wfs:Query>
</wfs:GetFeature>
```

SPARQL-Anfragefenster und Ergebnisdarstellung im SPARQL-Client Twinkle:



Testszenario IV

Originäre SPARQL-Anfrage:

```
PREFIX ont_ps: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/>
PREFIX iucnCode: <http://inspire.jrc.ec.europa.eu/vocabulary/themes/ProtectedSites/IUCNDesignationValue/>
PREFIX geo: <http://www.opengis.net/ont/OGC-GeoSPARQL/1.0/>
PREFIX geof: <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/>
PREFIX sf10: <http://www.opengis.net/def/dataType/OGC-SF/1.0/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>
Prefix lgdo: <http://linkedgeodata.org/ontology/>

SELECT ?schoolName ?schutzgebietName ?hostelName
WHERE
{
```

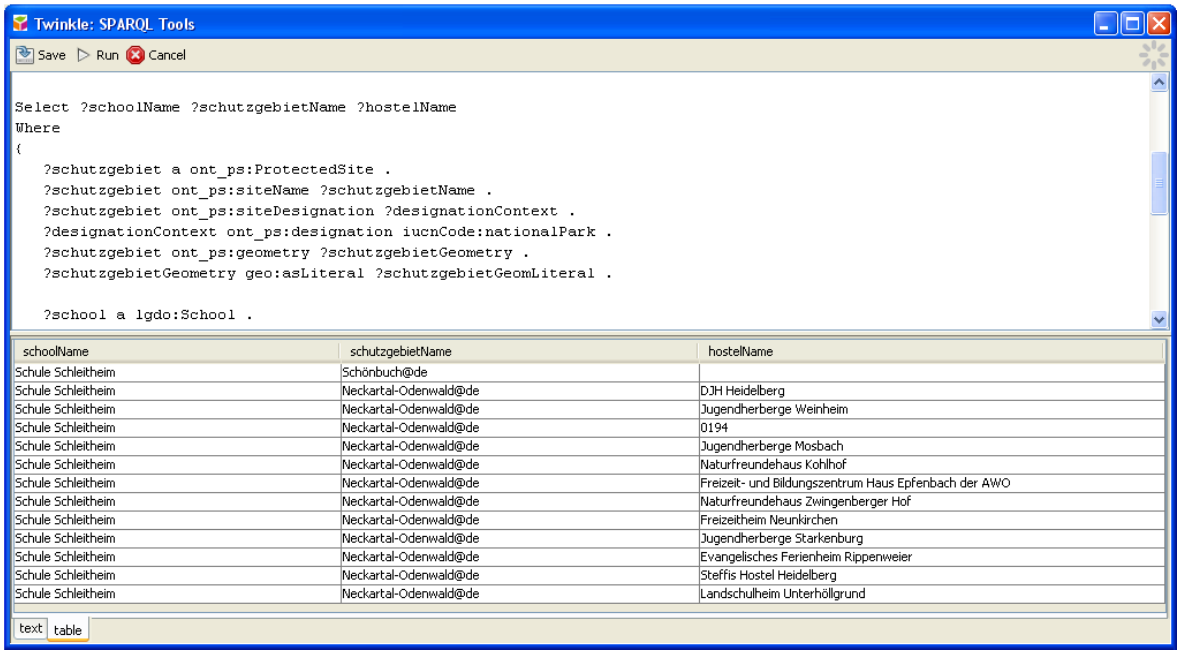
```
  ?schutzgebiet a ont_ps:ProtectedSite .
  ?schutzgebiet ont_ps:siteName ?schutzgebietName .
  ?schutzgebiet ont_ps:siteDesignation ?designationContext .
  ?designationContext ont_ps:designation iucnCode:nationalPark .
  ?schutzgebiet ont_ps:geometry ?schutzgebietGeometry .
  ?schutzgebietGeometry geo:asLiteral ?schutzgebietGeomLiteral .

  ?school a lgdo:School .
  ?school rdfs:label ?schoolName .
  ?school pos:lat ?schoolLat .
  ?school pos:long ?schoolLong .

  FILTER( regex(str(?schoolName), ".*Schleitheim.*" )
    && geof:disjoint(?schutzgebietGeomLiteral, geof:buffer(?schoolLong, ?schoolLat, 50000,
      <http://inspire.jrc.ec.europa.eu/vocabulary/base/CrsUomRegister/uom/length/meter>) ) )

  OPTIONAL{
    ?hostel a lgdo:Hostel .
    ?hostel rdfs:label ?hostelName .
    ?hostel pos:lat ?hostelLat .
    ?hostel pos:long ?hostelLong .

