RSW GEOMATICS LTD

1Spatial

Rob Walker Consultancy

Development of Technical Guidance for the INSPIRE Transformation Network Service

EC JRC Contract Notice 2009/S 107-153973

"State Of The Art Analysis"

Authors: Matthew Beare, Mark Howard, Simon Payne, Paul Watson

Editor: Rob Walker

Date: 15 December 2010
Version: 2.0
Status: Final

Document Information

This is the State Of The Art Analysis report for the INSPIRE Transformation Network Service.

Purpose

This report documents leading technologies and existing standards relating to Data Model Transformation and Network Services that are considered relevant to the provision of Technical Guidance to the requirements of the INSPIRE Transformation Network Service (TNS).

It forms the first deliverable within the scope of work for the EC JRC Contract Notice 2009/S 107-153973, as awarded to RSW Geomatics, 1Spatial and Rob Walker Consultancy.

Legal Notice

Table of Contents

1 Introduction

The general requirement of a data provider is to supply data to an external organisation according to the appropriate INSPIRE schema. Where the data schema used to describe the stored data is not the same as the INSPIRE schema, a transformation is required to transform the dataset from the source schema to the INSPIRE schema.

Any dataset to be transformed may not include all spatial objects, attributes and associations in the INSPIRE schema. It may be possible to derive attributes automatically from other sources. The source schema may also contain other spatial objects, attributes and associations that are not in the INSPIRE schema. Spatial objects and attributes are also likely to be differently named in the two schemas, especially as INSPIRE deals with an international, multi-lingual context. Attributes may also be of different data types. Consequently a mapping between the two schemas is required.

The transformation process may be carried out internally by the data provider or externally using a centralised service. In the former case, it will be specific to the internal schema used, while a centralised service will be sufficiently general to deal with a range of input schemas. In each case, spatial object types, their attributes and associations between them need to be transformed. It is assumed that the set of spatial objects to be transformed is determined by a separate process, which is required independently of whether a transformation is required.

This document describes the state of the art in relation to schema transformation services. It documents a number of existing standards and technologies on which the Technical Guidance may be based.

Further analysis will be performed in the following stages of this project to evaluate the most appropriate approach to be recommended in the Technical Guidance. These subsequent evaluation stages will seek to bring in a business context to the analysis, based on feedback expected from a data providers' survey.

The general approach to analysis has been to identify candidate solutions based on existing tools, industry knowledge and the main standards bodies, and to investigate each of these solutions to identify their strengths and weaknesses, based on pre-defined evaluation criteria. Surveys have been sent to major tool vendors in order to gather feedback on the capabilities of existing tools. All information included in the State of the Art Analysis is based entirely on publicly available sources.

The report begins with a brief **Background** to INSPIRE and the need for transformation services.

To enable consideration of the major aspects of the transformation service parameters, the main body of the State of the Art Analysis is divided into the following sections:

- **Schema Description Languages.** Schema description languages compatible with the INSPIRE data specification, source schema and mapping languages.
- **Model Mapping Languages.** Definition, representation and handling of mappings between source and target schema, including inherent limitations.

This is followed by a brief look at currently available transformation tools and finally a review of practical options for deployment within network service environments:

- **Existing Transformation Tools.** Discussion of the existing tools capable of performing the data transformation.
- **Enterprise Architecture.** Evaluation of the options for deployment of a publicly available transformation service.

The report **Conclusion** provides a summary of the key findings of the analysis. It gives an insight into how this will be used in the project's next steps towards the development of the Technical Guidance for the INSPIRE Schema Transformation Network Service (TNS).

2 Background

This section provides a brief introduction to INSPIRE, its purpose, architecture and the concepts surrounding spatial data and the need for schema transformation services. It is intended for those who may be unfamiliar with INSPIRE and sets the context for the State of the Art Analysis.

2.1 INSPIRE

The INSPIRE process is an ambitious initiative intended to build a European Spatial Data Infrastructure (ESDI), to support policy making in issues concerning protection of the environment.

INSPIRE is to be based on the National SDI that are (or will be) created and maintained by the EU Member States. Therefore, one of the main tasks of INSPIRE is to enable *harmonisation* across Europe through *interoperable* spatial data sets and services.

INSPIRE is supported by a legal directive, Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community, which was published in the official Journal on the 25th April 2007 [1]. The INSPIRE directive entered into force on the 15th May 2007.

To ensure that the national SDI of the Member States are compatible and usable in a Community and trans-boundary context, the Directive requires that common Implementing Rules (IR) are adopted in a number of specific areas. Implementing Rules are adopted as Commission Regulations and are binding in their entirety.

As part of the Directive, all public authority organisations that maintain and produce spatial data are required to provide that data to INSPIRE in a way that conforms to the Implementing Rules. The Directive terms these spatial data set providers as Legally Mandated Organisations (LMOs).

2.2 Architecture & Data Concepts

INSPIRE is to deliver integrated spatial information services to its users. The services should allow users to identify and access spatial or geographical information from a wide range of sources (from local to global level) for a variety of uses. All services are described by service metadata (service descriptions), allowing humans and software applications to discover specific service instances in the infrastructure. Similarly, data are described by dataset metadata (data and quality descriptions) to allow discovery and confirmation that the data is fit for the intended purpose.

The INSPIRE architecture is envisaged to be a service oriented architecture, where the components include: metadata, spatial data themes, spatial data services; network services and technologies; agreements on data and service sharing, access and use; coordination and monitoring mechanisms, processes and procedures.

Figure 1 illustrates the key conceptual elements in the INSPIRE architecture (from [4]).

Within this infrastructure, harmonised spatial data across Europe is presented within a geo-portal, supported by a network service layer. The core resource is the actual content (i.e. the spatial data, drawn from many disparate spatial datasets). The other resources (e.g. dataset metadata) are needed to find, access, interpret or use the spatial objects in the spatial datasets.

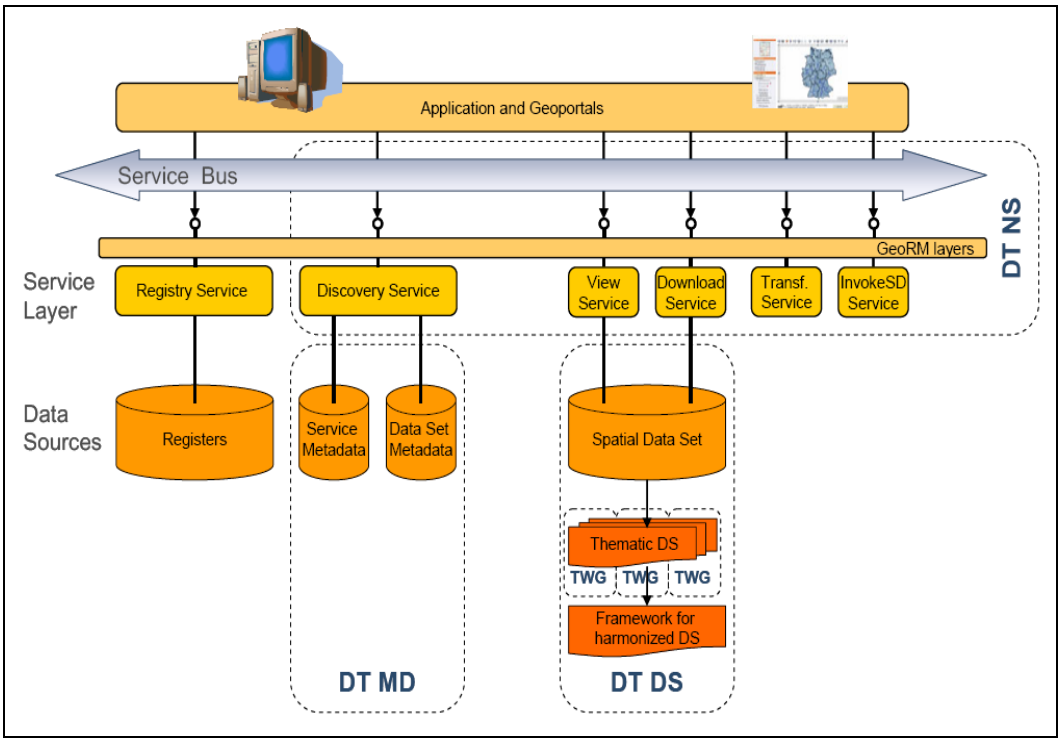

Figure 1 - INSPIRE Technical Architecture Overview (from [4]).

INSPIRE has convened Drafting Teams (DT) to carry out activities to shape the Implementing Rules for Metadata (MD), Data Specifications (DS) and Network Services (NS).

The data is categorised into a set of data themes, each organised into one of three Annexes. As far as the data component is concerned, the first stage of the INSPIRE implementation phase is to confirm the data specifications for the Annex I themes and for organisations to assess the impact on their processes of providing data in a form that is compliant with the specifications. The Annex I themes are: Cadastral Parcels, Administrative Units, Transport Networks, Hydrography, Addresses, Protected Sites, Geographical Names and supporting Coordinate/Grid referencing systems.

Guidelines for INSPIRE data specifications provide detailed technical provisions for data interoperability, but it is unrealistic to expect spatial data providers to migrate their local operating environments to these INSPIRE models and formats. Instead, to achieve compliance, data providers may prefer to establish transformation services, to enable data in local schemas to be transformed to the INSPIRE schema(s). Furthermore, it will be for the data provider to decide whether to conduct this transformation as part of an in-house process, prior to making the data available, or whether to serve the data in its local form, accompanied by a dynamic, open access transformation service.

Technical guidance for the transformation network service is to be primarily aimed at the last of these three scenarios, the open access service, but the guidance may also be applicable to the second scenario, for integration into in-house workflow solutions.

2.3 Implementing Rules for Transformation Services

The Draft Implementing Rules for Transformation Services document [2] outlines the functional requirements of the INSPIRE Transformation Service.

The objective of this project is to provide Technical Guidance for the implementation aspects of an INSPIRE-compliant service to meet the requirements associated with schema transformation. The guidance will minimise any potential ambiguity or the possibility of different interpretations of the purpose and meanings of operations and their parameters within the Implementing Rules.

Achieving this objective should provide confidence that such services can be realised in a practical way. With this knowledge, data providers can enhance their business plans for the introduction to their operations of a sustainable delivery process for INSPIRE-compliant data.

Within the broad context of INSPIRE, it is anticipated that a business request (or user scenario) will initiate a sequence of interoperable services on schema data. These will query and download data from a number of National SDI, to achieve a harmonised view of that data. In support of this, services to transform and integrate the data will be employed, and schema transformation is one such service.

An example service workflow is illustrated in Figure 2. Schema transformation services are anticipated to be one of the first services to be required in these scenarios. These will transform the required data from local schemas (as maintained by the data provider) into the appropriate INSPIRE data schema (along with transformation to INSPIRE-compliant coordinate reference system (CRS)).

To confirm any such service that integrates and/or transforms data, it is necessary to assure the compliance of that data to the specification. In the case of schema transformation, this will be compliance to the INSPIRE schemas and associated domain constraints.

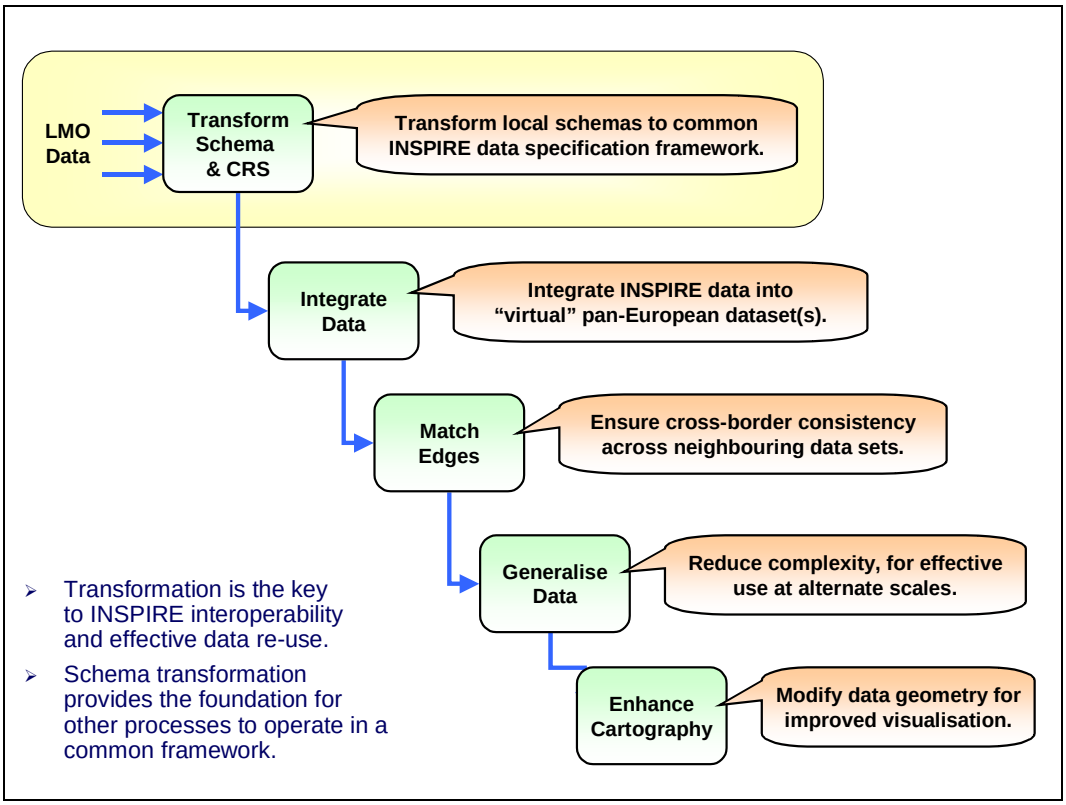

Figure 2 - Illustration of a possible transformation process sequence.

In Figure 3, the '*Transformation Web Service*' represents the generic web service, for which each type of transformation could be implemented i.e. for coordinate reference systems, natural languages, file formats, geometries and schema transformations. This report (and this project) targets schema transformation specifically.

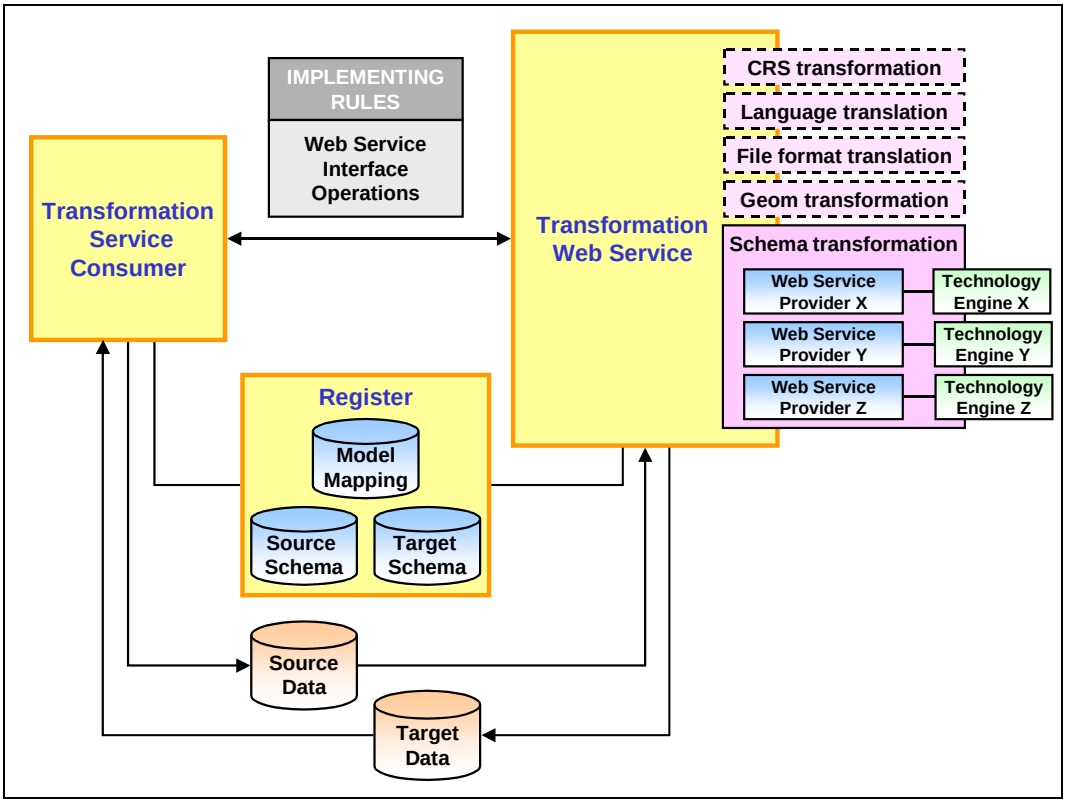

Figure 3 - Conceptual scope of Technical Guidance for INSPIRE Transformation Service.

The Implementing Rules define a set of service operations and accompanying parameters for the Transformation Web Service. These form the '*Web Service Interface*' between the Transformation Web Service and its '*Transformation Service Consumer*' (a client application or other service component).

