SOAP HTTP Binding Status

Survey on OGC and ORCHESTRA specifications relevant for the INSPIRE Network Services

Matteo Villa
TXT e-solutions

Roberto Lucchi, Michel Millot, Ioannis Kanellopoulos (Editors)
European Commission
Joint Research Centre
Institute for Environment and Sustainability
The mission of the Institute for Environment and Sustainability is to provide scientific-technical support to the European Union’s Policies for the protection and sustainable development of the European and global environment.

European Commission
Joint Research Centre
Institute for Environment and Sustainability

Contact information

Matteo Villa
TXT e-Solutions S.p.A.
Via Frigia, 27
20126 Milano (MI)
ITALY
E-mail: matteo.villa@txt.it
Tel.: +39 02 25771
Fax: +39 02 25771828

Roberto Lucchi (Editor)
European Commission Joint Research Centre
Institute for Environment and Sustainability
Spatial Data Infrastructures Unit
TP262, via Fermi 2749
I-21027 Ispra (VA)
ITALY
E-mail: Roberto.Lucchi@jrc.it
Tel.: +39 0332 78 5325
Fax: +39 0332 78 6325

Michel Millot (Editor)
European Commission Joint Research Centre
Institute for Environment and Sustainability
Spatial Data Infrastructures Unit
TP262, via Fermi 2749
I-21027 Ispra (VA)
ITALY
E-mail: Michel.Millot@jrc.it
Tel.: +39 0332 78 6146
Fax: +39 0332 78 6325

Ioannis Kanellopoulos (Editor)
European Commission Joint Research Centre
Institute for Environment and Sustainability
Spatial Data Infrastructures Unit
TP262, via Fermi 2749
I-21027 Ispra (VA)
ITALY
E-mail: Ioannis.Kanellopoulos@jrc.it
Tel.: +39 0332 78 5115
Fax: +39 0332 78 6325
### Table of Contents

1. Introduction ................................................................. 6
2. Purpose and Scope of the Document ........................................ 8
3. Normative References .................................................... 9
4. State of Play .............................................................. 10
   4.1 Inspire Discovery Service .................................................. 10
      4.1.1 OGC Catalogue Service (OGC CSW) ................................ 11
      4.1.2 Orchestra Catalogue Service ........................................ 21
      4.1.3 Inspire proposed Implementation .................................. 22
   4.2 Inspire View Service ...................................................... 24
      4.2.1 OGC WMS - WebMapping Service ................................. 24
      4.2.2 Orchestra Map&Diagram Service ................................... 29
      4.2.3 Inspire proposed Implementation .................................. 30
   4.3 Inspire Transformation Services ....................................... 32
      4.3.1 OGC WCTS - Web Coordinates Transforming Services .......... 32
      4.3.2 Orchestra Schema Mapping Service ................................. 32
      4.3.3 Other Relevant OGC Services ...................................... 35
   4.4 Inspire Invoke Services .................................................. 36
      4.4.1 OGC WPS - Web Processing Service ............................... 36
      4.4.2 Orchestra Processing Service ....................................... 39
   4.5 Inspire Download Service ................................................ 40
      4.5.1 OGC WCS - Web Coverage Service ................................ 40
      4.5.2 OGC WFS - Web Feature Service .................................. 44
      4.5.3 Orchestra Feature Access Service .................................. 48
5. Critical Review ............................................................ 50
   5.1 Inspire Discovery Service .................................................. 50
      5.1.1 Analysis of OGC-CWS implementations ............................. 50
      5.1.2 ORCHESTRA Catalogue Service .................................... 53
      5.1.3 INSPIRE Proposed Implementation .................................. 53
   5.2 Inspire View Service ...................................................... 54
   5.3 Inspire Transformation Services ....................................... 57
   5.4 Inspire Invoke Service ..................................................... 59
   5.5 Inspire Download Service ................................................ 60
   5.6 Common Issues .......................................................... 61
      5.6.1 SOAP Header ........................................................... 62
      5.6.2 Exception report SOAP encoding ..................................... 62
      5.6.3 Protocol Binding ........................................................ 62
      5.6.4 Data Encoding .......................................................... 63
      5.6.5 Binary Data .............................................................. 66
6. Conclusions ................................................................. 68
7. Annex A – Terms, Definitions and Abbreviations ......................... 69
8. Annex B – Inspire IR Reference ............................................ 70
9. Annex C – Bibliography ..................................................... 71
Table of Figures