    FILTER( geof:intersects(?schutzgebietGeomLiteral, geof:buffer(?hostelLong, ?hostelLat, 3000,
      <http://inspire.jrc.ec.europa.eu/vocabulary/base/CrsUomRegister/uom/length/meter>) ) )
  }
}
```

Aus der SPARQL-Anfrage generierte WFS GetFeature-Anfrage:

```
<?xml version="1.0"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd"
  xmlns:gn="urn:x-inspire:specification:gmlas:GeographicalNames:3.0"
  xmlns:base="urn:x-inspire:specification:gmlas:BaseTypes:3.2" xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
  xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
  xmlns:br="http://inspire.jrc.ec.europa.eu/schemas/br/2.0" version="1.1.0"
  resultType="results" outputFormat="text/xml; subtype=gml/3.2.1" traverseXlinkDepth="*">

  <wfs:Query xmlns:ps-f="urn:x-inspire:specification:gmlas:ProtectedSitesFull:3.0"
      typeName="ps-f:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:And>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:siteName/gn:GeographicalName/gn:spelling/gn:SpellingOfName/gn:text
            </ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:Not>
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:siteDesignation/ps:DesignationType</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
        <ogc:PropertyIsEqualTo matchCase="false">
          <ogc:PropertyName>ps:siteDesignation/ps:DesignationType/ps:designation</ogc:PropertyName>
          <ogc:Literal>nationalPark</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:Not>
```

```
          <ogc:PropertyIsNull>
            <ogc:PropertyName>ps:geometry</ogc:PropertyName>
          </ogc:PropertyIsNull>
        </ogc:Not>
      </ogc:And>
    </ogc:Filter>
  </wfs:Query>

  <wfs:Query xmlns:ps="urn:x-inspire:specification:gmlas:ProtectedSites:3.0"
      typeName="ps:ProtectedSite" srsName="EPSG:4326">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">