The scope of the Technical Guidance is to consider:

- Schema Descriptions. Assess standards such as Unified Modelling Language, XML Metadata Interchange, XML Schema Definition and Geography Markup Language.
- Model Mappings between source and target schemas. With few standards currently, consider leading technology neutral encodings and technology dependent mappings.
- Schema Transformation in a service oriented architecture environment. Look to describe the available paradigms that exist both within the GIS community and in mainstream IT as a whole, such as OGC Web Processing Service and Web Services Business Process Execution Language.
- The parameters of the service operations, whether passed in-line (byValue) with the operation request, or as an identifier/pointer (byReference) with the request.

2.4 Analysis of Terminology

The Humboldt project[1] gave an excellent analysis of terms related to schema transformation, in [60]. This identified the distinction between the concepts of data model and schema, conceptual model and conceptual schema. It also described the two-dimensional matrix between the level of abstraction that a modelling language uses, and the scope of the concepts it describes. Since this is both very clear, and also aligns well with INSPIRE terminology, it is proposed to follow it for the purposes of this document. The terms we re-use are given here; and are also included in Appendix C: Terms & Definitions.

Term	**Definition**
Data model	A model of the (geographic) data that is stored and/or exchanged.
Conceptual model	A model that defines concepts of a universe of discourse.
Conceptual schema	A platform-independent (or platform-specific), conceptual model expressed using a formal modelling language (such as UML).
Logical or application schema	A platform-specific description of the structure and constraints applicable to the storage of data for one or more applications (expressed, for example, as an XML Schema (XSD)).
Physical data model	Synonym for logical or application schema.
Physical schema	The concrete, implementation-specific description of how the data is organised in the storage technology of choice (expressed, for example, as SQL DDL).

To these definitions, we add the following for the terms *Data instance, Instance data and Schema*.

Data instance	A single item of data expressed in a concrete storage format (for example, an XML element or database record) which corresponds in some way to an object in the real world such that it is capable of being expressed as an object in an ontology, rather than merely as a predicate or attribute of an object.
Instance data	A collective term for data instances, also known as “row-level data” (especially in a database context).
Schema	A general-purpose term, rather imprecise in nature that may refer to a generic data model, ontology, or database storage structure depending on the context.

[1] The Humboldt project is introduced and discussed further in section 5, and further details can be found in [30].

3 Schema Description Languages

Transformation services require a definition of the source and target schemas in order to be able to apply mappings to data that is organised within these schemas. The transformation service parameters should include a description of the source and target schema, or a reference to such a schema description.

The schema descriptions are intended to describe the permissible constructs in the schema, primarily the syntactic use of the data. Although some schema descriptions also define a limited set of semantic properties of the schema (via constraints on how data may be used within the schema), these are not typically necessary for transformation services. Due to the nature of data modelling and requirements to define mappings in transformation services, the schema description is expected to be a closed world system - all data will be in a finite model that is predefined.

Given the distributed and diverse nature of geographic data, it is advantageous to keep a logical separation between the terms and definitions (ontology) of the application schema and the terms and definitions of the conceptual schema to which the mapping applies.

3.1 List of Identified Schema Description Languages

There are a number of standards that could be used for the schema description language in INSPIRE transformation services. This section provides an analysis of the most significant schema description languages. The candidate languages, based on our own experience and user feedback, are identified in Table 1.

Table 1 Candidate Schema Description Languages

Name / Version of Language	Originator
Unified Modelling Language (UML) 2.2 / XML Metadata Interchange (XMI) 1.1	OMG
XML Schema Definition (XSD) 1.1 / Geography Markup Language (GML) 3.2.1	W3C
Resource Description Framework (RDF) 1.0 / Web Ontology Language (OWL) 2.0	W3C

If required, additional languages could be added following a similar analysis.

The format of the section is as follows: -

- Listing of the criteria used to evaluate the various schema description languages.
- A section describing each of the schema description languages.

In view of the small number of languages evaluated here, it was not necessary to reduce the list any further as an initial exercise prior to detailed evaluation.

3.2 Methodology

The most common schema description languages have been selected for further evaluation. Each of these will be evaluated in terms of the following criteria, which are important for schema transformation services. The evaluation criteria have been devised based on the INSPIRE requirements and experience of the transformation services field. The strengths and weaknesses of each schema description language will be identified in relation to these criteria.

- **Expressiveness**
  The language must be able to represent all concepts required to define the source and target schemas. The main aspects of this are identified in the following section.

- **Mapping Compatibility**
  It should be suitable for use with the mapping language, either directly or indirectly though a well-defined process.
  ***Note**: that verification of the compatibility between a given schema description language and model mapping language will be investigated as part of the subsequent prototyping exercise of the schema TNS. For this reason, it is not covered in the detailed sub-sections below.*
- **Web Compatibility**
  A serialisation must exist that is suitable for use in web services. Must support namespaces that enable division of terms into unambiguous groupings.
- **Tool Support**
  Production tools should exist for editing schema descriptions and for inferring initial schema descriptions from existing data storage implementations (e.g. GML schema or relational database schema).
- **Technology Independence**
  The language should not be tied to any specific vendor or tightly coupled to a specific data-encoding format - it should give a conceptual description of the schema.
- **Intuitiveness**
  The language must have a simple and concise representation such that it is easy for data modellers to work with. This is required in order to allow data modellers to easily express concepts in the schema. Additional complexity should be avoided if possible - the primary focus of a schema description language should be modelling structural traits rather than behavioural traits.

3.2.1 Schema Description Expressiveness

The following list describes core schema aspects that a schema description language must be able to model. These are primarily based on the expressive capabilities of UML, the language used to express the INSPIRE conceptual data specifications, which are the target schema for INSPIRE schema transformations:

- **Object Types**
  A description of the structure of feature instances. This could be as a class in UML terms, or table in relational database terms.
- **Simple Properties**
  Within a type definition, properties or attributes may be defined. Each of these should, in turn, have an associated data type, drawn from a pre-defined set of common, built-in data types, to facilitate transformation.
- **Cardinality**
  Properties may have a defined cardinality, for example to permit voidable singular properties, mandatory singular properties and list properties.
- **Custom Data Types**
  It should be possible to provide custom data types, for example a range of geometric types.
- **Inheritance**
  Specification of generalisations between classes, in order to build an inheritance tree.
- **Aggregation**
  Associations between types, including unidirectional and bi-directional associations, with named association roles. Examples of these are references or foreign key constraints.
- **Composition**
  Composition relationships define complex properties within types. The format of the complex property is defined by a separate type definition which is linked to the first type by a composition relationship.

In addition to the mandatory aspects, it would be desirable if the following were also supported. These aid the understanding of the schema and hence are useful for definition of the mapping or practical implementation of the transformation service:

- **Property constraints**
  Constraints such as properties that must not be null, or ranges of valid values for properties.
- **Default values**
  To be used if no other values are supplied.
- **Annotations**
  Provide user documentation and explanation of schema elements.
- **Application information**
  Support for application specific (machine-readable) annotations, for example to explain the mapping between the schema descriptions and any optimisations or other changes between the logical schema and the physical schema. These will be dependent on the specific physical schema.
- **Grouping**
  It should be possible to organise types into groups or namespaces to simplify work with larger schemas.

3.2.2 Strength Grades

The conformance of the schema descriptions with the above criteria is assigned the following grades. In this document, wherever these grades are used, they have the same relative meaning in the context in which they are applied.

- **Strong**
  The candidate technology performs well in respect of this criterion: there are, apparently, no areas where implementation would encounter significantly costly issues in terms of extra development time or requirements being put at risk.
- **Acceptable**
  The candidate technology performs well in some respects for this criterion but not all: there are, apparently, areas where implementation may incur a risk of expensive issues where extra development time may be needed to overcome them or indeed certain requirements might be unable to be met.
- **Weak**
  The candidate technology performs poorly in respect of this criterion: implementation would, apparently, involve significant risk and expense in development effort to INSPIRE, or insurmountable problems exist.

3.3 Detailed Evaluations

3.3.1 Unified Modelling Language (UML) and XML Metadata Interchange (XMI)

Contact Organisation:

- Object Management Group (OMG)

Version under consideration:

- UML 2.2 [7]
- XMI 2.1.1 [8]

Description

These are standards and open methods used to specify, visualize, modify, construct and document the artefacts of an object-oriented software system under development.

Tools for editing and processing UML models are widely available and commonly used to model schema in both object form and relational database descriptions. Many organisations have a process whereby their schema is modelled in UML and then storage and encoding mechanisms are automatically derived from this, for example DDL for a relational database, or GML application schemas.

As this is an object-oriented design standard, it is easily capable of expressing both simple and complex schemas. The INSPIRE data specifications shall be expressed using UML 2.1. For the purposes of this analysis, there is not a significant difference between versions 2.1 and 2.2. The OMG website states "Version 2.2 is a minor revision to the UML 2.1.2 specification." [62]

UML itself does not specify an interchange format, although XMI is commonly used for this purpose. XMI is a standard for exchanging metadata information via 'Extensible Markup Language' (XML). It can be used for any metadata whose meta-model can be expressed in Meta-Object Facility (MOF). The most common use of XMI is as an interchange format for UML models, although it can also be used for serialization of models in other languages (meta-models).

References

- http://www.omg.org/spec/UML/2.2/ [7]
- http://www.omg.org/spec/XMI/2.1.1/ [8]
- http://www.omg.org/spec/MOF/2.0/ [9]

Evaluation

Criteria	Grade	Rationale
Expressiveness	Strong	UML class diagrams are ideally suited to structural data modelling. There are several features of OWL which are not fully supported in UML, including object enumeration, role fillers, atomic properties, property inclusion, property equivalence, synonym and antonym object specification, classes as instances and the universal class. These are however not generally found in concrete schemas and so seem less relevant for modelling schemas.
Web Compatibility	Strong	XMI is a standard, XML-based serialization language, therefore incorporation of XMI in Web services is trivial. Most UML design tools provide XMI import and export capabilities. However, the compatibility of this format with current tools is frequently identified as being quite poor. Recent testing [43] suggests that this may be improving with XMI 2.0.
Tool Support	Strong	There are a wide variety of tools for editing and manipulating UML models, including a number of open source options. However, tools like ShapeChange and FullMoon, which allow deriving GML schemas from UML models, do not use yet UML version 2. ShapeChange [57] only accepts UML 1.3 (XMI 1.0) and Full Moon [58] accepts UML1.4 (XMI 1.1). Although XMI is a standard, vendor extensions have limited the usefulness of this.
Technology Independence	Strong	UML is independent of toolsets and storage mechanisms.
Intuitiveness	Strong	UML diagrams are simple to understand and widely used. Training resources relating to UML are plentiful.

3.3.2 Geography Markup Language (GML) and XML Schema Definition (XSD)

Contact Organisation:

- GML - Open Geospatial Consortium (OGC)
- XSD - World Wide Web Consortium (W3C)

Version under consideration:

- GML 3.2.1
- XSD 1.0

Description

GML is an XML grammar defined to express geographical features. GML serves as a modelling language for geographic systems as well as an open interchange format for geographic transactions on the Internet.

GML Application Schemas are XML Schema (XSD) documents that import a number of other standard XSD documents in the GML namespace. In addition to the facilities provided by XML Schema, GML adds specific geospatial modelling support, including:

- Geometry types
- Definition and identification of feature types
- CRS, UOM, direction
- Topological, temporal and other related features
- References between features (through W3C xlinks)

GML was originally modelled on RDF, but recent releases are based on XSD since the structure of this is more easily connected to the existing geographic databases. It retains a number of features from its RDF heritage, such as child elements (objects) as properties of the parent object. A GML schema description (GML application schema) is a set of XSDs which import the standard GML schema.

XSD specifies how to formally describe the elements in an XML document. This description can be used to verify that each item of content in a document adheres to the description of the element in which the content is to be placed. This is strongly tied into concepts involved in XML document validation and parsing.

Many tools exist for parsing and editing XML schema, although these generally require significant prior knowledge of the XML schema syntax. Some aspects of data modelling, such as references and inheritance trees are particularly complex when modelled with XSD and are not well supported by tools.

XML Schema 1.0 has limited support for constraining application schema by use of co-occurrence constraints (rule-based constraints on the composition of elements). See the detailed evaluation of XSLT in Section XXX for a brief discussion of options for expressing these kind of constraints on XML documents.

References

- XSD - http://www.w3.org/TR/xmlschema-1/ [10]
- GML - http://www.opengeospatial.org/standards/gml [11]

Evaluation

Criteria	Grade	Rationale
Expressiveness	Acceptable	GML is an expressive language. For example, the INSPIRE data specification includes a GML representation. In addition, it has built-in support for types specifically found in geospatial databases such as various types of geometry. Like XSD on which

		it is based, GML supports only single inheritance between classes.
Web Compatibility	Strong	Standardised, XSD-based format.
Tool Support	Weak	Model editing and viewing is quite a low-level task with currently available tools. Most users of GML adopt a model-driven development approach, choosing to perform their conceptual modelling in a different format (often UML). There are few tools that yet support GML 3.2.1; most tool distributors are still developing for GML 3.1.1. or earlier versions.
Technology Independence	Weak	GML application schema is heavily tied into XML-based data. It is rarely used unless the data requires serialisation to XML/GML.
Intuitiveness	Weak	GML can be quite complex to use, as illustrated by the current scarcity of GML related tools despite a large standardisation effort.

3.3.3 Web Ontology Language (OWL) and Resource Description Framework (RDF) Schema

Contact Organisation:

- World Wide Web Consortium (W3C)

Version Under Consideration

- OWL 2
- RDF 1.0

Description

OWL is a set of knowledge representation languages for authoring ontologies. OWL ontologies are most commonly serialised using RDF syntax. A small number of editor tools are available that use OWL, including Protégé. In addition, standard XML editing tools may be used, but these will require more knowledge of the XML representation syntax.

OWL models data in terms of a set of classes and axioms, which place constraints on these classes, for example the types of relationships permitted between them. There are a number of limitations in definition of the properties of classes, for example assuming directionality in relationships and having little support for permitted ranges and enumerations.

The full OWL2 ontology specification permits the expression of a rich knowledge base regarding data, rather than just a description of data itself. The OWL2 DL profile is more suited to schema description and is designed to contain the features necessary to express conceptual schemas such as UML class diagrams and ER diagrams.

RDF Schema is an extensible knowledge representation language, providing basic elements for the description of ontologies, otherwise called Resource Description Framework (RDF) vocabularies, intended to structure RDF resources. Many RDFS components are included in the more expressive language Web Ontology Language (OWL).

References

- RDF Schema - http://www.w3.org/TR/rdf-schema/ [12]
- OWL - http://www.w3.org/TR/owl-guide/ [13]

Evaluation

Criteria	Grade	Rationale
Expressiveness	Acceptable	OWL is not able to model various constructs that may be modelled in UML, uncluding n-ary associations, qualified associations, bidirectional associations and association classes. Additionally, OWL does not directly support aggregation and composition. Ontology engineering uses a terminological knowledge representation approach to classify extensional knowledge and to infer new knowledge from it. If the extensional knowledge contradicts the ontology, it is identified as not satisfying the ontology. This doesn't map well to the realms of schema description languages, where the data must always comply with one or more concrete schema [63].
Web Compatibility	Strong	Designed for the web, includes XML based serialisations.
Tool Support	Acceptable	There are a few open source tools for authoring ontologies, but these are not as widely used as UML tools.
Technology Independence	Strong	There are several OWL representation languages and no inherent link to particular technologies, although it is frequently used with RDF. Considered from the viewpoints of platform independence and non-affiliation with a particular vendor technology, and given it is a W3C standard, RDF/OWL is strong on this point.
Intuitiveness	Weak	There is not a simple match between common data storage schema and OWL ontologies. Most data modellers will be unfamiliar with OWL and ontologies in general.

3.4 Summary of Schema Description Analysis

It is difficult to evaluate schema description languages in isolation from model mapping languages. UML with XMI representation would, on the face of it, appear a strong candidate, but it proves to be problematic in practice, because there is no consistent format for export of XMI documents from UML models. XSD/GML is generally considered a logical (or application schema) format, however its physical structure is quite verbose and this adds some effort in developing mappings against it. RDF/OWL is essentially an ontology language designed for the Semantic Web. What is interesting about these three candidate schema description languages is that they are all very different: UML is a conceptual schema modelling language, GML is designed for expressing application schemas and OWL, as just said, is designed for expressing ontological concepts. This does not, in principle, preclude at least either OWL or GML being suitable for use with mapping languages. The development of technical guidance and prototyping will assist in clarifying which of these languages is the best option.

4 Model Mapping Languages

A challenging aspect of the TNS problem domain is the lack of a standard meta-language for model mappings. Furthermore, few standards exist for schema transformations and schema mapping encodings. This makes it difficult to implement any TNS in a platform-neutral manner (i.e. without knowledge of the underlying transformation engine). A suitable model mapping language must be selected or designed for use in the transformation service request parameters.

This section analyses a number of ISO, W3C and OGC standards that may be used to assist in this area. These are what could be described as de jure standards. This collection is based on our own experiences and on user feedback. If required, additional languages/tools could be added, following a similar analysis.

The format of this section will be as follows.

- Overview of candidates: a list of candidate mapping languages, with background information, key strengths and weaknesses itemised.
- In order to reduce the number of candidates prior to detailed evaluation, an initial shortlist was prepared. A description is given of the grounds upon which candidates were, or were not, included in the shortlist for detailed evaluation.
- Detailed evaluation criteria: the criteria on which the evaluation of the shortlist of candidates is based.
- Detailed evaluation of selected subset: close scrutiny of the capabilities of the shortlist of candidates.
- Summary of evaluation.