Table 1 - Normative References ................................................................. 9
Table 2 - Definition of “getCapabilities” operation..................................... 11
Table 3 - Definition of “query” operation ................................................... 12
Table 4 - Definition of “present” operation ................................................ 12
Table 5 – Definition of “describeRecordType” operation ......................... 12
Table 6 - Definition of “getDomain” operation ......................................... 13
Table 7 – Definition of “transaction” operation .......................................... 13
Table 8 – Definition of “harvestResource” operation ................................. 13
Table 9 – Mapping of general model operations to CSW operations .......... 14
Table 10 – HTTP bindings ...................................................................... 14
Table 11 - Mapping CSW(T) ISO operations to CSW operations ............. 15
Table 12 - Operation request encoding .................................................... 16
Table 13 - CSW-ebRIM HTTP method bindings ....................................... 17
Table 14 - CSW-ebRIM exception codes ................................................... 17
Table 15 - EOPProfile-CSW operations mapping ..................................... 19
Table 16 - ORCHESTRA Catalogue Service operations ......................... 22
Table 17 - CSW ISO AP operations requirements for INSPIRE Discovery Services .................................................. 23
Table 18 – OGC WMS Operations ......................................................... 25
Table 19 - New WMS Operation added by Styled Layer Description Specification .................................................. 25
Table 20 - Summary of Service Operations .............................................. 30
Table 21 - View Service operations .......................................................... 31
Table 22 - Summary of Service Operations .............................................. 34
Table 23 - WPS Operations .................................................................... 37
Table 24 - Operations request encoding .................................................. 38
Table 25 - ORCHESTRA PS - OGC WPS Operations mapping ............. 39
Table 26 - WCS operations ................................................................... 41
Table 27 - WCS Operations Encodings ................................................... 41
Table 28 - WFS Operations ................................................................... 44
Table 29 - WFS Operations Request Encoding ...................................... 45
Table 30 - Request encoding and transport methods ............................. 45
Table 31 - WFS Summary of additional Service Operations ................... 48
Table 32 - FAS Operations Encodings .................................................... 49
Table 33 – Analysis of OGC-CWS profiles .............................................. 52
Table 34 – OGC-WMS vs. ORCHESTRA Map&Diagram operations ........ 53
Table 35 – OGC-WMS vs. ORCHESTRA Map&Diagram operations ........ 54
Table 36 – Analysis of Inspire View Service related specifications .......... 56
Table 37 - OGC and ORCHESTRA WCTS and WPS services summary .... 58
Table 38 - OGC WPS and ORCHESTRA PS service comparison ............. 59
Table 39 - OGC WCS, OGC WFS and ORCHESTRA FAS service comparison .................................................. 61
Table 40 - Data Encoding Summary ....................................................... 65
Table 41 - Binary data management summary ......................................... 66
Table 42 – Document Abbreviations Table ............................................. 69
Table 43 - Inspire IR References .............................................................. 70
Table 44 - Bibliography .......................................................................... 72
1. Introduction

The general situation on spatial information in Europe is of fragmentation of datasets and sources, gaps in availability, lack of harmonization between datasets at different geographical scales and duplication of information collection. These problems make it difficult to identify, access and use data that are available.

Fortunately, awareness is growing at both national and EU level about the need for quality geo-referenced information to support understanding of the complexity and interactions between human activities and environmental pressures and impacts. The INSPIRE directive proposal [2007/2/EC] is therefore timely and relevant, but also a major challenge, given the general situation outlined above and the many stakeholder interests to be addressed.

INSPIRE is complementary to related policy initiatives, such as the Commission proposal for a Directive on the re-use and commercial exploitation of Public Sector Information. [2003/98/EC]

INSPIRE is ambitious: the initiative intends to trigger the creation of a European spatial information infrastructure that delivers to the users integrated spatial information services. These services should allow the users to identify and access spatial or geographical information from a wide range of sources, from the local level to the global level, in an interoperable way for a variety of uses. The target users of INSPIRE include policy-makers, planners and managers at European, national and local level and the citizens and their organizations.

INSPIRE will require the Member States to implement various measures. Some of these measures will be directly transposed by the Member States, while others require more details which will be formulated in so called ‘Implementing Rules’. Hence, in parallel to the Co-Decision process [Co-Dec], the Commission has initiated actions to prepare draft Implementing Rules with the set-up of Drafting Teams mainly composed of European Union experts nominated by Spatial Data Interest Communities and Legally Mandated organizations. [INSPIRE WPPP]

Interoperability will be reached progressively as metadata, data and services compliant with the INSPIRE Implementing Rules are becoming available and will require the active involvement of all actors identified in INSPIRE, namely European Union Member States relevant Institutions and the Commission.

According to W3C Web Services definition (http://www.w3c.org/TR/ws-arch/), a Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

As the INSPIRE directive advises to utilize existing standards, OGC service bindings are taken as a guidance. Existing OGC Web Services (OWS) support a mix of protocols and technology bindings. These are Key-Value-Pairs send via HTTP GET, XML send via HTTP POST, SOAP via HTTP POST and combinations. In addition, the World Wide Web Consortium suggests the usage of SOAP as a messaging protocol for web services. INSPIRE services should utilize one standard technology binding for all service types. In order to streamline integration and implementation as well as getting a maximum benefit from the offered services, a mix of technologies is to be avoided. Taking all requirements, opportunities and risks into account, the default communication-protocol and binding technology for INSPIRE services should be SOAP (using document/literal encoding style).
According to the “SOAP Experiment report” [OGC 03-014], Web services offer a new platform for building loosely coupled, distributed systems. OGC has produced a set of discussion papers to discover the feasibility and usefulness of enabling SOAP communication in OGC services. The premise of the SOAP experiment is the belief that porting OWS services to Web Services will offer several key benefits, including:

- **distribution**: it will be easier to distribute geospatial data and applications across platforms, operating systems, computer languages, etc;
- **integration**: it will be easier for application developers to integrate geospatial functionality and data into their custom applications;
- **infrastructure**: the GIS industry could take advantage of the huge amount of infrastructure that is being built to enable the Web Services architecture – including development tools, application servers, messaging protocols, security infrastructure, workflow definitions, etc.

On the other hand, “INSPIRE – Network Services Architecture” [INSPIRE D3.5], reports that existing OGC Web Services (OWS) typically support HTTP GET and/or HTTP POST as messaging protocol and they publish the `GetCapabilities` operation to provide service Metadata. The World Wide Consortium suggests the usage of SOAP (currently version 1.2) as a messaging framework and WSDL (currently version 2.0) to describe service metadata.

A number of arguments can be listed in favour of SOAP/WSDL approach for the INSPIRE network services:

a) SOAP Web services are becoming the **standard** information technology and thus support more sustainable implementing rules;
b) SOAP Web service ensure smooth and complete **integration** in development environments;
c) SOAP Web services yield a direct and full **integration** with other web service environments;
d) SOAP Web services have the possibility to support geo rights **management services** by using SOAP envelope data.