      <!-- Filterung entspricht der des Feature Types "ps-f:ProtectedSite" -->

    </ogc:Filter>
  </wfs:Query>

</wfs:GetFeature>
```

SPARQL-Anfragefenster und Ergebnisdarstellung im SPARQL-Client Twinkle:



Anhang F
Kommentierung des GeoSPARQL-RfC

Der beim OGC erarbeitete GeoSPARQL-Standardisierungsentwurf (OGC 11-052r3) [Perry & Herring 2011] wurde im Juli 2011 veröffentlicht und zur Kommentierung freigegeben. Innerhalb der vorgegebenen Monatsfrist wurden unten stehende Kommentare an das OGC gesandt, die inhaltlich den Resultaten der GeoSPARQL-Überprüfung im Rahmen dieser Arbeit entsprechen. Dank der OGC-Mitgliedschaft und dem Beobachterstatus in der GeoSPARQL-Standard Working Group war es dem Autor möglich, noch vor der Veröffentlichung die interne Entwicklungsarbeit zu verfolgen, so dass eine umfangreiche Evaluation (siehe Abschnitte 2.4.3, 5.1.4 und 5.2.5) vor bzw. bei Erscheinen des GeoSPARQL-Entwurfes durchgeführt werden konnte.

1. **Requirement no.**: -
   Implementation Specification Section no.: General, most likely: 2
   Criticality: Major correction
   Comment: It seems that the terms 'Core' and 'Extension' hints at mandatory and optional parts for a GeoSPARQL-interface realization. But GeoSPARQL-interfaces won't be useful without at least one particular parametrized conformance class for Geometry Extension and maybe one for Geometry Topology Extension. So the specification lacks declaring at least the minimal configuration for a GeoSPARQL-implementation.

2. **Requirement no.**: 4/6/8
   Implementation Specification Section no.: 7.2/7.3/7.4
   Criticality: Minor correction
   Comment: It's not clear enough in which way these requirements differ from the usual SPARQL-result expectations (the dependency to SPARQL is already declared in Requirement 1). The GeoSPARQL topology vocabulary introduces these statically stored predicates but the assessment via a SPARQL-query is like with any other RDF predicates.

3. **Requirement no.**: 5/7/9
   Implementation Specification Section no.: 7.2/7.3/7.4
   Criticality: Editorial correction
   Comment: Typing error in the requirement texts, should be rdf:Propertyïnstead of rdf:Propery"

4. **Requirement no.**: -
   Implementation Specification Section no.: 8.4
   Criticality: Addition of feature
   Comment: Proposal for another standard property for 'geo:Geometry' which is called 'geo:captureResolution'. This property could transport a minimal information about geometrical data quality, could be used for appropriate visualization resolutions or help an (external) user interpret the data accuracy before data (re)usage.

5. **Requirement no.**: 12
   Implementation Specification Section no.: 8.4.4
   Criticality: Minor correction
   Comment: The predicate 'geo:isEmpty' is rather useless in the Semantic Web context. For relational geo-databases this information is captured due to the fact that empty field values could be expressed this way. In the Semantic Web the predicates are usually not set then. Furthermore a GeoSPARQL-conformant dataset with a Feature-instance relating to a Geometry-instance attributed with „my:geometry geo:isEmpty 'true'8sd:boolean“ does not make sense.

6. **Requirement no.**: 12
   Implementation Specification Section no.: 8.4.6
   Criticality: Minor correction
   Comment: The information of predicate 'geo:is3D' is already contained within predicate 'geo:coordinateDimension', so 'geo:is3D' could be omitted.

7. **Requirement no.**: -
   Implementation Specification Section no.: General, most likely: 8
   Criticality: Addition of feature
   Comment: For the reason of backward compability to well-known geographical information in Semantic Web databases, a simple storage form is proposed according to CRS WGS84 (the least common denominator). Besides, the most important geometry types are points (for social tagging/labeling) and bounding boxes/ envelopes. Every geo:Geometry instance could comprise therefore semantically unique predicates, for a reference location stored as a point: 'geo:refLocation_min', 'geo:refLocation_max' and for bbox-coordinates with: 'geo:bbox_minLat', 'geo:bbox_minLong', 'geo:bbox_maxLat', 'geo:bbox_maxLong'. These separated coordinates (not tuples or sequences of tuples as in WKTLiteral or GMLLiteral) could be derived out of the individual GeomLiteral and be easily compared with any conventionally encoded geographical information from LinkedGeoData, DBPedia, GeoNames etc. using W3C Basic Geo or other vocabularies. The main reason for this proposal is that GeoSPARQL vocabulary conformant datasets are then prepared for comparisons with plain old SPARQL functions and so are not dependent only to GeoSPARQL processing which leads to no gap between different domains in the meanwhile (GeoSPARQL-support or not). Overloading of GeoSPARQL-funtions with 'xsd:decimal' types for individual coordinates would enable GeoSPARQL-funtions be capable to handle both, the new complex geometry literals and the conventionally stored coordinate predicates, too.

8. **Requirement no.**: -
   Implementation Specification Section no.: 8 and 9 (topolical/non-topological functions)
   Criticality: Functional modification of feature
   Comment: The specification should clarify if an implementation with support for two serializations (e.g. GML 3.2.1 and WKT 1.0) and for Geometry Topology Extension must accept different serialized geometry literals in one topolical/non-topological function-call. Section 8.7 indicates possible errors but why should the user be constraint to apply a function only for one serialization type? In case of integration tasks (two or more datasets each with different serialization types) there would be a real hurdle otherwise.

9. **Requirement no.**: -
   Implementation Specification Section no.: 9
   Criticality: Major correction
   Comment: The redundancy between many topological functions (specially between Egenhofer and their Simple Features adoption with some new names) seems to be too sophisticated for this specification; consider Semantic Web users so far not affected by OGC specifications or geographical concerns. A handful of functions as provided by Simple Features with the ability to express every topological situation with function 'geo:relate' should work as well. If done so, there is no need for the 'relation_family' parameter and the function names could be shorter without confusing prefixes 'rcc8', 'sf' and 'eh'.

10. **Requirement no.**: -
    Implementation Specification Section no.: 10
    Criticality: Minor correction
    Comment: The conformance class 'Query Rewrite Extension' has three parameters 'relation_family', 'serialization', 'version'. The last two of them are not relevant for the rewriting functionality, temporary evaluations of serialization matters during run-time are not worthful for any query rewriting process. Parametrizing the 'Query Rewrite Extension' with 'serialization' and 'version' will overload/complicate the implementation documentation without any positive effect. The extension should be self-contained and optional and only support the parameter 'relation_family'.

11. **Requirement no.**: -
    Implementation Specification Section no.: General
    Criticality: General consideration
    Comment: The possibly numerous conformance classes due to many different serialization types and versions could emerge as a big disadvantage of GeoSPARQL. For example the usage of predicates 'geo:asGML', 'geo:asWKT' is not really convenient (the user will certainly expect something like a unique 'geo:geomLiteral') and their introduction are a major concession for future implementations which will not support so many geometry serializations. But then there must be e.g. a 'geo:asGML321', 'geo:asGML300', 'geo:asGML210' and so on, because the differentiation is used by the GeoSPARQL-functions. Besides WKT and GML, which other serializations are really promising? KML and GeoRSS with its geometry types deeply related to GML; GeoJSON is characterized like WKT as a plain text form, maybe better encoded than WKT, but with a subset of WKT geometry types. So WKT and GML are the only two promising serializations, why not constrain GeoSPARQL to these two and facilitate software products to get most features or all conformance classes implemented.

12. **Requirement no.**: -
    Implementation Specification Section no.: General
    Criticality: Addition of feature
    Comment: Besides directly stored topological predicates between 'geo:SpatialObject' instances (with Topology Vocabulary Extension) also two inverse functional predicates are proposed between 'geo:Geometry' instances: 'geo:generalizes' and 'geo:generalizedBy' for generalization hints. Another predicate could be derived from geo:hasGeometry and could be called 'geo:hasOriginalGeometry' to indicate the primary geometry which is just captured, corrected and therefore with the highest resolution under all 'geo:Geometry' instances for one 'geo:Feature' instance.