4.1 List of Identified Model Mapping Languages

This sub-section contains an overview of model mapping candidate technologies and tools, falling into three categories:

- **Standards**: languages based on specifications issued by recognised international standards bodies such as W3C and ISO/IEC (per INSPIRE Directive [1] para 28, it is required that appropriate consideration be given to these).
- **Other specifications**: languages based on specifications issued by other organisations.
- **Other languages**: languages originating from other publicly available sources.

Table 2 Candidate Model Mapping Languages

Language	Version or Date	Originator	Category
Extensible Stylesheet Language for Transformations (XSLT)	2.0	W3C	Standard
Web Ontology Language (OWL)	2.0	W3C	Standard
Rule Interchange Format (RIF)	1.0	W3C	Standard
Semantic Web Rule Language (SWRL)	21/05/2004	W3C	Standard
Query/View/Transform (QVT)	1.0	OMG	Standard
Common Logic (CL)	ISO/IEC IS 24707:2007	ISO	Standard
Ontology Mapping Language (OML)	06/10/2005	DERI OMWG	Specification

Rewerse II Rule Markup Language (R2ML)	0.5	WGI1	Specification
Tefkat	2.1.0.lawley266	DSTC Australia	Other

4.1.1 Extensible Stylesheet Language for Transformations (XSLT)

Contact organisation:

- World-Wide Web Consortium (W3C).

Version under consideration:

- Version 2.0.

Description

XSLT was defined as an open standard by the W3C, and v. 1.0 is the current most widely used version of the specification, although v. 2.0, which greatly increases the usefulness and expressiveness of the language with additions such as user-defined functions, has been available since 2007. It functions basically as a template-processor, using pattern-matching to parse XML documents. XSL 2.0 specification is less ambiguous than version 1.0 so that the different XSLT processor implementations could be more interoperable.

References

- XSLT Specification v. 2.0: see [14]
- RELAX-NG: see [15]

Evaluation

Apparent strengths:

- XML/XSLT + XML Schema (XSD) has many implementations and is a well-known language for sharing information.
- Stylesheets coded in XSLT are capable of executing precise and verifiable transformations between source and target XML schema.
- XSLT 1.0 has been proven to be Turing complete, which means that the language can perform any calculation that can be performed by a modern computer (see [25]).
- Work has been done at university masters' degree level to develop spatial extensions to XSLT (GeoXSLT see [26]). Whilst development on GeoXSLT has ceased, it indicates the possibility of extending XSLT to perform spatial transformations that could be used for data model transformation.
- RELAX-NG is an Oasis standard alternative language for modelling XML data structures. Mapping of data models from XSD to RELAX-NG opens the possibility for using RELAX-NG extensions for very flexible processing of schemas to overcome problems mapping e.g. co-occurrence constraints and substitution groups (see [15]). Co-occurrence constraints are discussed further as part of the detailed evaluation of three candidate mapping languages.

Apparent weaknesses

- Tightly bound to XML document structure and not easily mapped to other formats.

4.1.2 Web Ontology Language (OWL)

Contact organisation:

- World-Wide Web Consortium.

Version under consideration

- Version 2.0.

Description

OWL is based on Resource Description Framework (RDF) and RDF-Schema. OWL provides three increasingly expressive sublanguages designed for use by specific communities of implementers and users, of which OWL DL is the intermediate one. OWL DL is so named due to its correspondence with description logics, a field of research that has studied the logics that form the formal foundation of OWL. OWL provides a basis for the conceptual representation of ontologies, and is used to formally classify the different complexity classes of different sorts of logical expression.

References

- www.w3.org/TR/rdf-concepts/ [27]
- www.w3.org/TR/owl2-overview/ [28]

Evaluation

Apparent strengths

- OWL DL supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time).

Apparent weaknesses

- Although OWL supports various types of constraints and reasoning tasks, some kinds of constraints, especially reasoning over relationships, are not supported using the concepts defined in the language.
- OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class).

4.1.3 Rule Interchange Format (RIF)

Contact organisation:

- World-wide Web Consortium

Version under consideration:

- W3C Candidate Recommendation 1 October 2009

Description

RIF is a work-in-progress W3C recommendation for a format that allows logic rules to be exchanged between rule systems, with a normative XML schema representation. RIF is built around three key dialects: RIF Core, RIF Basic Logic Dialect (BLD) and RIF Production Rule Dialect (PRD). RIF Core provides the basic constructs of the language: alphabet, syntax and core semantics. RIF BLD provides the declarative presentation syntax and core semantic structures, XML schema/serialization syntax and conformance clauses. RIF PRD provides definitions of conditions, actions, rulesets, built-in functions and other features designed to support imperative programming statements.

References

- W3C Overview of Rule Interchange Format. [16]
- Introduction to RIF by Dr. Chris Welty of IBM. [17]
- RIF Use Case relevant to INSPIRE schema transformation. [18]
- RIF implementations. [19]
- Oracle's implementation of RIF. [20]

Evaluation

Apparent strengths

- RIF Use Case 4.8 ("Vocabulary Mapping for Data Integration", see references) seems to match very closely to the objectives of INSPIRE TNS.
- There are a number of partial implementations, including Oracle Business Rules (with strong integration with their BPEL engine).
- Other languages such as R2ML have contributed to RIF's development, increasing its usability in heterogeneous rule-processing systems.

Apparent weaknesses

- The current status of most of the RIF documents is "Candidate Recommendation" as of 2009-10-01.
- Most implementations so far appear to be based around singular evaluation of rules against small sets of data, such as XML documents used within BPEL scripts. Schema transformation has a different aim, in terms of repeated evaluation of rules against large datasets. This may, however, be more of a limitation of the current RIF implementations than the RIF standard.
- There are no current, publicly released validation libraries specifically for RIF-PRD, the production rules dialect of most use for mapping transformations.

4.1.4 Semantic Web Rule Language (SWRL)

Contact organisation:

- World-wide Web Consortium

Version under consideration:

- W3C Member Submission 21 May 2004

Description

SWRL combines OWL and RuleML. The latter is an outcome of the Rule Markup Initiative (see references). RuleML's core component is the Datalog sublanguage, which is derived from the intersection of the basic elements of SQL and Prolog. It defines facts and rules as the two key concepts around which the language is modelled. SWRL is built on top of this.

References

- W3C SWRL Specification. [21]
- ruleml.org/ Bijan Parsia; et al. (2005) (PDF). Cautiously Approaching SWRL. [22]

Evaluation

Apparent strengths

- Has full expressive power of OWL Description Logic (OWL DL).

Apparent weaknesses

- It lacks any explicit or existential quantifiers, i.e. it is not possible to express specific multiplicity in a relationship,
- It is not possible to express the existence or non-existence of an instance.
- SWRL contains a fixed set of built-in operators, which address only basic XMLschema data types and therefore has no support for derived geometric types.

4.1.5 Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification

Contact organisation:

- Object Management Group

Version under consideration:

- Version 1.0 (released April 2008)

Description

The QVT specification is one of a series related to developing the 2.0 revision of the OMG Meta Object Facility specification, referred to as MOF 2.0. QVT 1.0 extends OCL 2.0 and MOF 2.0, both of which, in turn, re-use UML 2 Infrastructure Specification elements. Meta Object Facility (MOF) is a meta-language in which models of models can be expressed. Object Constraint Language (OCL) is a language for refining of constraints on objects defined in UML models, and is able to query MOF models. See http://www.omg.org/spec/OCL/2.0/ for the current OCL specification. There are several dialects of QVT: QVT Relations provides the central declarative parts of the language; QVT Operational Mapping Language (QVTo) is an imperative dialect which allows expression of conceptual schema transformations that require additional stateful and/or control-flow processing. This last is the most likely candidate for INSPIRE data model transformations, among the QVT family of dialects.

References

- www.omg.org/spec/QVT/1.0/]. [24]
- http://www.omg.org/spec/OCL/2.0/ [23]
- http://www.omg.org/spec/UML/2.1.2/ [7]

Evaluation

Apparent strengths

- QVT is able to query and transform any MOF-related model. This includes all UML models.
- QVT Operational Mapping Language (QVTo) allows expression of transformations that requires additional stateful and/or control-flow processing.
- QVT provides support for incremental update of transformations, i.e. re-use of the objects associated with a completed transformation.

Apparent weaknesses

- QVT can only process models expressed in a MOF-compliant language.
- QVT is designed for conceptual schema transformations rather than data instance transformations.

4.1.6 ISO/IEC 24707:2007 - Common Logic (CL)

Contact organisation:

- International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC)

Version under consideration:

- ISO/IEC IS 24707:2007 (published 1 October 2007)

Description

Common Logic (CL) is a framework for a family of logic languages, based on first-order logic, intended to facilitate the exchange and transmission of knowledge in computer-based systems. CL has an XML markup language called XCL.

This language has been designed to be as compact and human-readable as possible, and suitable for Web technologies. There is full translatability between CL expressions and XCL expressions.

References

- Common Logic ISO/IEC Standard. [33]
- Common Logic. [34]

Evaluation

Apparent strengths

- CL is related to OWL and RDF, and can be defined as a superset of these and other logic-based languages.
- CL is human-readable in the form of Common Logic Controlled English (CLCE) and machine-readable /Web-exchangeable as Common Logic Interchange Format (CLIF) and these two dialects can be mapped to each other without loss of meaning.
- CL is expressible in XML and has other Web features like IRIs (global resource identifiers to ensure unique namespaces for object typology)

Apparent weaknesses

- The specification is not implemented or actively developed in any production software.

4.1.7 Ontology Mapping Language (OML)

Contact organisation:

- Ontology Management Working Group (OMWG)

Version under consideration:

- DERI OMWG Working Draft 06 October 2005

Description

OML (Ontology Mapping Language) with XML/RDF Serialization has been extended to produce the Geospatial Ontology Mapping Language (gOML) which was used by the Humboldt team as part of the requirements analysis input to the Humboldt Alignment Editor/Conceptual Schema Translation Service project. OML enables users to specify correspondences between two ontologies. The Humboldt team also devised a geographic extension to OML named gOML.

References

- www.omwg.org/TR/d7/d7.2/. [29]
- www.esdi-humboldt.eu/home.html. [30]

Evaluation

Apparent strengths

- OML is able to represent complex correspondences independently from the language in which the ontologies are modelled, and thus to represent any kind of schema mapping.
- Demonstrated by Humboldt team to be capable of extension to cover schema transformation on spatial features.

Apparent weaknesses

- OMWG not an internationally recognised standards body.
- Not a final specification, just a working draft.

4.1.8 REWERSE II Rule Markup Language (R2ML)

Contact organisation:

- REWERSE Working Group I1

Version under consideration:

- Version 0.5 (Release Date: August 23, 2007)

Description

R2ML is an XML rule format that supports interchanging rules between different systems and tools and enriching ontologies by rules. R2ML is comprehensive in the sense that it integrates the Object Constraint Language (OCL), the Semantic Web Rule Language (SWRL) and the Rule Markup Language (RuleML). R2ML allows structure-preserving markup and does not oblige users to translate their rule expressions into a different language paradigm.

References

- REWERSE Working Group I1. [31]
- R2ML Guide. [32]

Evaluation

Apparent strengths

- Has Eclipse-based tool support for visualisation of rules at design time (Strelka).
- Builds on OCL, SWRL and RuleML.

Apparent weaknesses

- Version numbering implies a draft (pre-production) version.
- Project has not been updated since 2007.
- Issue tracking on project website is not clearly managed.

4.1.9 TefKat

Contact organisation:

- Distributed Systems Technology Centre (ceased operations on 30 June 2006)

Version under consideration:

- tefkat.feature - 2.1.0.lawley266

Description

Tefkat is a declarative model transformation language based on QVT and designed for Model-Driven Development (MDD) and data transformation. Unlike XSLT, Tefkat has a simple and familiar SQL-like syntax, is specifically designed for writing scalable and re-usable transformation specifications using high-level domain concepts rather than operating directly on XML syntax. Thus, Tefkat offers both a language and an implementation of the language.

References

- Tefkat on Wikipedia. [35]
- Tefkat's Sourceforge site. [36]

Evaluation

Apparent strengths

- Mappings can be defined using UML models, thus capable of visualisation in ways programmers can understand readily.

- It is implemented as an Eclipse plugin that leverages the Eclipse Modelling Framework (EMF) to handle models based on MOF, UML2, and XML Schema.

Apparent weaknesses

- The organisation sponsoring Tefkat's development has ceased operations and it is unclear whether any further development is planned.

4.2 Methodology

This sub-section describes the rationale behind the short-listing of three-candidate model mapping technologies for more in-depth evaluation.

4.2.1 Included Candidates

RIF, QVT and XSLT were selected for more detailed evaluation. The reasoning is as follows:

- RIF: W3C standard, formal theoretical basis, alignment with OWL and RDF which is becoming central to the Semantic Web and open data initiatives, and the existence of a number of experimental implementations, among which products from Oracle and IBM are included.
- QVT: Based on the MOF/UML family of languages arising from years of practical MDA experience in real-world implementations in general information modelling domains. Has tool support from Eclipse Modelling Framework e.g. M2M, IBM tools.
- XSLT: Although limited to XML processing, has great familiarity and wide adoption for a decade with two versions of specification released; strengths and weaknesses are well known; relatively straightforward extensibility via Java or other 3/4GL languages.

4.2.2 Excluded Candidates

OWL, SWRL, Common Logic, OML, R2ML and Tefkat were de-selected for further evaluation. The reasoning is as follows:

- OWL: Very rich and expressive ontology mapping (classification) language. However it has comparatively limited facilities for expressing constraints and production rules.
- SWRL: Lacks quantification operators and prohibits user-defined functions that could facilitate extensibility. Disjunction and negation in statements are excluded from the language, thus severely limiting its rule-definition capabilities.
- Common Logic: Although supported by an international standard, scarcity of implementations prevents further consideration of CL as the basis for an INSPIRE model mapping language.
- OML: It is a working draft with no clear plans to transform it into a standard propounded by a recognised international standards body.
- R2ML: Interesting option conceptually, in that seeks to span the divide between OWL/RDF (AI) and MOF/UML (MDA) language families. However, mainly academic examples of implementations are available.
- Tefkat: Was written by the Distributed Systems Technology Centre, an Australian Cooperative Research Centre which ceased its activities in 2006 (see [1]. Therefore it must be considered as an academic rather than production standard language/implementation. In addition, an investigation [38] discovered a long list of shortcomings of Tefkat that would need to be addressed in order for it to provide the basis for an INSPIRE transformation model mapping solution.

4.2.3 Evaluation Criteria

This analysis documents options for model mappings and compares them using common evaluation criteria as set out below. The evaluation criteria are split into expressiveness and implementation features.

Expressiveness Criteria

The language must be capable of expressing non-trivial mappings, such that it will be possible to define appropriate mappings for the transformation of the vast majority of data providers' data to the INSPIRE data specification. This criterion logically subsumes the categories of expressiveness defined for schema description languages (see section 3.1). The sub-criteria applied, based on the problems likely to be encountered in a transformation service, are the ability to express mappings that include:

- **Instances and Properties**
  Includes basic types, literals and lists, built-in functions. Whether the language is able to express fundamental syntactic tokens and functions.
- **Classes**
  Includes classes of instances (i.e. sets and membership). Whether the language is able to model categorisation of objects into classes.
- **User-defined functions**
  Includes ability to define predicates or functions (such as spatial functions) which are necessary for INSPIRE transformations.
- **Relations**
  Includes relationships between classes (association, composition, aggregation). Whether the language is able to model how classes interact with one another.
- **Inheritance**
  Includes inheritance between classes (single and multiple). Whether the language allows for the expression of trees of sub-classes and super-classes.
- **Quantification**
  Includes universal and existential quantification ("for all", "exists").
- **Countability**
  Includes both cardinality (how many instances) and ordinality (order of the instances e.g. in a sequence).
- **Negation**
  Whether the language supports the concept of negation of truth-valued terms.
- **Co-Occurrence Contraints**
  These types of constraint specify which objects can be included together in relationships between different sets of objects. The INSPIRE data specifications make extensive use of them to control the choice of object types available when composing target instances.

Implementation Criteria

- **Technology Independence**
  Model mappings can be organized in two ways: (i) technology neutral / via generic mapping encodings; and (ii) technology dependent mappings used by different technologies. Our investigations consider possibilities whereby the model mappings are encoded using a generic language. These require each platform-specific transformation adapter that is integrated into the solution, to obtain the mapping and convert it into its own format as it chooses.

- **Practical Feasibility**
  Proof of implementation of the language in an efficient and practical manner must be demonstrated. This should ideally be in existing production software. If no direct implementation exists, but an implementation of an equivalent or similar system exists, then this is also considered. Additionally, availability of tools to aid with the design of mappings would be considered beneficial.
- **Manageability**
  For reasons of manageability and ease of comprehension, it is important that the rules language has a concise grammar.
- **Intuitiveness**
  It is important that the language be naturalistic and easy to learn, for ease of understanding and widespread use.
- **Web Compatibility**
  The language should be compatible with feature data that is scattered across multiple physical and organisational barriers. It must have a representation that is suitable for use in web services.
- **Logical Portability**
  Given the distributed and diverse nature of geographic data, it is advantageous to keep a logical separation between the terms and definitions (ontology) of the application schema and the terms and definitions of the conceptual schema to which the mapping applies.
- **Custom Extensions**
  Some concepts involved in INSPIRE schema transformation require a language to support custom spatial extensions in addition to its core expressivity. Examples are coverage functions (raster-type mappings from a domain e.g. coordinates to a range of attributes e.g. temperatures, water quality, or soil composition categories) and spatial analytical functions (e.g. boundary or nearest neighbour algorithms).