However, since the OGC interface specifications support HTTP GET or HTTP POST, there are only a few attempts to use SOAP/WSDL for geo-information services. Recent experiments (e.g. within OGC, but also within EU funded Projects as ORCHESTRA) provided interface specification for SOAP/WSDL based geo-information services but also indicated some issues in realising operational services based on SOAP/WSDL, e.g. a lack of support by current development tools.
2. Purpose and Scope of the Document

The goal of this document is to describe the status of SOAP HTTP binding of those services relevant to Inspire Network Services, thus including OGC specifications (CSW, WMS, WFS, WPS, WCTS and WCS) and implementation (OWS3-5), as well as IST ORCHESTRA relevant services.

This document is structured in the following way:

- **Chapter 1** is the introduction.
- **Chapter 2**, this one, explains the purpose of the document.
- **Chapter 3** provides normative references.
- **Chapter 4** describes the state of play for the following Inspire services: Discovery Services (OGC CWS, Orchestra Catalogue service), View Services (OGC WMS, Orchestra Map&Diagram service), Transformation Services (OGC WCTS, Orchestra Schema Mapping service), Invoke Services (OGC WPS, Orchestra Processing Service) and Download Service (OGC WCS, OGC WFS, Orchestra Feature Access Service).
- **Chapter 5** provides a critical review analysis, taking into account the different specifications and/or implementative choices for each service, and identifying a set of common issues (like data encoding, binary data, SOAP header, exception reporting, ...)
- **Chapter 6** provides some conclusions about this study.
3. Normative References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

<table>
<thead>
<tr>
<th>REF</th>
<th>AUTHORS</th>
<th>TITLE</th>
<th>YEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>OGC 03-014</td>
<td>J Sonnet C. Savage</td>
<td>OWS 1.2 SOAP Experiment report</td>
<td>2003</td>
</tr>
<tr>
<td>OGC 06-121r3</td>
<td>A. Whiteside</td>
<td>OGC Web Services Common Specification</td>
<td>2007</td>
</tr>
<tr>
<td>OGC 04-060r1</td>
<td>J. Sonnet</td>
<td>Common Architecture: WSDL SOAP UDDI</td>
<td>2004</td>
</tr>
<tr>
<td>OGC 08-009r1</td>
<td>B. Schäffer</td>
<td>SOAP/WSDL Common Engineering Report</td>
<td>2008</td>
</tr>
<tr>
<td>RFC 2616</td>
<td>R. Fielding et al.</td>
<td>Hypertext Transfer Protocol – HTTP/1.1</td>
<td>1999</td>
</tr>
</tbody>
</table>

These documents could contain list of normative references that are also applicable to this study. 

*NOTE: Additional normative references to be confirmed, for a complete set of references see the Bibliography chapter.*
4. State of Play

4.1 Inspire Discovery Service

The INSPIRE Directive [2007/2/EC] asks Member States to “establish and operate a network of services” for the discovery of spatial data sets and services “for which metadata have been created”. Discovery services should “make it possible to search for spatial data sets and services on the basis of the content of the corresponding metadata and to display the content of the metadata. […] Those services shall take into account relevant user requirements and shall be easy to use, available to the public and accessible via the Internet or any other appropriate means of telecommunication.”

In INSPIRE these services are referred to as Discovery Services.

Summarizing, the goal of discovery service is to support discovery, evaluation and use of spatial data and services through their metadata properties. Metadata is the information and documentation, which makes these resources understandable and sharable for users over time.

The network of services should also include the technical possibility to enable public authorities to make their spatial datasets and services available. The INSPIRE Directive specifies that Member States shall “ensure that public authorities are given the technical possibility to link their spatial datasets and services to the network”. This ‘linking’ service is also offered in the context of a discovery service.

For the purpose of the search services, as a minimum, the following combination of search criteria shall be implemented:

- “keywords;
- classification of spatial data and services;
- spatial data quality and accuracy;
- degree of conformity with the implementing rules provided […];
- geographical location;
- conditions applying to the access to and use of spatial datasets and services;
- the public authorities responsible for the establishment, management, maintenance (and)
- distribution of spatial datasets and services.”

The INSPIRE survey of Novak and Craglia [2006, J.N&M.C] shows the deployment of Catalogue Service standards by the INSPIRE SDICs and LMOs. OGC standards are applied to 39% of all metadata holdings.

The OGC Catalogue Service for the Web specification is the most widely adopted standard for the INSPIRE SDICs and LMOs: it offers the functionality needed to fulfil the INSPIRE use cases for discovery.

Within the context of the OGC-CSW specifications, three relevant proposed implementations of the Catalogue Service for the Web are taken into account:

1. Metadata Application Profile (OWS)
2. ebRIM Application Profile (OWS)
3. a draft of Application Profile for Earth Observation (OGC in collaboration with ESA).
Another relevant approach related to a Discovery Service (specifications and implementation) is represented by the **Orchestra Catalogue Service**.

Finally, it is necessary to take into account the **INSPIRE** draft proposed implementation.

### 4.1.1 OGC Catalogue Service (OGC CSW)

#### 4.1.1.1 Overview
Catalogue Services support the ability to publish, access and search digital catalogues of metadata for geospatial data, services, and related information resources. Metadata in catalogues represent resource characteristics that can be queried and presented for evaluation and further processing, by both humans and software. [OGC 07-006r1]

Catalogue services are required to support the discovery and binding to registered information resources within an information community.

#### 4.1.1.2 Specification
Current “OGC Catalogue Service for the Web” specification version is 2.0.2 and can be found at: [http://portal.opengeospatial.org/files/?artifact_id=20555](http://portal.opengeospatial.org/files/?artifact_id=20555).