4.2.4 Explanation of How Criteria are Applied

The languages and tool candidates are assessed against each expressiveness criterion based on whether they meet, or fail to meet, the criterion: this corresponds to a Yes or No evaluation.

Assessment against the implementation criteria is done based on a simple enumeration: strong, acceptable or weak. The meanings of these grades are the same as those described in Section 3.1.2.

4.3 Detailed Evaluations

This sub-section contains the detailed evaluations of the short-listed candidates.

4.3.1 Detailed Evaluation of RIF

The detailed evaluation of RIF against the identified assessment criteria is described here. To read into these sources, please see references [39],[40] and [41].

Expressiveness

Criteria	Grade	Rationale
Instances and properties	Yes	Basic types: see www.w3.org/TR/2009/CR-rif-dtb-20091001/#sec-constants.Predicates: see www.w3.org/TR/2009/CR-rif-prd-20091001/#sec-builtins.Literals: see www.w3.org/TR/2009/CR-rif-bld-20091001/#Alphabet_of_RIF-BLD.Lists: see www.w3.org/TR/2009/CR-rif-prd-20091001/#sec-conditions-abstract-syntax.Built-in functions: see www.w3.org/TR/2009/CR-rif-dtb-20091001/#List_of_RIF_Built-in_Predicates_and_Functions.
Classes of instances	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#sec-terms.
Relationships between classes	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#Abstract_syntax.
User-defined functions	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#Built-in_functions.2C_predicates_and_actions.
Inheritance between classes (single)	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#sec-terms.
Inheritance between classes (multiple)	Yes	This is able to be expressed using multiple, overlapping inheritance statements. See www.w3.org/TR/2009/CR-rif-prd-20091001/#Operational_semantics_of_rules_and_rule_sets
Universal and existential quantification	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#Rules.
Negation	Yes	See www.w3.org/TR/2009/CR-rif-prd-20091001/#sec-formulas.
Countability	Yes	For cardinality, see http://www.w3.org/TR/2009/CR-rif-dtb-20091001/#Functions_and_Predicates_on_RIF_Lists. For loop iteration, see http://www.w3.org/2005/rules/test/repository/tc/Modify_loop/Modify_loop-premise.rifps.
Co-occurrence Constraints	Yes	A co-occurrence constraint is essentially a rule, where the rule conditions include multiple target instances and the rule action specifies production of a given target object pattern. Since RIF supports reasoning on target instances, there is nothing to prevent it expressing these structures. See e.g. the “Discount Rule” in http://www.w3.org/TR/rif-prd/#sec-running-example.

Implementation

Criteria	Grade	Rationale
Technology independence	Strong	Has been defined as an independent standard with the objective to enable portability (see www.w3.org/TR/rif-ucr/#Negotiating_eBusiness_Contracts_Across_Rule_Platforms). Is sponsored by W3C as a public standard.
Practical feasibility	Acceptable	Has a number of production-level implementations including Oracle Business Rules , IBM WebSphere ILOG/JRules (see www.w3.org/2005/rules/wg/wiki/List_of_Rule_Systems). RIF-PRD (RIF Production Rules Dialect) supports chaining of rules in ways that practical, business-oriented systems can implement (see www.w3.org/TR/2009/CR-rif-prd-20091001/#Production_rules_and_rule_sets). RIF use case 4.8 "Vocabulary Mapping for Data Integration" relates closely to the INSPIRE schema transformation problem (see www.w3.org/TR/rif-ucr/#Vocabulary_Mapping_for_Data_Integration).
Intuitiveness	Acceptable	The RIF specification uses some formal mathematical notation that is difficult for non-mathematicians to understand. A presentation language, with mapping from the normative XML syntax, is also available and this is easier to read (for an example, see www.w3.org/TR/rif-bld/#Overview).
Manageability	Strong	The RIF-CORE, RIF-BLD and RIF-PRD dialects are defined in compact and manageable specifications. For example, the RIF-CORE specification document runs to a little over 1,000 lines of text, RIF-BLD to 2,700 lines and RIF-PROD to 3,000 lines.
Web compatibility	Strong	Has been defined with Semantic Web in mind (see www.w3.org/TR/2009/WD-rif-overview-20091001/#Introduction), so that a) namespaces are defined by IRIs, for manageability on the Web, b) is compatible with other SW languages such as RDF and OWL. Normative expression is in XML and an XML schema exists, which makes it possible to validate a RIF document simply using XML schema validation (see the document RIF Combination with XML data at www.w3.org/TR/2009/WD-rif-xml-data-20091001/). These factors enable RIF documents to be transmitted as parameters to web service operations, within the body of SOAP messages.
Logical portability	Strong	RIF is essentially a format for porting rules between one system and another, where the disparate systems use different rule encodings (see www.w3.org/TR/2009/WD-rif-overview-20091001/#Introduction). RIF is based on OWL and RDF, and there are well-defined compatibility relationships between these languages. Rules expressed in other formats, e.g. SWRL, can also be exchanged using RIF.
Custom Extensions	Strong	It is possible to add new custom dialects to RIF using the RIF Framework for Logic Dialects (RIF-FLD), see http://www.w3.org/TR/2009/CR-rif-fld-20091001/. Alternatively, XML schema can be imported into RIF, see http://www.w3.org/TR/2009/WD-rif-xml-data-20091001/#Importing_XML_documents_and_schemas_in_RIF. In this case, the schema contains the interface to a set of spatial operations or functions and the parameter datatypes required as input or output to them.

Discussion

RIF is capable of expressing a wide variety of rules. It has support for all the standard datatypes such as String, number, boolean, timestamp, and built-in functions to operate on them (see RIF Datatypes and Built-Ins 1.0 standard), plus collection types and operations, classes and inheritance. RIF-CORE and RIF-BLD enable definition of logical rules. RIF-PRD enables definition of custom actions as required. RIF-PRD has support for negation statements (IF NOT this, THEN that).

The overall impression of RIF is of a rich and expressive language that is capable of describing the required model mappings for INSPIRE data model transformation.

However, it is quite difficult to access owing to the non-existence of introductory material available publicly for training programmers who wish to implement systems using RIF, and a background in formal logic is helpful in this regard. Nevertheless, several large software vendors have developed software supporting RIF, which indicates its growing adoption. The development of tool support for the construction of RIF documents will assist greatly in the process of establishment of RIF as a *de facto* player in the transformational model mapping domain.

4.3.2 Detailed Evaluation of QVT

The detailed evaluation of QVT against the identified assessment criteria is described here. To read into these sources, please see references [7], [9], [23] and [24].

Expressiveness

Criteria	Grade	Rationale
Instances and properties	Yes	Basic types: UML 2 infra s. 10.1 "Types Diagram", s. 12.1 "Primitive Types"; OCL 2.0 s. 7.4. Predicates: QVT 1.0 s. 7.1.1 Literals: UML Infra s. 9.1.1 "Literals"; OCL 2.0 s. 7.4. Lists: OCL 2.0 ss. 7.6, 8.2. Built-in functions: OCl 2.0 s. 7.11 on OCL Standard Library.
Classes of instances	Yes	UML 2 Infra. s. 10; OCL 2.0 s. 7.5.
Relationships between classes	Yes	UML 2.0 Infra s. 9.17; OCL 2.0 s. 7.5.
User-defined functions	Yes	QVT 1.0 s. 9.16.
Inheritance between classes (single)	Yes	UML 2 Infra. s. 10 "Classes".
Inheritance between classes (multiple)	Yes	UML 2 Infra. s. 10 "Classes".
Universal and existential quantification	Yes	See OCL 2.0 v. 7.6.3 ("forall") s. 7.6.4 ("exists").
Negation	Yes	See OCL 2.0 s. 7.4 on boolean variables.
Countability	Yes	See UML 2.0 Infra s. 9.1.2 "Multiplicities".
Co-occurrence Constraints	Yes	OCL is fundamentally a language for defining constraints on associations and models. See e.g. OCL 2.0 s. 7.5 on associations.

Implementation

Criteria	Grade	Rationale
Technology independence	Strong	Has been defined as a public standard, not restricted to use by one vendor or in a single tool.
Practical feasibility	Strong	QVT has tool support from the Eclipse Modelling Framework (EMF). It is part declarative and part imperative which makes it easier to develop with than purely declarative languages.
Intuitiveness	Strong	The QVT Specification is easy to read and accessible for programmers seeking to apply the concepts to develop transformations. Familiar analogy such as that of the Java Virtual Machine is used to aid understanding, together with UML models to express key concepts. Custom notation is explained

		using ordinary language.
Manageability	Acceptable	Defined as an extension to OCL and MOF, hence the QVT specification itself is not very big. However, reliance on four specifications (UML Infrastructure, MOF, OCl and QVT) implies extra challenges.
Web compatibility	Strong	All EMOF compliant models can be serialized into XML format using XMI.
Logical portability	Acceptable	QVT is based on MOF 2.0 and can transform to and from any model expressed in a language compliant with that standard. However, QVT can't transform models not expressed in MOF-based languages.
Custom Extensions	Strong	QVT includes a mechanism called “Black Box” which allows incorporation of mappings expressed in other languages e.g. XSLT, see http://www.omg.org/spec/QVT/1.0/PDF/ s. 6.2.2.

Discussion

QVT is capable of expressing full-scale mappings between different schema e.g. UML to RDBMS. Has scope for inheritance hierarchies, object relationships, complex algorithms, validation-only transformations, bi-directional transformations, incremental updates on re-application of a transformation when the source schema has changed, re-usability of transform libraries. Mappings can also be checked formally for correctness.Also, the language relaxes the idea of source and target schema, so that a schema can be either source or target, depending on how the transformation is defined. Bi-directional transformations (at least using QVT Relations) support this aspect of the language.

QVT has added value in permitting incremental transformations at no extra development cost. This is achieved by trace instances which can be created implicitly i.e. without programmer effort and re-used multiple times when the source schema changes and the changes need propagating. See QVT spec 2 4.1.1 glossary->trace instances.

4.3.3 Detailed Evaluation of XSLT

The detailed evaluation of XSLT against the identified assessment criteria is described here. To read into these sources, please see references [10], [14] and [42].

Expressiveness

Criteria	**Grade**	**Rationale**
Instances and properties	Yes	Literals: see www.w3.org/TR/xpath20/#id-literals Basic types: see www.w3.org/TR/xpath20/#id-types Predicates: see www.w3.org/TR/xpath/#section-ExpressionsLists: see [2]www.w3.org/TR/xslt20/#creating-new-nodes Built-in functions: see http://www.w3.org/TR/xpath/#corelib
Classes of instances	Yes	By pattern matching and templates: see www.w3.org/TR/xslt20/#rules, nodes of different types can be output, giving similar functionality to classes.
Relationships between classes	Yes	Relations between elements (and transformations applied to them) can be defined in templates or expressed as XLinks.
User-defined functions	Yes	This is new as of XSLT 2.0, see www.w3.org/TR/xslt20/#dt-stylesheet-function.

Inheritance between classes (single)	Yes	Stylesheets can import other stylesheets, thus providing functionality similar to subclasses/superclasss; also, named attribute sets (www.w3.org/TR/xslt20/#attribute-sets) enable re-usable construction of new nodes similar to Java "extends".
Inheritance between classes (multiple)	Yes	XML Schema supports an element being able to participate in multiple substitution groups, and thus implements multiple inheritance. However, this is a new feature in XSD 1.1 which is still at working draft stage and there are unlikely to be any tools that support this feature at present.
Universal and existential quantification	Yes	See www.w3.org/TR/xpath20/#id-quantified-expressions.
Negation	Yes	By XQuery: see www.w3.org/TR/xquery-semantics/#sec_comparisons and www.w3.org/TR/xquery-operators/#boolean-functions.
Countability	Yes	Cardinality: by use of XPath sequence and aggregate functions, see www.w3.org/TR/xquery-operators/#sequence-functions and www.w3.org/TR/xquery-operators/#func-count. Ordinality: see www.w3.org/TR/xslt20/#sorting
Co-occurrence Constraints	Yes	XSLT is reliant upon the expressiveness of the schema description language, and requires it to be at least an XML based language. If the schema is expressed using W3C XML Schema, there are differences in respect of co-occurrence constraints between v. 1.0 and v. 1.1 of that standard. XML Schema v. 1.0: offers few possibilities for expressing this: e.g. one can define a container element with an xsi:type attribute that references a complex type in which the content specifies which element(s) are permitted. XML Schema v. 1.1: introduces constraining rules using the `xs:assert` syntax on complex types and `xs:restriction` on simple types. In addition, `xs:alternative` allows multiple possible contained elements depending on the outcome of a test. There are several alternatives to XML Schema which could be used, two of which are Schematron and RELAX-NG. Schematron: a rule-based XML validation language (Document Schema Description Language or DSDL) which is ideally suited for defining expected element content based on identifying patterns in a document. Conforms to ISO/IEC standard 19757. RELAX-NG: an XML schema language which can be used together with XML Schema and provides support for defining patterns to control XML document structure (and is another ISO/IEC 19757 implementation). For example, features such as choices and named patterns make definition of co-occurrence constraints routine. Hence, there are options for expressing co-occurrence constraints with XSLT but they depend on the choice of schema description language. XSD 1.0, the version referred to in the GML 3.2.1 specification, offers only limited support for expressing these constraints.

Implementation

Criteria	Grade	Rationale
Technology independence	Strong	XSLT was defined as an open standard by the W3C, see www.w3.org/TR/xslt20/. XSLT v. 1.0 is the current most widely used version of the specification, although v. 2.0 has been available since 2007.
Practical feasibility	Weak	XSLT is used very widely in production environments across the world, including implementations involving high data volume processing and significant system complexity. It is a seasoned technology whose constraints, and the designs able to circumvent them, are well known. However, the challenge of processing GML at production volumes would raise serious problems. XSLT works on the basis of building a Document Object Model (DOM) tree containing the XML elements in the document it is tasked with transforming. If multiple gigabytes worth of GML character data were parsed into such a DOM tree in order to attempt a transformation, this would quickly become unfeasible for lack of available memory resources.
Intuitiveness	Weak	XSLT is a declarative language (see en.wikipedia.org/wiki/XSLT). This has the features that definitions are typically stateless and direct control-flow tokens are unavailable (this is not strictly true for XSLT which has e.g. if-else statements) so development of transformations can be harder. Also, it is dependent on XQuery/XPath for selecting nodes from XML documents (see www.w3.org/TR/xslt-xquery-serialization/). It is expressed in XML so that the processing language has the same syntax as the format it seeks to process, and this can be confusing.
Manageability	Acceptable	The XSLT specifications vary in compactness: v. 1.0 more so, it's specification runs to about 3,500 lines of documentation. v. 2.0's is four times longer, about 12,000 lines of documentation. This compares with about 6,000 lines for the UML 2.1.2 specification. The reason for the greater size of XSLT 2.0 is the accommodation of more flexible processing and backward compatibility with v. 1.0 while adding many new features (see www.w3.org/TR/xslt20/#whats-new-in-xslt2).
Web compatibility	Strong	XSLT is encoded in XML so models are passable as parameters on the Web.
Logical portability	Weak	It is only capable of performing transformation on source data encoded in XML (see en.wikipedia.org/wiki/XML_transformation_language). This means that any dataset expressed using a different encoding needs firstly to be translated into XML (and potentially back again to provide the target format).
Custom Extensions	Acceptable	Since it is both a mapping language and a transformation engine, XSLT is only as expressive as the processors that implement it. Few spatial extensions exist for XSLT (see comments above about GeoXSLT). However, there is some scope for extending XSLT processors: e.g. Xalan-Java [72], Saxon [73] and xsltproc [74] all support extensibility using standard programming languages such as C and Java.

Discussion

XSLT was defined as an open standard by the W3C, and v. 1.0 is the current most widely used version of the specification, although v. 2.0 has been available since 2007. Stylesheets coded in XSLT are capable of executing precise and verifiable transformations between source and target XML schema. However, resource consumption and performance are concerns that limit the practicality of processing very large GML documents in DOM trees.

4.4 Summary of Model Mapping Analysis

This section summarises the strengths and weaknesses of each of the closely evaluated mapping tools and technologies, against the identified criteria.

Expressiveness

Criteria	**RIF**	**QVT**	**XSLT**
Instances and properties	Yes	Yes	Yes
Classes of instances	Yes	Yes	Yes
Relationships between classes	Yes	Yes	Yes
User-defined functions	Yes	Yes	Yes
Inheritance between classes (single)	Yes	Yes	Yes
Inheritance between classes (multiple)	Yes	Yes	Yes
Universal and existential quantification	Yes	Yes	Yes
Negation	Yes	Yes	Yes
Countability	Yes	Yes	Yes
Co-Occurrence Constraints	Yes	Yes	Yes

Implementation

Criteria	**RIF**	**QVT**	**XSLT**
Technology independence	Strong	Strong	Strong
Practical feasibility	Acceptable	Strong	Weak
Intuitiveness	Acceptable	Strong	Weak
Manageability	Strong	Acceptable	Acceptable
Web compatibility	Strong	Strong	Strong
Logical portability	Strong	Acceptable	Weak
Custom Extensions	Strong	Strong	Acceptable

The evaluation demonstrates that all three languages have considerable expressiveness and would be capable, in theory, of supporting the variety of expressiveness needs of an INSPIRE schema transformation. Combined with the implementation criteria results, however, the evidence is that XSLT is a much weaker contender than either RIF or QVT for selection as a suitable model mapping language.