OGC defines a *General Catalogue Interface Model* [OGC 07-006r1, section 7] providing a set of abstract service interfaces that support the discovery, access, maintenance and organization of catalogues of geospatial information and related resources. The interfaces specified are intended to allow users or application software to find information that exists in multiple distributed computing environments, including the World Wide Web (WWW) environment.

Amongst all the defined operations, seven are used as basis for further specifications of the CSW:

1) **“getCapabilities”** operation:

<table>
<thead>
<tr>
<th><strong>Definition</strong></th>
<th>It allows clients to retrieve service metadata describing Catalogue Service instance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Receives</strong></td>
<td>Optional identifier(s) of requested parts of the complete service metadata document</td>
</tr>
<tr>
<td><strong>Returns</strong></td>
<td>Service metadata document for Catalogue Service instance. Some document contents depend on the set of classes that are associated with the Catalogue Service class, as defined by the specific protocol binding, and on other details of that protocol binding. Other document contents depend on the types of data defined by the specific application profile, and on other details of that profile.</td>
</tr>
<tr>
<td><strong>Exceptions</strong></td>
<td><em>Invalid Parameter Value, Missing Parameter Value</em></td>
</tr>
<tr>
<td><strong>Pre-conditions</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Post-conditions</strong></td>
<td>Service metadata document returned to requesting client, either complete or including selected parts</td>
</tr>
</tbody>
</table>

*Table 2 - Definition of “getCapabilities” operation*

2) **“query”** operation

| **Definition** | It allows clients to ask a catalogue to execute a query that searches the catalogued metadata and produces a result set containing (zero or more) references to all the registered resources that satisfy the query. The server may maintain the result set for subsequent retrieval requests |

---

Page 11 of 75
3) “present” operation:

| Definition | It allows clients to retrieve selected metadata for some or all of the resources referenced in a specific previous result set or a list of resource identifiers. This operation can be used repetitively to retrieve more of the result set, each time retrieving metadata for a maximum number of the resources listed, starting at a specified position. |
| Receives | Specifications of sorting and of metadata to be returned, optionally including maximum number of records for which metadata is to be returned |
| Returns | Metadata document containing selected metadata for some or all of the specific result set, after it is sorted as specified by the client. Most of the metadata returned depends on the metadata requested and on the types of data defined by the specific Application Profile |
| Exceptions | Missing Parameter Value, Invalid Parameter Value, Unrecognized collection identifier |
| Pre-conditions | Client has previously performed “search” operation, and the server has provided a result set identifier that the client can use to perform the present operation |
| Post-conditions | Metadata document returned to requesting client, containing selected metadata for some or all of sorted result set |

4) “describeRecordType” operation:

| Definition | It allows clients to retrieve type definition(s) used by metadata of one or more registered resource types |
| Receives | Optional identifications of requested record type(s) and of desired format |
| Returns | Type definition document containing definition(s) of type(s) used by the metadata of one or more registered resource types. This type definition shall include the structure (schema), queryables, element sets, and formats of the metadata used for one or more registered resource types. The contents of the result of this operation depend on the types of metadata that can currently be used by registered resources. |
| Exceptions | Missing Parameter Value, Invalid Parameter Value, Nonexistent type |
| Pre-conditions | None |
| Post-conditions | Type definition document returned to requesting client, containing definition(s) of type(s) used by the metadata of one or more registered resource types |

5) “getDomain” operation:
Definition | It allows clients to retrieve the domain (allowed values) of a metadata property or request parameter at the time the request is invoked. The returned information may be static domain information, but may also be dynamic in that the allowed values are determined at runtime. The operation does a best attempt at returning information about a metadata property or request parameter.  
Receives | Names of one or more requested metadata properties or request parameters.  
Returns | Descriptions of domains of one or more requested metadata properties or request parameters  
Exceptions | Missing Parameter Value, Invalid Parameter Name  
Pre-conditions | None  
Post-conditions | Descriptions of domains returned to requesting client, containing the domain descriptions for all the identified metadata properties or request parameters

Table 6 – Definition of “getDomain” operation

6) ”transaction” operation

Definition | It allows clients to request a specified set of “insert”, “update”, and “delete” actions on the content managed by a Catalogue Service instance.  
Receives | Specification of set of “insert”, “update”, and “delete” actions, plus an optional identifier. At least one action shall be included  
Returns | A summary of the transaction results that identifies newly created entries when applicable. Most contents of the result depend on the types of data defined by the specific protocol binding and Application Profile.  
Exceptions | Missing Parameter Value, Invalid Parameter Value, Transaction Failed  
Pre-conditions | User is authorized to modify catalogue contents  
Post-conditions | Catalogue entries are inserted, updated, and/or deleted as requested, and the integrity and consistency of catalogue contents are preserved

Table 7 – Definition of “transaction” operation

7) “harvestResource” operation

Definition | It allows a user to request that a catalogue service attempt to retrieve a resource from a specified location, and to optionally create one or more entries for that resource. A harvest attempt may occur periodically if an interval is specified  
Receives | A request message containing the source of the resource to be harvested  
Returns | An acknowledgement that a harvestRequest has been received and validated (if a responseHandler is specified) or a summary of the harvest results that identifies newly harvested records (if a responseHandler is not specified). Most contents of the result depend on the types of data defined by the specific protocol binding and Application Profile.  
Exceptions | InvalidRequest, ResourceNotFound  
Pre-conditions | The user is permitted to modify catalogue contents, unless the scope of the harvest does not include an insert or update transaction  
Post-conditions | One or more records are harvested from a remote system and optionally new catalogue entries are created or existing entries are updated, and the integrity and consistency of the catalogue contents are preserved