5 Transformation Tools

Sections 3 and 4 have identified and evaluated methods of defining schemas and mappings between schemas. This included standard languages and other specifications as well as other languages. This section investigates the practicalities of implementing transformation services using existing transformation tools. Investigation of the capabilities of existing tools helps to validate the practicality of requirements defined in the implementing rules and also helps identify possible constraints that may need to be considered in the Technical Guidance, to ensure that the transformation services can be implemented by multiple vendors.

The format of this section is as follows: -

- The list of identified transformation tools
- Outline of the methodology used to select which tools were evaluated, and the criteria which were applied
- Detailed evaluation of each of the evaluated tools.

5.1 List of Identified Transformation Tools

Using industry knowledge, a comprehensive (but not exhaustive) list of transformation tools was constructed for the purposes of this project. The following vendors/distributors were contacted as part of the State of the Art Analysis.

Vendor or distributor of tool	**Name of tool**	**Tool Version**
SAFE Software	FME Server	2010
Snowflake Software	GO Publisher	1.4
interactive instruments GmbH	XtraServer	3.2
1Spatial	Radius Studio	2.1.0.15
Geodan	*Not specified*	*Not specified*
GIS4EU	GIS4EU server	*Not specified*
con terra GmbH	FME Server *	2010
lat / long GmbH	Deegree WPS	3.0
Talend	Talend Integration Suite	*Not specified*
Humboldt	Humboldt Alignment Editor/Conceptual Schema Translation Service	HALE 2.0.0-M1, CST 1.0.0-RC1
Oracle	Oracle Spatial	11g
ERDAS	RedSpider Enterprise	3
Altova	MapForce	2010
GeoTools	GeoTools library	*Not specified*
52° North	GeoProcessing, Semantics	*Not specified*
GeoServer	GeoServer	2.0.0
AuScope	AuScope Grid **	*Not specified*

* con terra GmbH advised in response to our survey that they use FME Server 2010.

** AuScope Grid uses GeoServer.

5.2 Methodology

A two-pronged approach was undertaken:

a) An online survey was conducted to obtain information from the vendors/distributors about the suitability of their product for schema transformation; a list of requirements for schema transformation, i.e. schema transformation levels, provided input to this survey; tool providers were asked to comment on the suitability of their tool to achieve each of these levels of transformation.

b) Once responses were received, further information was obtained from the public websites of the organisations who had responded, in order to build up a set of outline evaluations of the candidate tools.

5.2.1 Survey

The online survey contained questions relating to:

- How schemas and mappings are defined using the tool, particularly with reference to standard languages if applicable.
- The level of transformation functionality provided by the tools (c.f. the expressiveness of the schema mapping language). Responses were requested using the context of a set of transformation functionality levels as discussed below.
- How widely used the tool is, in terms of customers and data sizes; and any scalability limitations.
- Options for deployment of the tool, for example in a web compatible manner.

The full set of questions is reproduced in Appendix A.

The Data Model Transformation Tools survey was sent to 17 potential transformation tool vendors or non-profit based projects, using the online survey tool *www.surveymonkey.com*. An open invitation for responses to this survey was also published on the INSPIRE Web Forum. Responses were received from eight organisations. The authors would like to thank the individuals and organisations that kindly responded:

- interactive instruments (XtraServer).
- lat/long GmbH (Deegree).
- SAFE Software (FME Server).
- Snowflake Software (GO Publisher).
- Talend (Talend Integration Suite).
- 1Spatial (Radius Studio).
- Humboldt (HALE and CST).
- AuScope Limited (AuScope Grid & GeoServer).

5.2.2 Schema Transformation Levels

To aid discussions on transformation functionality, a number of capability levels were defined. These describe different types of functionality that may be required in order to transform schema of varying complexity. When the source schema is closely aligned to the target schema, a lower level of transformation functionality is required. Each level incorporates all functionality from earlier levels, i.e. if a transformation service supports functionality in level n, it should also support all functionality in level n–1.

- **Level 1** - Renaming classes and attributes.
- **Level 2** - Simple attribute derivation.

- **Level 3** - Aggregating input records.
- **Level 4** - Complex derivation and dynamic type selection.
- **Level 5** - Deriving values based on multiple features.
- **Level 6** - Conflation and model generalisation

These levels are described in more detail in Appendix B.

5.3 Detailed Evaluations

The follow sub-sections relate to the evaluation of existing transformation tools, some of which are commercial and some of which are research projects where the software is made available publicly. This evaluation relates only to the task of schema transformation, the implication of which is that issues related to other forms of transformation (for example, data format translation), whilst useful in evaluating tools, are not the main focus. The sub-sections are organised with commercial tools listed first and research projects following them.

Tool Distributor / Name / Version	Type
Interactive Instruments / XtraServer / 3.2	Commercial
Safe Software / FME Server / 2010	Commercial
Snowflake Software / GO Publisher / 1.4	Commercial
Talend / Integration Suite Enterprise Edition / 3.2.3	Commercial
1Spatial / Radius Studio / 2.1.0.15	Commercial
OSGeo / Deegree Web Processing Service / 2.2	Research
Humboldt Project / Humboldt Alignment Editor & Conceptual Schema Translation Service / 1.0.0-RC1, 2.0.0-M1 respectively	Research
AuScope Limited / AuScope Grid / *version not specified*	Research

5.3.1 XtraServer

Contact organisation

- Interactive Instruments

Version under consideration

- Version 3.2

Description

Interactive Instruments' Xtraserver provides Web Feature Server (WFS) and Web Map Server (WMS) implementations. The WFS supports versions 1.0 and 1.1 of the OGC WFS Specification. XtraServer WFS is capable of performing schema transformations to meet all of the six levels of requirement. It uses a proprietary, XML-encoded mapping language which enables complex mapping rules to be refined to cover the required scenarios. In addition, XtraServer supports customisation using SQL to further refine the mapping from the source schema to the target schema.

References

- www.interactive-instruments.de/index.php?id=xtraserver&L=1. [46]
- www.opengeospatial.org/standards/wfs. [47]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - SUPPORTED

Apparent strengths

- Mappings can be defined using UML.
- XtraServer schema mappings, being expressed using XML, are suitable for transmission as parameters to Web services operations.
- Extensive support for spatial data formats, including GML 3.2

Apparent weaknesses

- Mapping language is proprietary hence it is not possible to exchange mappings with other tools or use mappings developed elsewhere.
- SQL transformations can only be performed in a relational database.

5.3.2 FME Server

Contact organisation

- Safe Software

Version under consideration

- FME 2010 Professional Edition

Description

FME Server is a spatial ETL platform offering flexible spatial data distribution and scalable data loading and conversion. It offers, among other features, server-based spatial translation and transformation services. Its schema mapping language is proprietary. The functionality is made available to users through a graphical user interface. Pluggable transformers offer discrete processing steps and are chainable to form transformation flowlines.

References

- www.safe.com/products/server/overview.php. [48]
- www.safe.com/c/inspire/inspire.php. [49]
- http://downloads.safe.com/fme/brochures/transformers.pdf [50]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED

- Level 6: Model conflation and generalisation - SUPPORTED

Apparent strengths

- FME Server is very flexible and offers extensive data format and transformation support.
- User interface is well-designed and intuitive making it easier for non-programmers to use.

Apparent weaknesses

- Mapping language is proprietary hence it is not possible to exchange mappings with other tools or use mappings developed elsewhere.

5.3.3 GO Publisher

Contact organisation

- Snowflake Software

Version under consideration

- Version 1.4

Description

Snowflake Software's GO Publisher is a flexible, production standard platform for publishing spatial data. GO Publisher has a core transformation engine that is configurable by the GUI.

References

- http://www.snowflakesoftware.co.uk/markets/inspire/solution.htm. [51]
- http://www.snowflakesoftware.co.uk/tv/index.htm [52]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - PARTIAL

Apparent strengths

- Graphical user interface enabling transformations to be configured by non-programmers.
- Extensive support for spatial data formats including GML 3.
- In addition to the core transform capability, the engine as three extension points where users can use database functions, XSLT fragments or pure Java coding to enhance the transformation support.

Apparent weaknesses

- Mapping language is proprietary hence it is not possible to exchange mappings with other tools or use mappings developed elsewhere.

5.3.4 Talend Integration Suite Enterprise Edition

Contact organisation

- Talend

Version under consideration

- Talend Open Studio v3.2.3

Description

Talend Open Studio, on which Talend Integration Suite is based, is a general purpose ETL (Extract-Transform-Load) platform. Talend also distribute a Spatial Data Integrator based on Open Studio. It is capable of processing several GIS formats including PostGIS, ESRI Shapefile and MapInfo MIF/MID.

References

- www.talend.com/products-data-integration/talend-integration-suite.php. [53]
- www.vividsolutions.com/jts/jtshome.htm. [54]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - SUPPORTED

Apparent strengths

- Spatial Data Integrator provides the ability to write custom transformations using the Java Topology Suite, which supports two-dimensional simple vector features.

Apparent weaknesses

- Spatial Data Integrator is not able to output GML, hence adapting it for INSPIRE transformations would require significant development effort.

5.3.5 Radius Studio

Contact organisation

- 1Spatial Group Ltd.

Version under consideration

- Version 2.1.0.15

Description

Radius Studio is an enterprise spatial data integration platform that enables users to rapidly analyse scattered geospatial data to assess their quality and content. It does so by allowing users to collaboratively define and apply business rules to measure and maintain geospatial data quality. Radius Studio provides data mining, rules-based conformance checking, data cleaning and re-engineering capabilities that facilitate data transformation and reuse. It is a scalable solution that support grid processing.

References

- www.gsdi.org/gsdiconf/gsdi11/papers/pdf/283.pdf. [55]
- http://www.1spatial.com/products/#1265028216640_1/5 [56]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - SUPPORTED

Apparent strengths

- Radius Studio has a collaborative editing graphical user interface for defining rules and actions which can be configured by non-programmers.
- As part of the Radius Studio platform, 1Spatial has developed a dedicated XML grammar to make possible the exchange of mathematically rigorous schema transformation models, based on first-order predicate logic and using an approach that is conceptually similar to SWRL, but with the addition of support for spatial operators. This language is known as SQUIRL (Spatial QUality Integration Rules Language). The language is expressed in XML syntax and is thus portable to other XML-based formats and is Web friendly.

Apparent weaknesses

- Mapping language is proprietary hence it is not possible to exchange mappings with other tools or use mappings developed elsewhere. 1Spatial is contemplating the release of this mapping language as a public standard under the stewardship of the open source community.

5.3.6 Deegree Web Processing Service

Contact organisation

- lat/lon GmbH / University of Bonn Department of Geography, GIS Unit

Version under consideration

- Version 2.3

Description

Deegree is a free and open source implementation of OGC and ISO standards concerning Web-based GIS processing. The Deegree distribution includes a Web Processing Service. This provides supporting capability for developer-defined processes. The processes themselves require to be written, in Java, by implementors. A Java class needs to be extended to implement an abstract `execute()` method which takes a map of input parameters (potentially source and target schema references) and an output definition (which could correspond to a model mapping in the INSPIRE TNS context). Since it uses Java, the WPS can thus perform any kind of processing in principle. It is thus suitable for integration with a transformation technology such as XSLT or any of the model mapping languages discussed in this chapter.

References

- http://www.deegree.org/. [44]

- http://download.deegree.org/deegree2.3/docs/wps/html/deegree_wps_documentation_en.html . [45]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - SUPPORTED

Note: However, these features would require to be developed as an implementation of the WPS and integrated into the server.

Apparent strengths

- It is possible to implement any form of processing, including schema transformations, using the Java programming language or other technology such as XSLT incorporated programmatically via Java.

Apparent weaknesses

- Development of schema transformations, while possible, requires significant development effort.

5.3.7 Humboldt Alignment Editor / Conceptual Schema Transformer

Contact organisation

- Fraunhofer Institute for Computer Graphics, Darmstadt, Germany

Version under consideration

- Humboldt Alignment Editor & Conceptual Schema Translation Service / 1.0.0-RC1, 2.0.0-M1 respectively

Description

The Humboldt Project aims to facilitate and support cross-national spatial data harmonisation. It has given fresh contributions to the process of modelling data to support harmonisation, and the discovery of web services providing harmonisation capabilities. In addition to this, there has been the development of a toolchain, the Humboldt Alignment Editor (HALE) which is a user interface to enable design of schema mappings by data experts without needing to understand the underlying mapping language format. Also, the Conceptual Schema Translation Service (CST) has provided an engine for schema transformations. The project makes extensive use and re-use of existing open source software to further interoperability and standardisation of interfaces.

References

- Humboldt Project web site. [30]
- Humboldt application scenarios. [68]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED

- Level 4: Complex derivation - SUPPORTED
- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation - SUPPORTED

Apparent strengths

- The project, as a whole, is very aware of its context within INSPIRE domain, and as an example the application scenarios concentrate on specific challenges of INSPIRE data integration.
- Combination of server and user interface tools makes task of defining mappings easier.

Apparent weaknesses

- Ontology Mapping Language, selected as core mapping format, is not from a publicly recognised standards body and no longer under development. This choice may make difficult interoperability with existing SDI toolsets.
- Capability to handle large volumes of data and high-demand deployments is unknown.
- Software is made available as open source downloads under a free licence which permits its integration in proprietary toolsets (see [66]). This makes it vulnerable to being subsumed in commercial offerings rather than forming a platform for open, community-driven development.
- The project is targetted for completion in summer 2010.

5.3.8 AuScope Limited / AuScope Grid

Contact organisation

- AuScope Limited, University of Melbourne, Australia

Version under consideration

- Not specified

Description

AuScope Limited is a project coordinated by leading Australian research organisations to develop a national earth science infrastructure program. As part of this project, AuScope Grid is seeking to federate access to earth sciences datasets and facilitate development of toolsets based on them. This includes integration of data between disparate themes such as water, environment, air and spatial data. It is designed to support both cartographic and other spatial data presentation methods (e.g. drill-hole cores). The AuScope solution aims at seamless, Web based interoperability based on a publish/find/bind model, is XML-based and makes use of OGC and ISO standards, as well as the GeoSciML GML application schema, to define standardised interfaces for coordination, capture and manipulation of data.

References

- AuScope web site. [69]
- Introductory article on AuScope. [70]

Evaluation

Participation in tool vendor survey reported the following capabilities:

- Level 1: Basic renaming of attributes and instances - SUPPORTED
- Level 2: Basic derivation - SUPPORTED
- Level 3: Aggregation - SUPPORTED
- Level 4: Complex derivation - SUPPORTED

- Level 5: Multiple derivation - SUPPORTED
- Level 6: Model conflation and generalisation – NOT SUPPORTED

Apparent strengths

- Able to support terabytes of data volume.
- Use of open source software reduces potential cost of deployment.
- Standards based design principle makes interoperability with third party systems more feasible.
- Geo-science focus and support for non-cartographic modelling is of great interest for INSPIRE Annexe II and III data themes (e.g. geology, soil, atmospheric data).

Apparent weaknesses

- The project focusses on one country - albeit a large and diverse one - and therefore does not encounter the same trans-national (e.g. linguistic, historic) challenges that INSPIRE does.

5.4 Summary of Transformation Tools Analysis

The table of results from evaluation of transformation tools is as follows.

Note:

1. These tools have not been given gradings in order to avoid what might be perceived as a purely subjective conclusion.
2. These findings are based on the responses provided and is not necessarily an indication of suitability for use within the context of INSPIRE schema transformation services. Any claims will need to be verified further with tool vendors.

Tool Distributor / Name / Version	Current Support Claimed For Transformation Levels	Future Support Claimed For Transformation Levels
Interactive Instruments / XtraServer / 3.2	1-6	
Safe Software / FME Server / 2010	1-6	
Snowflake Software / GO Publisher / 1.4	1-6	6
Talend / Integration Suite Enterprise Edition / 3.2.3	1-6	
1Spatial / Radius Studio / 2.1.0.15	1-6	
OSGeo / Deegree Web Processing Service / 2.2	1-6	
Humboldt Project / Humboldt Alignment Editor & Conceptual Schema Translation Service / 1.0.0-RC1, 2.0.0-M1 respectively	1-6	
AuScope Limited / AuScope Grid	1-5	

Generalisations of key aspects of the responses to the survey are described below.

- This survey strengthens the argument that there are no widely used standards for schema descriptions or model mappings, highlighting the importance of this project to achieve interoperability between transformation service implementations. Most respondants use some form of proprietary language for schema descriptions and model mappings.
- Most respondents claimed support for all levels of transformation functionality identified. This is significant since it implies that transformation services should be feasible for the INSPIRE project, with a wide choice of potential tool vendors. Additionally, it permits the Technical Guidance to select a model mapping language that is sufficiently expressive to describe all mappings that are likely to be required. There is no need to simplify the functional requirements of a transformation service (so that it can be implemented more easily), at the expense of being able to use it on a wide variety of source data.
- There is a wide variety of supported input and output data formats. Most tools support GML, Oracle Spatial and ESRI Shape files.
- The mapping definition process varies from intuitive user interfaces to editing structured text files.
- Most tools have optional support for syntactic validation of the model mapping, to ensure that it meets the rules for the target schema. Few support semantic validation to ensure that the target data is fully compliant with the data specifications.
- Most tools are capable of running in a wide variety of environments, including combinations of Linux, Windows, 32bit and 64bit. Desktop application and batch processes are frequently provided, with some tools providing web accessible interfaces.
- The majority of tools claim support for scalable processing, including processing of multiple simultaneous requests. As a result, it is likely that the performance requirements defined in the implementing rules will be met by many tools.