Table 8 – Definition of “harvestResource” operation

Such operations are then mapped to the Catalogue Service for the Web (CSW) operations [OGC 07-006r1, section 10.4], as shown in Table 9 (the table does not list the general model operations that are not mapped to CSW operations)
CSW servers shall indicate whether or not SOAP encodings of operation requests can be transferred, using HTTP POST. This indication shall use an `<ows:Constraint>` element named **PostEncoding**, within the `<ows:OperationsMetadata>` section of the Capabilities document. This **PostEncoding** constraint shall specify the formats that can be used with HTTP POST transfer of operation requests. The value “SOAP” for **PostEncoding** shall indicate that SOAP encoded operation requests can be handled. The value “XML” shall indicate that (bare) XML encoded operation requests can be handled. If the connect point URL is the same for all SOAP-encoded and base-XML operation requests, the `<ows:Constraint>` element shall be included in the `<ows:OperationsMetadata>` element. If the connect point URL is different for SOAP-encoded and base-XML operation requests, this `<ows:Constraint>` element shall be included in each `<ows:Post>` element.

We can now analyse how to bind these methods to HTTP and to SOAP:

1. **HTTP:** Table 10 summarizes allowed HTTP method bindings and request data encodings for all CSW requests; optional method bindings and data encodings are enclosed in parentheses.

<table>
<thead>
<tr>
<th>Request</th>
<th>HTTP method binding(s)</th>
<th>Data encoding(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET (POST)</td>
<td>KVP (XML)</td>
</tr>
<tr>
<td>DescribeRecord</td>
<td>POST (GET)</td>
<td>XML (KVP)</td>
</tr>
<tr>
<td>GetDomain</td>
<td>POST (GET)</td>
<td>XML (KVP)</td>
</tr>
<tr>
<td>GetRecords</td>
<td>POST (GET)</td>
<td>XML (KVP)</td>
</tr>
<tr>
<td>GetRecordByld</td>
<td>GET (POST)</td>
<td>KVP (XML)</td>
</tr>
<tr>
<td>Harvest</td>
<td>POST (GET)</td>
<td>XML (KVP)</td>
</tr>
<tr>
<td>Transaction</td>
<td>POST</td>
<td>XML</td>
</tr>
</tbody>
</table>

*a* XML = application/xml using POST (with a charset parameter if necessary. UTF-8 is strongly recommended)

*b* KVP = URL-encoded key/value pairs using GET or application/x-www-form-urlencoded using POST

2. **SOAP:** A client may send CSW requests to a compatible catalogue using the body of a SOAP envelope. The client simply encodes the CSW request as the content of the `<soap:Body>` element in the request message. The CSW shall then respond by generating a SOAP message where the response to the client’s request is the content of the `<soap:Body>` element. If an exception is encountered while processing a CSW request encoded in a SOAP envelope, the CSW server shall generate a SOAP response message where the content of the `<soap:Body>` element...
is a `<soap:Fault>` element. The following skeleton XML fragment shall be used when generating the `<soap:Body>` element in the event that the CSW server encounters an exception:

```xml
<soap:Body>
  <soap:Fault>
    <soap:Code>
      <soap:Value>soap:Server</soap:Value>
    </soap:Code>
    <soap:Reason>
      <soap:Text>A server exception was encountered.</soap:Text>
    </soap:Reason>
    <soap:Detail>
      <ows:ExceptionReport>…</ows:ExceptionReport>
    </soap:Detail>
  </soap:Fault>
</soap:Body>
```

The `<soap:Value>` element in the `<soap:Code>` element shall have the content “`soap:Server`” indicating that this is a server exception. The `<soap:Text>` element in the `<soap:Reason>` element shall have the content “Server exception was encountered.”. This fixed string is used since the details of the exception will be specified in the `<soap:Detail>` element using an `<ows:ExceptionReport>` element as defined in document [OGC 06-121r3]. The `<soap:Detail>` element shall contain an `<ows:ExceptionReport>` element detailing the specific exception that the server encountered.

### 4.1.1.3 OWS ISO Metadata Application Profile

Referring to [OGC 07-045], we analyse the ISO Metadata Application Profile. Table 11 shows how the operations of this profile (CSW(T) ISO) are mapped to the operations specified by the CSW 2.0.2 specification. This is a full mapping in that all of the CSW(T) ISO operations have a corresponding CSW operation.

<table>
<thead>
<tr>
<th>CSW OPERATION</th>
<th>CSW(T) ISO OPERATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>OGC_Service.GetCapabilities</td>
<td>OGC_Service.GetCapabilities</td>
</tr>
<tr>
<td>CSW Discovery.DescribeRecord</td>
<td>CSW Discovery.DescribeRecord</td>
</tr>
<tr>
<td>CSW Discovery.GetDomain</td>
<td>CSW Discovery.GetDomain</td>
</tr>
<tr>
<td>CSW Discovery.GetRecords</td>
<td>CSW Discovery.GetRecords</td>
</tr>
<tr>
<td>CSW Discovery.GetRecordById</td>
<td>CSW Discovery.GetRecordById</td>
</tr>
<tr>
<td>CSW Publication.Harvest</td>
<td>CSWT Manager.Harvest</td>
</tr>
<tr>
<td>CSW Publication.Transaction</td>
<td>CSWT Manager.Transaction</td>
</tr>
</tbody>
</table>

*Table 11 - Mapping CSW(T) ISO operations to CSW operations*

Apart from the `GetCapabilities` operation, all the other ones must support the embedding of requests and responses in SOAP messages. In this case only SOAP messaging (via HTTP/POST) with `document/literal` style has to be used. Messages must conform to SOAP 1.2 (http://www.w3.org/TR/SOAP/). The message payload will be in the body of the SOAP envelope. Most operations support the encoding of the request messages as **Keyword-Value Pairs (KVP)** within
a request URI; all operations support the usage of a XML entity-body and responses are XML-encoded.

The HTTP encoding of catalogue operation requests shall use HTTP GET with Keyword-Value Pairs (KVP) encoding and HTTP POST with XML encoding as specified in Section 11 of [OGC 06-121r3]. Note that the parameter names in all KVP encodings must be handled in a case insensitive manner while parameter values shall be handled in a case sensitive manner.

Table 12 summarizes the CSW(T) ISO operations and their encoding methods that are applied in this profile. The mandatory method bindings and data encodings are printed in bold.

<table>
<thead>
<tr>
<th>CSW(T) ISO OPERATION</th>
<th>REQUEST ENCODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>XML(POST+SOAP) and KVP (GET)</td>
</tr>
<tr>
<td>DescribeRecord</td>
<td>XML (POST+SOAP) and KVP (GET)</td>
</tr>
<tr>
<td>GetDomain</td>
<td>XML (POST+SOAP) and KVP(GET)</td>
</tr>
<tr>
<td>GetRecords</td>
<td>XML (POST+SOAP) and KVP(GET)</td>
</tr>
<tr>
<td>GetRecordById</td>
<td>XML (POST+SOAP) and KVP(GET)</td>
</tr>
<tr>
<td>Harvest</td>
<td>XML (POST+SOAP) and KVP(POST)</td>
</tr>
<tr>
<td>Transaction</td>
<td>XML (POST+SOAP)</td>
</tr>
</tbody>
</table>

Table 12 - Operation request encoding

There is no KVP encoding for transaction operation request, because there is no convenient way of encoding the transaction payloads using keyword-value pairs.

**Technical issues:**

- **HTTP:** The base communication protocol is HTTP 1.1 as specified by IETF RFC 2616 ([http://www.ietf.org/rfc/rfc2616](http://www.ietf.org/rfc/rfc2616)). All bindings of operations (see Table 5) MUST be consistent with HTTP/1.1 semantics. Alternative bindings may be specified for a specific service instance. Any HTTP/1.1 response message containing an entity-body must include a Content-Type header field defining the media type of that body ([RFC 2616], 7.2.1). This includes the charset parameter ("application/xml; charset=utf-8").

- **SOAP:** Only SOAP messaging (via HTTP/POST) with document/literal style has to be used. Messages must be compliant with SOAP 1.2 ([http://www.w3.org/TR/SOAP/](http://www.w3.org/TR/SOAP/)). The message payload will be in the body of the SOAP envelope.

**Security considerations:**

- Security issues are part of the implementation specification of a catalogue service. It is recommended that HTTP Basic Authentication is used to prevent access to the URLs of the transaction interface, unless the requestor can provide user/password credentials. This basic authentication should be used in conjunction with HTTPS as part of a security solution.

**4.1.1.4 OWS ebRIM Application profile of CSW**

OWS followed also another way to implement CSW, trying to join the CSW interfaces with the OASIS ebXML registry information model (ebRIM 3.0), to provide a general and flexible Web-based registry service that enables users, human or software agents to locate, access and make use of resources in an open distributed system. [OGC 07-110r2]

**Use of HTTP methods:** the HTTP/1.1 specification [RFC 2616] defines eight methods for manipulating and retrieving representations of resources. Only the GET and POST methods are
supported in this application profile. Service requests are bound to HTTP methods as indicated in Table 13:

<table>
<thead>
<tr>
<th>CSW-ebRIM OPERATION</th>
<th>HTTP METHOD BINDING(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET and POST</td>
</tr>
<tr>
<td>GetRecords</td>
<td>POST and GET</td>
</tr>
<tr>
<td>GetRecordById</td>
<td>GET and POST</td>
</tr>
<tr>
<td>DescribeRecord</td>
<td>GET and POST</td>
</tr>
<tr>
<td>GetDomain</td>
<td>GET and POST</td>
</tr>
<tr>
<td>GetRepositoryItem</td>
<td>GET</td>
</tr>
<tr>
<td>Transaction</td>
<td>POST</td>
</tr>
<tr>
<td>Harvest</td>
<td>POST</td>
</tr>
</tbody>
</table>

*Table 13 - CSW-ebRIM HTTP method bindings*

Proper use of the *Content-Type* header is required by this profile; all request and response messages shall correctly indicate the content type of the message (if not empty). The following message content types shall be recognized by all conforming implementations:

a) *application/xml* – the body contains an XML request or response entity;
b) *application/x-www-form-urlencoded* – the body contains a KVP-style encoding of request parameters;
c) *application/soap+xml* – the body contains a SOAP 1.2 envelope;
d) *multipart/related* – the message is a compound object containing related parts structured in accord with RFC 2387 [RFC 2378].

**Exception codes**: The OWS Common specification defines a number of general exception codes that shall be supported by all implementations (OGC 05-008, cl. 8.3). This profile defines additional exception codes and also requires the proper use of HTTP status codes, as shown in the following table:

<table>
<thead>
<tr>
<th>CODE</th>
<th>REASON</th>
<th>STATUS CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wrs:InvalidRequest</td>
<td>The request message is invalid for some reason.</td>
<td>400</td>
</tr>
<tr>
<td>Wrs:NotImplemented</td>
<td>The request is not supported by the service.</td>
<td>500</td>
</tr>
<tr>
<td>Wrs:NotFound</td>
<td>The requested resource does not exist or could not be found.</td>
<td>404</td>
</tr>
<tr>
<td>Wrs:NotSupported</td>
<td>A service option or media type is unsupported.</td>
<td>415</td>
</tr>
<tr>
<td>Wrs:TransactionFailed</td>
<td>The requested transaction could not be completed.</td>
<td>500</td>
</tr>
</tbody>
</table>

*Table 14 - CSW-ebRIM exception codes*

**Use of the SOAP messaging framework - HTTP binding**: this implementation may support the W3C SOAP 1.2 messaging framework. If so, this shall be advertised in the capabilities document by including the value *application/soap+xml* for the operation-specific *Content-Type* constraint. A service that supports SOAP shall conform to the SOAP 1.2 HTTP binding (SOAP Part 2); such a binding shall also be declared in a WSDL service description if one is available.