6 Enterprise Architectures

An important aspect of a transformation network service is consideration of how it fit into the overall architecture and any impact that this may have on the request and response parameters or functionality of the service.

The INSPIRE Directive [1] Article 11 states that transformation services (in keeping with other INSPIRE services) should be "*available to the public and accessible via the Internet or any other appropriate means of telecommunication.*" The Draft Implementing Rules [2] propose several potential service architectures for transformation services. Each of these has various strengths and weaknesses, depending on the technical decisions made for the transformation services and also the business processes in which they will operate. In this State of the Art Analysis, we document the pros and cons of these service architectures.

6.1 Service Architecture Evaluation Criteria

Each of the service architecture options is evaluated where possible against the following criteria. Where non-functional influences may be significant, these are highlighted.

- Performance
- Scalability
- Reliability, resilience and availability.
- Flexibility
- Extension of functionality
- Cost

Additional factors that may be relevant when deciding the most appropriate service architecture are listed below. These are tied into the consideration of the deployment architecture, i.e. where each service will run and who will be responsible for maintaining it, which is not within the scope of this project. Therefore, they are not included in the evaluation.

- Platform infrastructure and deployment
- Service monitoring
- Maintenance, installation, Testing, upgrade
- Localisation
- Network traffic/bandwidth
- Service Level Agreement and contractual management
- Security

6.2 Architecture Options

6.2.1 Independent Service Node

This is illustrated in Figure 4, from the Implementing Rules [1].

In this case, an INSPIRE application first calls the TNS to transform the query, then a Download Service to obtain the data, then finally the TNS again, so that the result set can be transformed into the standard format.

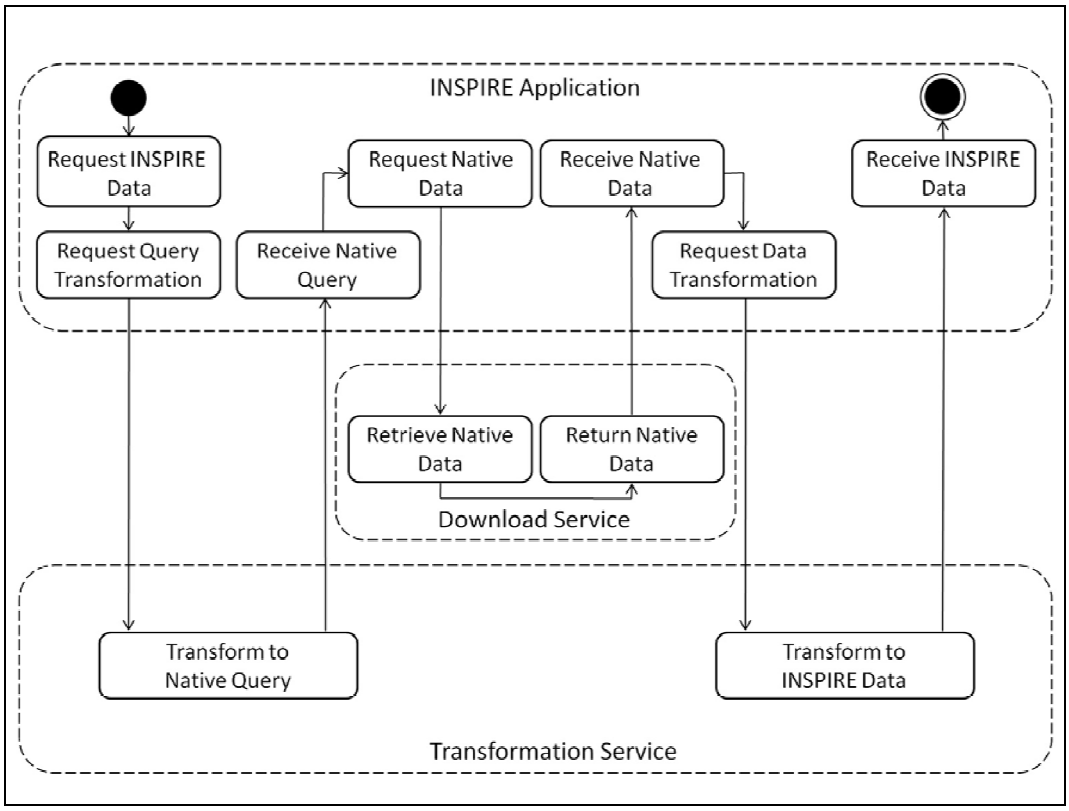

Figure 4 - Event sequence for independent service node (from [2]).

Strengths

- Flexibility. From a client perspective: the client is able to modify the process in any way it wants to (however, for transformation service extensibility see under Weaknesses).

Weaknesses

- Complexity in client application. This will increase costs for all client applications.
- Query transform complexity. The client requires some knowledge of both the INSPIRE and source schema and the ability to initiate a transformation of queries, which may be non trivial.
- Performance. Large quantities of data are being shipped between the client and different services. If the client is not co-located with the transformation and download services (which will frequently be the case), then this may cause significant issues with both throughput and latency.
- Scalability. Due to the ad-hoc orchestration being performed by the INSPIRE application, it will be difficult to ensure that the services scale appropriately to meet demand.
- Reliability. The transformation service must be able to perform fully automated transformation. Initial reports from INSPIRE transformation testing indicate that the process will frequently require both automated and manual processes in order to produce data that is fit for purpose. The additional complexity will also hinder reliability.
- Extensibility. In order to accommodate diverse client applications, the extensibility of the transformation service will need to be limited behind well-defined interfaces.

6.2.2 Tightly coupled proxy facade

This is illustrated in Figure 5, from the Implementing Rules [2].

Here a client calls the TNS and there is no direct access to the content access service. The TNS therefore orchestrates any combination of calls to relevant content access services (e.g. WFS and WMS) to fulfil the request.

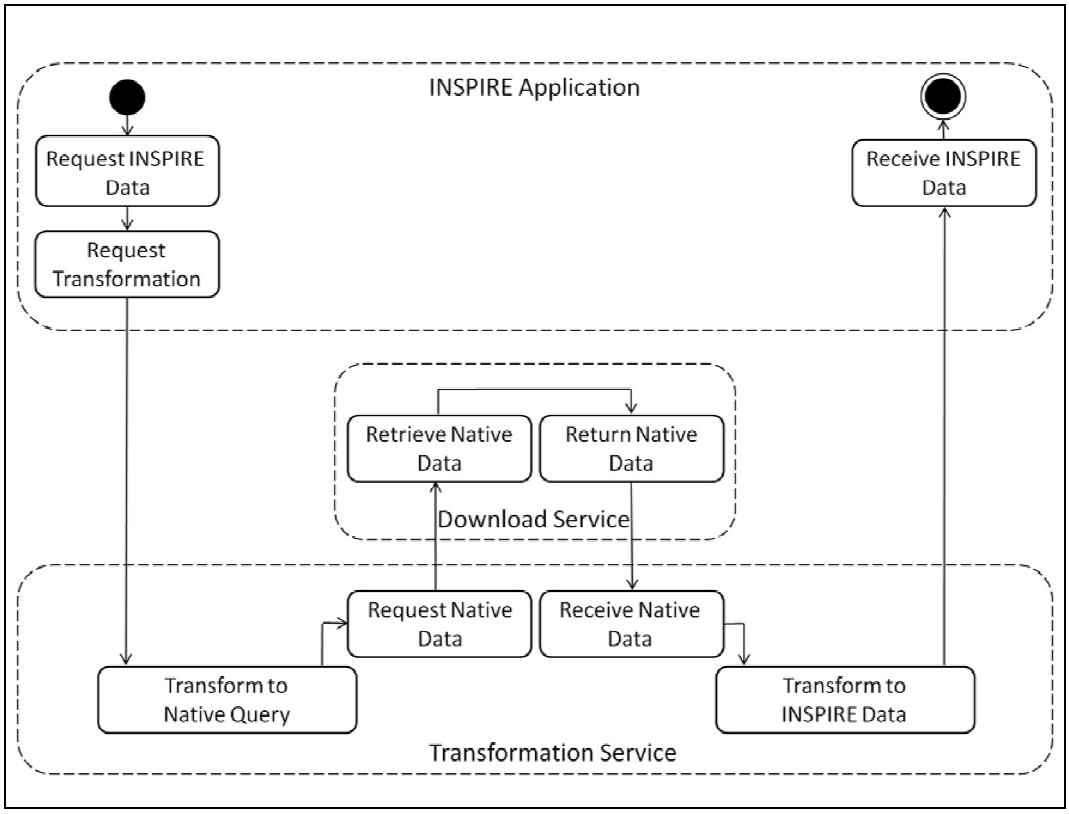

Figure 5 - Event sequence for proxy facade (from [2]).

Strengths

- Client applications will be much simpler to implement.
- Performance and scalability are likely to be better than the previous option, assuming that the transformation service and download service are co-located.
- Data providers are solely responsible for defining the required transformations and configuring the download service to use these transformations. This reduces exposure of the data provider's internal schema.
- Flexibility & Extensibility. Transformation service can be implemented in many ways as it has control of the process flow. For example, it can be modified to accommodate different download data sources and file formats.

Weaknesses

- Performance. Transformations are performed every time the data is requested.
- Reliability. The transformation service must be able to perform fully automated transformation.
- There is no opportunity for quality assurance and so the resultant data may contain errors.
- Reliability of the download service may be compromised by close coupling with the transformation service.

6.2.3 Encapsulation in Other Services

This is illustrated in Figure 6, from the Implementing Rules [2].

Here a client will query for data from a content access service (e.g. download service), specifying the input and output format and mapping required. The service will compile the dataset and then invoke the TNS on the way back, so that the content received by the client is in the standard format. In this use case, the actors are the target content access service and the TNS. There is no direct call to the TNS by the client.

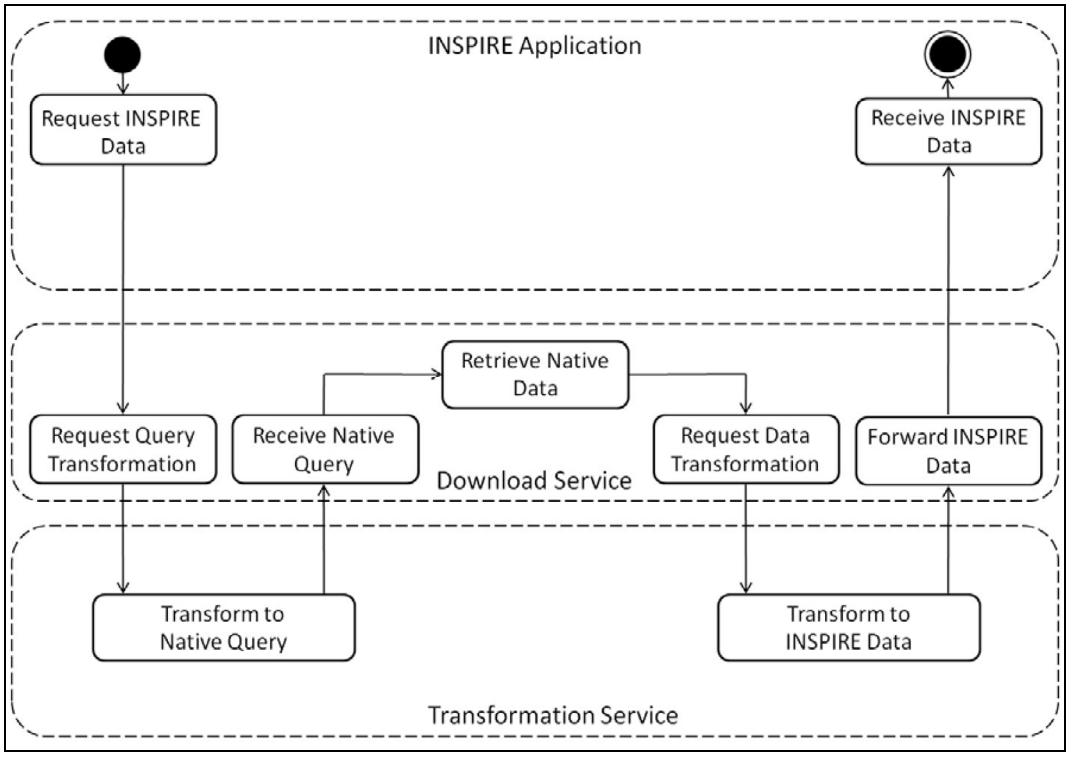

Figure 6 - Event sequence for encapsulated service (from [2]).

Strengths

- These are generally the same as the Tightly coupled proxy facade.

Weaknesses

- All those listed for the proxy facade, plus …
- The download service must be implemented / modified to have knowledge of the transformation service. In many cases, a download service might not need to use a transformation service, so this is adding additional complexity and reducing the possibility for re-use of standard components.
- Transformation service will be restricted in its extensibility to features that the download service already supports.

6.2.4 Bulk Transformation & Caching

This is an alternative approach to transformation services that is not documented in the implementing rules. Transformation of source data to the INSPIRE schema is pre-computed and the transformed data resulting from this is then cached in some form of database. The download service and client applications communicate with this data cache in the INSPIRE model and do not need to know about the existence of data in any other format. NB the bulk transformation service is considered as an online service in which the first requester will incur the penalty of the additional time required to wait for the results of the transformation, and subsequent requesters will received the pre-transformed dataset.

Strengths

- Performance - There will be a significant reduction in repeated transformation activity, since data will be transformed once, rather than every time it is requested.
- Performance - Since data is already cached in the INSPIRE model, there will be a lower latency for accessing data, therefore all INSPIRE applications will be more responsive.
- Scalability - The data can be cached using proven scalable technology.
- Flexibility & Extensibility - Data providers will be free to define their own business process for their interactions with INSPIRE. For example, this could include integration of data inspection stages to ensure that the transformed data is semantically valid in the INSPIRE schema, before it is made available to client applications (in general, this could be a large issue for INSPIRE since definition of transformation mappings is dependent on knowledge of the semantics of the source data, which may change over time). Additionally, data providers will be able to choose the most appropriate transformation update process to fit their data update cycles, for example transforming portions of frequently-changing datasets whenever the source data changes, or transforming infrequently changing datasets periodically (e.g. monthly).
- Cost: client applications and intermediate services will be simplified significantly. They will only have to handle the INSPIRE schema.
- There will be no necessity to translate a request query from the INSPIRE client application into domestic formats.

Weaknesses

- Since the data is cached, it may not always be up to date. This is dependent on the data provider's business process for performing the transformation and updating the cache, so is not necessarily an issue. INSPIRE mandates that any change in the source data must be reflected in the INSPIRE data within 6 months of the change being applied.
- There is a potential difference in the deployment architecture, since an additional dataset must be maintained in the INSPIRE schema, rather than the data being passed transiently through the network. Depending on the overall deployment architecture, this could be either a weakness or strength.

6.2.5 Service Architecture Summary

The above sections have identified several strengths and weaknesses of different approaches to the integration of transformation network services. Selection of the appropriate set of functionality to be supported by a transformation network service will be heavily dependent on the non-technical decisions identified above, primarily the decision as to where the data transformation is performed at the data provider's site, at the data user's site, or by a third party. In order to achieve a practical and scalable implementation of the overall INSPIRE service architecture, it is vital that these issues are considered and resolved as soon as possible.

For the current Technical guidance, it is assumed that guidance will be required for each of the options proposed in the Implementing Rules, even though it is unlikely that transformation services will support each of these options. These may have external dependencies, for example regarding communications with a download service; in such cases, the Technical Guidance will refer to external INSPIRE Technical Guidance documents, or related aspects of the overall INSPIRE architecture.

6.3 Considerations for Request and Response Parameters

In addition to the required functionality, schema descriptions and model mappings, several other aspects must be considered when implementing transformation network services. This section identifies some areas that might affect the request and response parameters. These will be analysed in more detail in the next stage of the project as the Technical Guidance develops.

6.3.1 Feature Data Transfer

It is necessary to share data in order to provide the TNS engine with the inputs it needs to perform a transformation. Since this data is made available across the network and interoperability requires platform neutrality, there should not be any 'hidden' interfaces (e.g. creating a system where two web services share a single database to store their state).

This means that all data must be made available as parameters to Web Service calls. As noted earlier, parameters can represent pure data or, alternatively, references to sources where the data itself can be obtained (i.e. indirect references). It is preferable that pure data be passed as often as possible (within performance and feasibility constraints), because each time indirection is introduced, it requires another service to be made available to convert reference parameters into the actual data, and thus the system becomes more complex. In addition, the system is architecturally stronger with data passed in-line since this provides a greater separation of concerns between the services within the INSPIRE architecture - the transformation service will be responsible only for transforming data.

Many current examples of Web Service implementations and standards are based on the finance industry, which tends to exemplify numerous, small transactions such as stock tickers. However, spatial datasets tend to be very large, meaning that the process of modelling Web Service interfaces to enact spatial data operations is challenging with regard to feasibility and performance.