The HTTP binding employs two message exchange patterns: “Request-Response” and “SOAP Response”. In general, a request bound to the GET method is restricted to the “SOAP response”...
pattern; POST requests are restricted to the “Request-Response” pattern. A conforming implementation shall support both patterns. If a request message does not contain a SOAP envelope, as in the GET method, but a SOAP response is desired, the Accept request header shall be used to indicate this preference. A normal response is produced if the service cannot function as a SOAP node.

EXAMPLE Accept: application/soap+xml; application/xml;
Request and response that contain a SOAP message construct shall satisfy all requirements in section 5 in the W3C SOAP 1.2 specification. The SOAP body element shall contain the appropriate XML request or response entity. This specification imposes no constraints on the content of the optional SOAP header element; it may be ignored if present. The action parameter in the Content-Type request header field value, if present, may be used to optimize message dispatching and routing [RFC 3902]. The value should include the following service type identifier: urn:ogc:serviceType:WebRegistryService:1.0.

Fault handling: In the event that an exception report is produced for any reason, a single SOAP Fault element information item shall be included as the only child element information item of the SOAP body element (see SOAP1.2, sec. 5.4). The elements of the SOAP Fault are constructed as follows:

a) env:Code/env:Value = “env:Sender” or “env:Receiver”, depending on the source of the error and the HTTP status code (i.e., 4xx, 5xx);

b) env:Code/env:Subcode/env:Value = value of ows:Exception/@exceptionCode (an OWS exception code value shall be set above with “ows:”);

c) env:Reason/env:Text = ows:Exception/ows:ExceptionText, where @xml:lang = ows:ExceptionReport/@language;

d) env:Detail contains the original ows:ExceptionReport element

Security considerations - authentication:
Authentication is the process of verifying the identity of a user. It is strongly recommended that implementations employ an authentication mechanism to verify the identity of a user that seeks to modify registry content. The standard HTTP authentication schemes documented in RFC 2617 [RFC 2617] are widely employed and are suitable for most applications:

a) basic with TLS (or SSL v3) to establish a secure communication channel;

b) digest.

In the event that a request requires valid authentication credentials and these are either missing or invalid, the response status code shall be 401 (Unauthorized).

4.1.1.5 OGC Application Profile for EO Products
Amongst the possible implementations of the Catalogue Service for the Web, OGC proposed a solution to find and order data products from catalogues of Earth Observation (EO) products. The work, consultable at [OGC 05-057r3], describes a first set of SOAP/HTTP protocol binding that have been inspired from on the catalogue abstract model.

Message-based SOAP (Simple Object Access Protocol) over HTTP or HTTPS for secure communication is used as protocol between the client application and the catalogue service. SOAP is firewall-friendly and platform independent; it is thus well suited to integrate services in a heterogeneous environment.
This profile identifies four basic operations:

a) the “Search” operation is used to perform a query on a remote catalogue. It returns the metadata available in the catalogue for the EO products matching the search parameters such as area of interest and time interval. The client may present the search results in textual (list) format, and graphically on a map;

b) the “Present” operation allows requesting detailed information about a search result (information on the metadata and possibly a graphical overview of the EO product);

c) the “Order” operation allows a product to be requested. In many instances the EO product may not be available online: at one extreme it may have to be tasked by the satellite, or possibly it may have to be retrieved from an archive. The order may require some processing to be performed before its delivery to the user. The order allows the selection of reduced area (scene selection) too;

d) the “OrderMonitor” operation allows the status of the order to be tracked.

The profile supports synchronous operations, which mean that the service will immediately return the result.

The mapping between OGC CSW operations and the EO AP ones is summarised below:

<table>
<thead>
<tr>
<th><strong>EOProfile</strong></th>
<th><strong>CSW</strong></th>
<th><strong>COMMENT</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>EOProfile:SearchRequest</td>
<td>CSW:GetRecords</td>
<td>The operation shall be renamed to be in line with the CSW implementation</td>
</tr>
<tr>
<td>EOProfile:PresentRequest</td>
<td>CSW:GetRecordById</td>
<td>The operation shall be renamed to be in line with the CSW implementation</td>
</tr>
<tr>
<td>OGC_Service:GetCapabilities</td>
<td></td>
<td>Service metadata discovery should be aligned with INSPIRE in a future iteration of the EO profile</td>
</tr>
<tr>
<td>-</td>
<td>CSW:DescribeRecord</td>
<td>Since currently we intend only to specify SOAP bindings and described schema are published and referenced just via WSDL, it seems little added value to provide this interface within the EO profile</td>
</tr>
</tbody>
</table>

*Table 15 - EOProfile-CSW operations mapping*

As shown in the previous table, the “Search” and “Present” functionalities are the only ones directly pertinent with CSW definitions.

The required binding to be mandated by this profile is SOAP; it is not anticipated to require a KVP encoding.