Methods do exist for processing Web Service operation parameters efficiently, e.g. compression and use of sophisticated binary parameter transmission techniques such as MTOM/XOP that embeds the raw bytes of parameter data in the SOAP message rather than using inefficient base-64 encoding.

6.3.2 Persistence Architecture

Persistence in one sense relates to how the data is held: permanently or transiently.

Consideration will have to be given to the role of persistent storage in the system; how many storage nodes are required; what data is being stored and how often it must be updated. Business processes will make an important contribution to this aspect of the design, in particular the definition of the required data lifecycle. The persistence architectures could be very different depending on whether transformation is performed separately by each data provider, by the client or in a centralised data processing environment. We await feedback from data providers as input for the Technical Guidance in this area.

6.3.3 Metadata Repository

In addition to feature data, there is a requirement to maintain descriptions of source schema and model mappings. In order to achieve interoperability, it may be best for this to be maintained in a centralised repository that is independent of the TNS. This would then make schema descriptions and model mappings available via a Web Service interface and hosted on the Internet. The Technical Guidance will consider options for such a repository and, if necessary, investigate registry providers. There are a number of existing production metadata registries and standards for accessing these, such as OASIS ebXML.

6.3.4 Service Discovery

If parameters are passed by-reference, then the transformation network service will be required to resolve this reference. This will typically involve finding a suitable endpoint from which the data can be received (e.g. a specific instance of a download network service), handling credentials and retrieving the referenced data.

Several options exist for handling such references, including passing specific details (URLs, username and password) directly within the TNS parameters and passing only resource indicators in the parameters, which must then be resolved to specific instances through the use of an intermediate service. This may increase complexity of the service, but should improve reliability. Handling of credentials will be discussed as part of the investigation into security.

6.3.5 Rights Management

The INSPIRE Directive [1], article 11(1)(c), refers to rights management. While INSPIRE does not mandate any paid access to transformation services, the content that is to be transformed may have special rights restrictions placed upon it.

When implementing a TNS, we therefore need to cater for a seamless mode of security for request processing that enables dynamic single-step rights management, e.g. via management of security credentials and authentication/authorisation tokens at the entry-point to the TNS (so that no further challenge to the caller is made, or an appropriate exception is sent back when the authentication/authorisation fails).

It is therefore necessary to consider use of standards relevant to authentication and authorisation. OASIS offers the following standards that could apply in an INSPIRE network services context:

- Security Assertion Markup Language (SAML) v. 2.0.
- eXtensible Access Control Markup Language (XACML) v. 2.0.

In order to achieve interoperability, the schema transformation Technical Guidance will identify that transformation services may need to support some form of rights management and possibly rights management for additional services that they invoke. Specific details of the rights management approach is to be provided by the INSPIRE network architecture guidance.

6.3.6 Multi-Lingual Aspects

INSPIRE exists within a European context where exchange of data and concepts occurs not just in multiple computer languages but in many natural (i.e. human) languages as well. The European Union has 23 official languages. The EU does not impose a language policy on its member states. This means that few assumptions can be made about how information will be stored in terms of the natural language used to express feature and attribute labels, metadata and other textual components of instance data. For example, Bulgaria has been a member of the European Union since 2007. Its official language, Bulgarian, is generally written using the Cyrillic alphabet, which it shares with Russian (see the following EUR-Lex link for a flavour of the differences [64]). It is very probable that Bulgarian geographical data will generally be encoded using Unicode characters rather than ASCII-compliant UTF-8 character format, which was developed initially with reference to US English. This results in technical challenges. XML is capable of containing unicode characters but other languages may not be. See [65] for the W3C report on how Unicode should be supported in XML documents. Attention is also drawn to INSPIRE Generic Conceptual Model Section 11 [66] although this is possibly more applicable to natural language transformation than schema transformation.

7 Conclusion

This report has identified and analysed important factors to be considered in the definition of request and response parameters for transformation services. These will be used as inputs to the process of creating the Technical Guidance for INSPIRE transformation services. The Technical Guidance will include a concrete interface specification and detail any other characteristics required by such transformation services. As a result of sharing a common interface, this will allow interoperability between different vendors' implementations of transformation services relating to the INSPIRE project.

Further details of the business processes under which the transformation services operates will be vital to the success of INSPIRE transformation services. This includes details of the lifecycle of data from capture in the source schema to use in the INSPIRE schema, particularly including responsibilities and physical connectivity for any data processing. Examples of potential business processes include the data provider automatically transforming data when it changes, a third party transforming data periodically, or an end-user accessing a transformation service directly when needed. Details of the characteristics of the transformation service, for example how it accesses data or how it supports rights management, are heavily dependent on knowledge of this business process. Several technical decisions relating to the transformation service need to be based on details of the specific operating environment. This will allow the creation of transformation services that are performant, reliable, resilient and secure.

Regardless of the operating environment, the core functionality of a transformation service is to construct data in the INSPIRE schema from a wide variety of source schemas. In order to achieve this, the transformation service will require details of the source and target schemas and a description of the model mapping that is to be performed between instances in each of these schemas. This document has identified that there are no common standards for such descriptions, but has analysed a number of de jure standards and de facto tools with proprietary mapping descriptions that may be relevant to a transformation service. A survey of existing tools with transformation capabilities has identified that there are already many tools that can perform highly expressive transformations. The selection of a common description for these transformations should therefore result in a system that is implementable by several tool vendors, whilst still being sufficiently expressive to represent transformations from the vast majority of data providers' source data to the INSPIRE schema.

Ideally, the schema descriptions and model mappings used in the parameterisation of the INSPIRE transformation services should be based on open standards. Of these open standards, there currently appear to be two groups of schema description and model mapping languages that provide the highest level of compatibility with our requirements: UML/QVT and OWL[2]/RIF. The Technical Guidance document will identify which combination of standards are best placed to meet the specific needs of geospatial transformation to the INSPIRE data specifications and if necessary suggest how these should be customised or used in this domain.

[2] RIF, which is compatible with OWL, is the candidate, not OWL itself. However, RIF was developed by the same standards organisation (W3C) as OWL and is a partner with it in the Semantic Web domain. RIF is able to interchange with RDF and OWL. It is different from UML/QVT, because UML is actually part of the dependencies for the QVT specification along with OCL and MOF.

Appendix A – Schema Transformation Tools – Vendor Survey

Here follows the introductory text and questions used to conduct the survey of existing data model transformation tools, from the vendors' perspective. This was supported by a discussion document on schema transformation levels [see Appendix B].

JRC project vendor survey

As part of the INSPIRE initiative to harmonise spatial datasets across Europe, RSW Geomatics, 1Spatial and Rob Walker Consultancy are preparing a "state of the art" analysis of schema transformation tools and technologies for the Joint Research Centre (JRC).

An overview of the project can be found at http://www.1spatial.com/research/index.php?res=E&res1=9#1264998176949_4/4.

The project involves assessing tools that are capable of transforming datasets in various formats and encodings, containing both spatial and non-spatial data, into the GML format in the common INSPIRE schema. This survey will provide guidance input to the "state of the art" analysis, providing an opportunity for your software to gain important visibility as a schema transformation solution. Please answer the following questions which will help us assess the domain and the available solutions. Please note that we will include your responses as an appendix to our "state of the art" analysis.

How to complete this survey

Some of the questions require a tick in the appropriate box. In other questions, there is space for a free-text reply. Please provide relevant links to your website containing further information. Please only supply publicly available information in your replies.

The survey contains 28 questions spread over five pages.

Responses received by Friday, 22nd January 2010 are most likely to be able to be considered in relation to the "state of the art" analysis, although we will endeavour to consider any responses we receive after that date.

Should you have any questions or wish for clarification of either the content or the context to this survey, please email jrc-tns@1spatial.com.

Questions

General Information

1. What is the name of your organisation?

   _____

2. Whom may we use as a named contact for your organisation?

   _____

3. What is the name and most recent version of your schema transformation software?

   _____

Functional Aspects

4. What input and output data formats/encodings does your software support (please indicate which "routes" - e.g. Oracle to GML - are currently supported, and which are in development or are planned for the future)?

_____

5. Is your software able to map data to a target model that is not aligned with the source model (i.e. where the feature types/tables/columns/attributes etc. in the source model differ from those in the target model)?

YES / NO

6. If you answered YES to question 5, please give the details of any recognised standard language that is used to describe/store the mapping.

_____

7. If you answered YES to question 5, how do users define a mapping (please link to screen shots if possible)?

_____

8. Does your software perform any automated configuration of default initial mappings?

YES / NO

9. What level of extensibility is available to users, e.g. an API or framework to enable development of custom model mappings?

_____

10. We have defined a set of transformation functionality levels [see Appendix B] relating to the ability to perform transformation where the source and target schema are not well aligned. Please would you select those levels of functionality, and hence schema re-alignment, your software supports and/or is planned to support in future.

Transformation level:

Level 1:	Renaming
Level 2:	Basic derivation
Level 3:	Aggregation
Level 4:	Complex
Level 5:	Multiple feature derivation
Level 6:	Model conflation and generalisation

11. Do you support any other styles of transformation that may be relevant for INSPIRE? See the transformation functionality levels [see Appendix B] for the levels we have identified.

_____

12. Does your software verify and and/or validate the results of a schema translation or model transformation (e.g. does it verify that target format schema rules are obeyed by the resultant data and that the output dataset is semantically valid)?

YES / NO

13. If you answered YES to question 12, how does your software perform the verification and validation?

_____

14. Can your software handle de/compression of datasets? (Please tick the boxes that apply)

Input Decompression?

Output Compression?

15. If you answered SUPPORTED to either part of question 14, what compression formats does your software support?

_____

16. In what environments is your software capable of running (e.g. hardware platform, OS, Java or .Net version, or other dependencies)?

_____

17 Once a transformation has been defined, what modes of operation are supported (e.g. desktop application, batch mode, Software-as-a-Service (SaaS), SOAP web service, cloud computing, etc.)?

_____

18. Does your software conform to any internationally recognised, open standards for file format conversion, schema transformation, transformational model mapping or semantic rule definition?

YES / NO

19. If you answered YES to question 18, which standards are supported?

_____

20. Do you provide an open source distribution of your software?

YES / NO

21. If you answered YES to question 20, please give more details.

_____

Non-functional Aspects

22. How scalable is your software, in terms of support for multiple execution nodes, and synchronous threads of execution?

_____

23. Can your software support 5 concurrent requests or more?

YES / NO

24. What is your largest current tested installation of the software, measured in Gigabytes of feature data, numbers of layers/features per layer and simultaneous threads of execution?

_____

25. Do you have any processing benchmark figures for performance per feature?

YES / NO

26. If you answered YES to question 25, please give more information.

_____

27. How many copies of the software have you distributed? (Please tick the appropriate box)

1-50

51-500

501-5000

5000+

28. Do you have any other comments you would like to add?

_____

Appendix B – Schema Transformation Levels

To aid discussions of transformation functionality, the following capability levels have been defined. These describe different types of functionality that will be required in order to transform schema of varying complexity. When the source schema is closely aligned to the target schema, a lower level of transformation functionality will be required.

Each level incorporates all functionality from earlier levels, i.e. if a transformation service supports functionality in level n, it should also support all functionality in level n–1. The levels are:

- Level 1 - Renaming classes and attributes.
- Level 2 - Simple attribute derivation.
- Level 3 - Aggregating input records.
- Level 4 - Complex derivation and dynamic type selection.
- Level 5 - Deriving values based on multiple features.
- Level 6 - Conflation and model generalisation.

Level 1 – Renaming classes and attributes

The transformation service will be capable of renaming any attributes and classes, given a fixed source name to target name mapping.

The structure of the source and target schema needs to be closely aligned in order for this level of schema transformation functionality to be sufficient.

Level 2 – Simple attribute derivation

Attributes in the target feature may be derived from attributes in the single related source feature using basic derivation functionality. The transformation at this level is always intra-feature, i.e. a target feature can be created by examining a single source feature in isolation.

Types of attribute derivation at this level include:

- Transforming data types (e.g. numbers into text or strings into timestamps)
- Transformation based on basic geometric functions (e.g. bounding_box, convex_hull, area)
- Transformation based on non-spatial functions (e.g. uppercase, truncate, substring, round, regular expression)
- Transforming units of measure.
- Setting default values where data is not supplied.
- Replacing values based on lookup tables (e.g. code lists)
- Entering identifiers for referenced objects (e.g. based on GML xlink or relational database foreign key in the source data).

Mapping of identifiers of related objects may be a complex operation, depending on the source storage format. It is however an important aspect of schema translation which is required in most cases. It is therefore included at level 2 even though it may not always follow the general pattern of only interrogating the source feature in order to produce the target feature.

It is expected that a transformation service may support only a subset of this functionality. This will need to be clearly expressed through the service's capability metadata.

Note that further logical combinations of attribute derivation functions are supported by level 4.

Level 3 – Aggregating input records

Permits features to be built up as individual feature objects in the target feature based on multiple storage records in the source schema. This is commonly required when the source data is stored in a relational database, which results in data being distributed throughout the database. Data is often split into multiple records in the source database to allow:

- Multiple cardinality attributes (stored as several rows in a table in the source database, but just as attributes in the target feature).
- Efficient storage of data in multiple tables (e.g. a primary table for frequently accessed data and secondary tables for component data, or potentially voidable data. These would be transformed into in-line complex properties in the target feature).

At level 3, this is restricted to the specific case where each group of source records matches a single target feature, i.e. there is a 1 to 1 mapping between logical features in the source and target schema. The splitting of features in the source schema is assumed to be solely due to the capabilities of the storage mechanism rather than related to the structure of the object model. Level 3 effectively describes a basic Object Relational Mapping (ORM) system.

The transformation service must be capable for determining which source records are required for each target feature and efficiently querying the source data to obtain this.

More complex aggregation is introduced in level 6.

Level 4 – Complex derivation and dynamic type selection

This is an extension of level 2 – simple attribute derivation, such that the value to use in the target attribute may be chosen by evaluating logical operations involving attributes in the source features or attributes derived from the source features. In effect, these permit different derivation functions to be applied conditionally based on inspection of the source feature.

In addition, this logical evaluation functionality may be used to determine the target class, based on attribution in the source feature. A single source feature class could be mapped onto several different target feature classes, depending on the value of attributes in that class.

The new derivation types and logical operators introduced at this level include:

- Conditional tests (if… then… else…)
- Testing scalar relationships (e.g. equals, less than, greater than)
- Logical combinations of tests (e.g. or, and, not)
- Evaluating simple derivation rules (from level 2)
- Looping over attributes or derived values (foreach element in array, foreach part in complex geometry)

These logical statements may be applied to any combination of values in the source feature in order to derive a value in the target feature. In the case of multiple-cardinality attributes, these logical statements can be used to determine when additional instances of the attributes should be created. This may be needed if a source feature contains a list of n items, but the target feature is required to contain a list of m items. The is removing the limitation of support for multiple-cardinality attributes in level 3 whereby all items in a multi-cardinality attribute are copied from source to target feature.

It is expected that a transformation service may support only a subset of this functionality. This will need to be clearly expressed through the service's capability metadata.

Level 5 – Deriving values based on multiple features

Levels 2-4 have allowed the creation of an output feature based on a single source feature. At level 5, contextual information may be used to derive values for the target feature.

This may include data from additional source features that are found by:

- Evaluating spatial relationships (within, overlaps, intersects)
- Evaluating referential relationships (e.g. foreign keys, or related by identifiers)
- Universal qualification (e.g. for all features x then….)
- Existential qualification (there exists a feature x such that…, …).

Each of these may be qualified using any of the features from earlier levels and may be used as part of the logical derivations from earlier levels.

At level 5, the transformation is no longer intra-feature. The transformation service will be required to query the source data to find any related features that need the be considered based on the mapping definitions.

Level 6 – Conflation and model generalisation

Target features may be created from multiple source features. Multiple target features may be created from a single source feature. It is possible that source features relating to one target feature may be in multiple classes, or even multiple source schema.

Examples include:

- Creating separate features representing polygons and boundary lines, or polygons and position points from a single source feature
- Creating a target feature from a source feature in schema A with additional attribution from a separate feature in source feature B
- Creating a single target feature from two coincident source features
- Generalising multiple touching source features into a single, larger target feature.

There is not necessarily a direct link from an identifiable source feature to a single identifiable target feature.

Appendix C – Terms & Definitions

Acronym / Term	Definition
Application Schema	A platform-specific description of the structure and constraints applicable to the storage of data for one or more applications (expressed, for example, as an XML Schema (XSD)).
BPEL	Business Process Execution Language: a language permitting the description of synchronous and asynchronous orchestrations of Web Service interactions.
Business Rule	A business rule is a statement that defines or constrains some aspect of a business domain. It is intended to assert business structure or to control or influence the behaviour of the business. See also Production Rule, Transformation Rule.
CL	Common Logic: a framework for a family of logic languages, based on first-order logic, intended to facilitate the exchange and transmission of knowledge in computer-based systems, specified in ISO Standard ISO/IEC 24707:2007.
CLIF	Common Logic Interchange Format: a dialect of Common Logic, specified in ISO Standard ISO/IEC 24707:2007, which permits existential predicates, use of variables, conjunctions/disjunctions, negation and the definition of relations between objects.
Conceptual Model	A model that defines concepts of a universe of discourse.
Conceptual Schema	A platform-independent (or platform-specific), conceptual model expressed using a formal modelling language (such as UML).
CRS	Coordinate Reference System.
Data Instance	A single item of data expressed in a concrete storage format (for example, an XML element or database record) which corresponds in some way to an object in the real world such that it is capable of being expressed as an object in an ontology, rather than merely as a predicate or attribute of an object.
Data Model	A model of the (geographic) data that is stored and/or exchanged.
DDL	Data Definition Language: this commonly refers to a subset of SQL commands used for defining database schema objects, although it can be used in a generic sense to refer to any formal language for describing information storage structures.
DL	Description Logics: a knowledge representation formalism which underpins the OWL DL dialect. OWL DL supports those users who want the maximum expressiveness without losing computational completeness (all entailments are guaranteed to be computed) and decidability (all computations will finish in finite time) of reasoning systems.
Data Harmonisation	Providing access to spatial data through network services in a representation that allows for combining it with other harmonised data in a coherent way by using a common set of data product specifications.
DM	Data Model: an abstract software engineering model that describes how data are represented and accessed.
DOM	Document Object Model: a cross-platform and language-independent convention for representing and interacting with objects in XML and other markup language documents.
DSDL	Document Schema Description Language, a language that supports syntactical validation of XML documents, defined by ISO/IEC standard 19757.
ebXML	Electronic Business using XML: A modular suite of specifications that enables enterprises to conduct business between different locations over the Internet, using a standard method to exchange business messages.
GML	Geography Markup Language: the XML grammar defined by the Open Geospatial Consortium (OGC) to express geographical features.

Acronym / Term	Definition
Humboldt Team	An international, European project seeking to enable organisations to document, publish and harmonise their spatial information, and thus contribute to the implementation of a European Spatial Data Infrastructure.
INSPIRE	Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE).
INSPIRE Data Specifications	Harmonised data product specification for a theme adopted as an Implementing Rule [DS-D2.5].
INSPIRE Generic Conceptual Model	A conceptual model is a model that defines concepts of a universe of discourse [ISO 19101]. The universe in this case is the INSPIRE business context. This is the basis for all INSPIRE technical architecture and underpins the approach for using interoperability components. The basis for modelling individual themes from INSPIRE Annex I, II and III.
Instance Data	A collective term for data instances, also known as “row-level data” (especially in a database context).
Interoperability	The ability of diverse systems and organisations to work together. Possibility for spatial data sets to be combined, and for services to interact, without repetitive manual intervention, in such a way that the result is coherent and the added value of the data sets and services is enhanced [INSPIRE Directive].
IEC	International Electrotechnical Commission: the world-leading international electrical and electronic standards organisation.
ISO	International Organisation for Standardization: the world's largest developer and publisher of international standards.
LMO	Legally Mandated Organisations (LMOs) are all the Member States’ public authorities, institutions and bodies who already have or will get a legal mandate to set up and run one or some of the components of national and regional SDIs, and which are eligible to become the MS’ contributors to the INSPIRE for a particular component. These components cover all fields of activity targeted by INSPIRE and can be either of a technical nature, or of a policy and organisation related nature.
Logical Schema	Synonym for Application Schema.
MDA	Model-Driven Architecture: a software design approach for the development of software systems using models.
Metadata	Information describing spatial data sets and spatial data services and making it possible to discover, inventory and use them [INSPIRE Directive].
MOF	The Meta-Object Facility (MOF) Specification defines an abstract language and a framework for specifying, constructing, and managing technology neutral metamodels.
NSDI	National Spatial Data Infrastructure. An SDI for a nation. Typically, the mission of which is to help avoid duplication, or erroneous modification of spatial data (considered to be a national asset).
OASIS	Organization for the Advancement of Structured Information Standards: a not-for-profit consortium that drives the development, convergence and adoption of open standards for the global information society.
OCL	Object Constraint Language: a formal language used to describe expressions on UML models.
OGC	Open Geospatial Consortium: An international voluntary consensus standards organisation.
OMG	OMG™ is an international, open membership, not-for-profit computer industry consortium developing enterprise integration standards for a wide range of technologies and industries.
Ontology	A representation of terms in a given vocabulary and their interrelationships.

Acronym / Term	Definition
OWL	Web Ontology Language: a language for expressing ontologies.
Physical Data Model	Synonym for Logical Schema or Application Schema.
Physical Schema	The concrete, implementation-specific description of how the data is organised in the storage technology of choice (expressed, for example, as SQL DDL).
Production Rule	A rule that defines some aspect of the behaviour of a software model or system by specifying conditions to be asserted against a fact base (the "if") and associated actions to be executed if the assertion returned true (the "then"). See also Business Rule, Transformation Rule.
R2ML	Rewerse II Markup Language: an XML rule format that permits rule interchange and rule-based ontology enrichment, and provides tool support for working with these concepts.
RDF	Resource Description Framework: a graph-structured modelling language for data interchange on the Web.
RDFS	RDF Vocabulary Description Language: an application of RDF providing the means to describe specific vocabularies (i.e. namespaces) to be used when building RDF statements.
RIF	Rule Interchange Format (RIF) is a language for exchanging rules among rule systems, in particular among Web rule engines.
RuleML	Rule Markup Language: a Web language for rules using XML markup and formal semantics, designed to express both forward-chaining and backward-chaining rules. *Forward-chaining* means that a set of propositions are evaluated using inference rules, to deduce a logical conclusion; *backward-chaining* means that the propositions are derived working backwards from the conclusions and the inference rules.
QVT	A family of languages, defined by meta-models, designed to express transformations between object models.
Schema	A general-purpose term, rather imprecise in nature, that may refer to a generic data model, ontology, or database storage structure, depending on the context.
Schema Language	
Schema Transformation	The transformation of a query or data instance from one application schema to another.
SDI	An SDI (Spatial Data Infrastructure) is a framework of policies, institutional arrangements, technologies, data and people which enables the sharing and effective usage of geographic information.
Spatial Data	Means data with a direct or indirect reference to a specific location or geographic area [INSPIRE Directive]. NOTE The use of the word "spatial" in INSPIRE is unfortunate as in the everyday language its meaning goes beyond the meaning of "geographic" – which is considered by the Drafting Team as the intended scope – and includes subjects such as medical images, molecules, or other planets to name a few. However, since the term is used as a synonym for geographic in the draft Directive, this document uses the term "spatial data" as a synonym for the term "geographic information" used by the ISO 19100 series of International Standards.
Spatial Data Set	Identifiable collection of spatial data [INSPIRE Directive].
Spatial Object	Means an abstract representation of a real-world phenomenon related to a specific location or geographical area [INSPIRE Directive]. NOTE It should be noted that the term has a different meaning in the ISO 19100 series. It is also synonymous with "(geographic) feature" as used in the ISO 19100 series.
SQL	Structured Query Language: a language for defining relational database storage structures and creating, reading, updating or deleting data within those structures.

Acronym / Term	Definition
SWRL	Semantic Web Rule Language: a rule language combining elements of OWL DL and Lite with RuleML.
TNS	Transformation Network Service: a service that enables spatial data sets to be transformed with a view to achieving interoperability.
Transformation Rule	A rule that defines some aspect of the transformation to be applied to a model or fact base, in order to map source data or constructs to target ones. Transformations can be semantics-preserving (where the meaning of the source element is preserved when mapped to the target domain) or semantics-changing (where the meaning is lost but the content of the data is retained). See also Business Rule, Transformation Rule, Schema Transformation.
UML	Unified Modelling Language: a language designed for modelling software systems.
UoM	Unit Of Measure: a real scalar quantity**,** defined and adopted by convention, with which any other quantity of the same kind can be compared to express the ratio of the two quantities as a number.
W3C	The World Wide Web Consortium: an international community that develops standards to ensure the long-term growth of the Web.
WFS	Web Feature Service: an OGC interface standard allowing requests for geographical features across the web using platform-independent calls.
XCL	XML for Common Logic: a dialect of Common Logic designed as a concrete (serialization) syntax for Common Logic.
xlink	An element inserted into an XML document to create and describe links between resources (see the XLink specification [58]).
XMI	XML Metadata Interchange: an Object Management Group (OMG) standard for exchanging metadata information (for example, UML models) via XML.
XML	Extensible Markup Language: a simple, very flexible text format design for larg-scale publishing and data exchange on the Web.
XSD	The XML Schema Definition Language offers facilities for describing the structure and constraining the contents of XML documents and is itself represented in an XML vocabulary.
XSLT	Extensible Stylesheet Language Transformations: a language for transforming XML documents into other XML documents and character formats.

Appendix D – References

[1] *DIRECTIVE 2007/2/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE),* http://eur-lex.europa.eu/JOHtml.do?uri=OJ:L:2007:108:SOM:EN:HTML.

[2] *Draft Implementing Rules for Transformation Services*, v3.0, September 2009, http://inspire.jrc.ec.europa.eu/documents/Network_Services/INSPIRE_Draft_Implementing_Rules_Transformation_Services_(version_3.0).pdf.

[3] *INSPIRE Website*, http://inspire.jrc.ec.europa.eu/ .

[4] *INSPIRE Technical Architecture Overview,* http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/INSPIRETechnicalArchitectureOverview_v1.2.pdf .

[5] *INSPIRE Generic Conceptual Model,* http://inspire.jrc.ec.europa.eu/index.cfm/pageid/201/consultation/31107.

[6] JRC INSPIRE Transform Service Introduction, http://inspire.jrc.ec.europa.eu/documents/Network_Services/INSPIRE_Draft_Implementing_Rules_Transformation_Services_(version_3.0).pdf.

[7] *UML Infrastructure Specification v. 2.2,* http://www.omg.org/spec/UML/2.2/.

[8] *XMI Specification c. 2.1.1,* http://www.omg.org/spec/XMI/2.1.1/.

[9] *Meta-Object Facility Specification v. 2.0,* http://www.omg.org/spec/MOF/2.0/.

[10] *XML Schema Definition Specification v. 1.0*, http://www.w3.org/TR/xmlschema-1/ .

[11] *Geography Markup Language Specification v. 3.2.1*, http://www.opengeospatial.org/standards/gml.

[12] *RDF Schema Specification v.1.0* - http://www.w3.org/TR/rdf-schema/.

[13] *Web Ontology Language Specification v. 2.0,* http://www.w3.org/TR/owl-guide/.

[14] *XSLT Specification v. 2.0*: see www.w3.org/TR/xslt20/.

[15] *RELAX-NG:* see http://relaxng.org/spec-20011203.html.

[16] *W3C Overview of Rule Interchange Format,* http://w3c.org/TR/rif-overview/.

[17] *Introduction to RIF by Dr. Chris Welty of IBM,* http://www.w3c.org/2005/rules/wiki/images/b/b0/W3C_RIF-CW-0-09.pdf.

[18] *RIF Use Cases,* www.w3.org/TR/rif-ucr/#Vocabulary_Mapping_for_Data_Integration.

[19] *RIF Implementations*, www.w3.org/2005/rules/wiki/Implementations.

[20] *Oracle Business Rules,* www.oracle.com/technology/products/ias/business_rules/index.html.

[21] *W3C SWRL Specification*, http://www.w3c.org/Submission/SWRL/

[22] *Cautiously Approaching SWRL, Bijan Parsia; et al. (2005) (PDF),* http://www.mindswap.org/papers/CautiousSWRL.pdf.

[23] *OCL Specification v. 2.0,* http://www.omg.org/spec/OCL/2.0/.

[24] QVT Specification v. 1.0, http://www.omg.org/spec/QVT/1.0/

[25] *Universal Turing Machine in XSLT,* www.unidex.com/turing/utm.htm.

[26] *GeoXSLT,* http://www.svisj.no.fredrik/geoxslt.

[27] *RDF Concepts,* http://www.w3.org/TR/rdf-concepts/

[28] *OWL 2 Specification Overview*, http://www.w3.org/TR/owl2-overview/

[29] *OML Specification* http://www.omwg.org/TR/d7/d7.2/.

[30] *Humboldt Project,* http://www.esdi-humboldt.eu/home.html.

[31] *REWERSE Working Group I1, http://rewerse.net/I1/oxygen.informatik.tu-cottbus.de/rewerse-i1/default.htm*

[32] *R2ML v. 0.5 Guide,* http://rewerse.net/I1/oxygen.informatik.tu-cottbus.de/rewerse-i1/@q=node_2f6.htm

[33] *Common Logic ISO/IEC Standard,* http://www.standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip.

[34] *Common Logic,* http://common-logic.org

[35] *Tefkat on Wikipedia,* http://en.wikipedia.org/wiki/Tefkat

[36] *Tefkat Project Page at Sourceforge,* http://tefkat.sourceforge.net

[37] Investigation of Model Driven Transformation Tools, http://osera.modeldriven.org/documents/AssessQVT.doc

[38] *OSERA QVT Transform For Eclipse,* http://osera.modeldriven.org/documents/AssessQVT.doc

[39] *Rule Interchange Format Core Dialect W3C Candidate Recommendation 1 October 2009,* http://www.w3.org/TR/2009/CR-rif-core-20091001/

[40] *Rule Interchange Format Production Rules Dialect W3C Candidate Recommendation 1 October 2009,* http://www.w3.org/TR/2009/CR-rif-prd-20091001/

[41] *Rule Interchange Format Basic Logic Dialect W3C Candidate Recommendation 1 October 2009,* http://www.w3.org/TR/2009/CR-rif-bld-20091001/

[42] *XPath Specification v. 1.0,* http://www.w3.org/TR/xpath/

[43] *OMG Model Interoperability Demonstration,* http://www.omg.org/news/releases/pr2010/01-04-10.htm

[44] *Deegree Project Website,* http://www.deegree.org/.

[45] *Deegree Web Processing Service Documentation v. 2.3,* http://download.deegree.org/deegree2.3/docs/wps/html/deegree_wps_documentation_en.html

[46] *XtraServer Documentation,* www.interactive-instruments.de/index.php?id=xtraserver&L=1

[47] *Web Feature Server Specification,* www.opengeospatial.org/standards/wfs.

[48] *SAFE Software FME Server Overview,* www.safe.com/products/server/overview.php.

[49] *SAFE Software INSPIRE Overview,* www.safe.com/c/inspire/inspire.php.

[50] *FME Server Transformers Brochure,* http://downloads.safe.com/fme/brochures/transformers.pdf.

[51] *SnowFlake Software INSPIRE Overview,* http://www.snowflakesoftware.co.uk/markets/inspire/solution.htm.

[52] *Snowflake Software Cadastral Parcels INSPIRE Demo,* http://www.snowflakesoftware.co.uk/tv/index.htm.

[53] *Talend Integration Suite Overview,* www.talend.com/products-data-integration/talend-integration-suite.php

[54] *Java Topology Suite,* www.vividsolutions.com/jts/jtshome.htm

[55] *SDI Communities: Data quality and knowledge sharing,* www.gsdi.org/gsdiconf/gsdi11/papers/pdf/283.pdf.

[56] *1Spatial Radius Studio Overview*, http://www.1spatial.com/products/#1265028216640_1/5.

[57] *Interactive Instruments' ShapeChange,* http://www.interactive-instruments.de/index.php?id=28&L=1.

[58] *FullMoon XML Processing Framework*, https://projects.arcs.org.au/trac/fullmoon/wiki/FullMoon.

[59] *XML Linking Language (XLink) Version 1.0, http://www.w3.org/TR/xlink/.*

[60] *Humboldt Project, A5.2-D3 [3.3] Conceptual Schema Specification and Mapping,* http://www.gdmc.nl/publications/reports/Humboldt_0987.pdf.

[61] *Sparx Systems' Enterprise Architect 7*.5, http://www.sparxsystems.com/products/ea/index.html.

[62] *Unified Modeling Language (UML) Version 2.2 introductory page,* http://www.omg.org/technology/documents/formal/uml.htm.

[63] *A Detailed Comparison of UML and OWL,* http://madoc.bib.uni-mannheim.de/madoc/volltexte/2008/1898/pdf/TR2008_004.pdf.

[64] *Bulgarian Language version of INSPIRE Directive,* http://eur-lex.europa.eu/JOHtml.do?uri=OJ%3AL%3A2007%3A108%3ASOM%3ABG%3AHTML.

[65] *Unicode in XML and other Markup Languages,* http://www.w3.org/TR/unicode-xml/.

[66] *INSPIRE Generic Conceptual Model,* http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.5_v3.2.pdf.

[67] *GNU Lesser General Public Licence*, http://www.gnu.org/licenses/lgpl.html.

[68] *Humboldt Project,* http://www.esdi-humboldt.eu/home.html.

[69] *Humboldt application scenarios,* http://www.esdi-humboldt.eu/scenarios.html.

[70] *AuScope*, http://www.auscope.org.au/.

[71] *Australia's Amazing AuScope*, http://www.geoplace.com/ME2/dirmod.asp?sid=&nm=&type=MultiPublishing&mod=PublishingTitles&mid=13B2F0D0AFA04476A2ACC02ED28A405F&tier=4&id=AD3F6EAC1BBC4BB39493B9E4356F81B8.

[72] *Xalan-Java Extensions,* http://xml.apache.org/xalan-j/extensions.html.

[73] *SAXON Extensions,* http://saxon.sourceforge.net/saxon6.5.2/extensions.html.

[74] *xsltproc (part of the XSLT C Library for Gnome),* http://xmlsoft.org/XSLT/extensions.html.