The hand shaking between the SOAP/HTTP server and a SOAP/HTTP client will be the same for both Search and Present request messages. The SOAP/HTTP client sends a SOAP request message. When the response is ready, the SOAP/HTTP server builds the response message and sends the SOAP response back to the client. The message-based variant of SOAP will be used. This means that *style="document"* are used in the WSDL file as attribute of the <binding> element.
4.1.1.6 Change proposals/requests

A change request about SOAP and HTTP binding in CSW has been submitted providing the following guidelines [OGC 06-089]:

a) A client may send CSW requests to a compatible catalogue using the body of a SOAP envelope. The client simply encodes the CSW request as the content of the <soap:Body> element in the request message. The CSW may then respond by generating a SOAP message where the response to the client’s request is the content of the <soap:Body> element.

b) In case an exception is encountered while processing a CSW operation request, encoded in a SOAP envelope, the CSW server must generate a SOAP response message where the content of the <soap:Body> element is a <soap:Fault> element. The following skeleton XML fragment must be used when generating the <soap:Body> element in the event that the CSW server encounters an exception:

```
<soap:Body>
  <soap:Fault>
    <soap:Code>
      <soap:Value>soap:Server</soap:Value>
    </soap:Code>
    <soap:Reason>
      <soap:Text>A server exception was encountered.</soap:Text>
    </soap:Reason>
    <soap:Detail>
      <ows:ExceptionReport>
        ...
      </ows:ExceptionReport>
    </soap:Detail>
  </soap:Fault>
</soap:Body>
```

The <soap:Value> element in the <soap:Code> element shall have the content soap:Server indicating that this is a server exception. The <soap:Text> element in the <soap:Reason> element shall have the content “Server exception was encountered.”. This fixed string is used since the details of the exception are specified in the <soap:Detail> element using an <ows:ExceptionReport>.

The <soap:Detail> element shall contain an <ows:ExceptionReport> element detailing the specific exception that the server encountered.

c) CSW servers shall indicate whether or not SOAP encodings of operation requests can be transferred using HTTP POST. This indication shall use an <ows:Constraint> element named PostEncoding, within the <ows:OperationsMetadata> section of the Capabilities document.

d) This PostEncoding constraint shall specify the formats that can be used with HTTP POST transfer or operation requests. The value “SOAP” for PostEncoding shall indicate that SOAP encoded operation requests can be handled. The value “XML” shall indicate that (bare) XML encoded operation requests can be handled. If the connect point URL is the same for all SOAP-encoded and base-XML operation requests, the <ows:Constraint> element shall be included in the <ows:OperationsMetadata> element. If the connect point URL is different for
SOAP-encoded and base-XML operation requests, this `<ows:Constraint>` element shall be included in each `<ows:Post>` element.

e) This profile imports the HTTP protocol binding from [OGC 07-006r1].

f) Compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only `document-literal` or `rpc-literal` bindings can be employed. The former style entails little more than wrapping a WFS message with a SOAP envelope. In the latter style, the child element of the `<soap:Body>` element consists of a wrapper element bearing either the name of the operation or the name of the operation suffixed with "Response"; the corresponding message parts appear in the guise of child accessor elements. In most cases a SOAP binding does little more than provide a simple envelope that wraps existing messages (for the `document-literal` encoding style). For the `rpc-literal` encoding style, the schema alone is not sufficient to validate the message body: the RPC rules must be known. Given the state of current SOAP tools, interoperability issues may arise.

g) **Message encodings:** many CSW messages admit two encoding styles: an XML representation (`application/xml`) based on an XML Schema type definition, or a key-value pair (KVP) representation that can be readily incorporated within a Request-URI (`application/x-www-form-urlencoded`). Unfortunately these usually have inconsistent content models so they must be defined separately (e.g. `GetCapabilities` KVP representation uses different names and adds the REQUEST parameter). Retrieval operations that are commonly bound to the GET method may also require alternative message definitions.

### 4.1.2 Orchestra Catalogue Service

#### 4.1.2.1 Overview

The Catalogue Service supports the ability to publish, query and retrieve descriptive information (meta information) for resources (i.e. data and services), meta information about Orchestra Source Systems just like meta information for other Orchestra services and instances of feature types that are referred to by extensions of the OMM_FeatureType, such as documents, schemas, dictionaries, equations and models.

#### 4.1.2.2 Specification

Current “Catalogue Service” specification version by ORCHESTRA is 1.1/2.2.1 and can be found at the following URL:


The abstract specification outlines the following operations:

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>This operation returns the service capabilities</td>
</tr>
<tr>
<td>Search</td>
<td>Given a request in a given language, the catalogue returns a list of identifiers for corresponding features</td>
</tr>
<tr>
<td>getMetaInformation</td>
<td>Given some identifiers of features managed by the catalogue, as returned by a previous query, this method returns the associated meta information instances</td>
</tr>
</tbody>
</table>
### 4.1.2.3 Implementation

ORCHESTRA defined an implementation specification of the Catalogue Service for the Web Services Platform, providing a formal description of the implemented interfaces according to the rules defined in the specification of the “ORCHESTRA Web Services Platform”. The proposed implementation can be found at [ORCHESTRA IS-CAT].


### 4.1.3 Inspire proposed Implementation

The *Network Services Drafting Team* has made a proposal for the technical content of the INSPIRE Implementing Rules for Discovery Service; such proposal is available and described in “Draft Implementing Rules for Discovery and View Services (IR1)” [INSPIRE D3.7].

The CSW ISO AP conforms to the ISO 19115 and ISO 19119 standards for metadata of spatial data (sets) and services. ISO 19115 has a broad adoption in geo-community in Europe and CSW ISO AP offers all the functionalities needed for the Discovery Services use cases: describe, discover, publish, manage and harvest.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getQueryDomain</td>
<td>This operation is used by catalogue clients in order to get the domain of values that are applicable to a query parameter. getQuery-Domain delivers allowed values of a property of a meta information type</td>
</tr>
<tr>
<td>getMetaInformationType</td>
<td>Given a list of catalogue entry types managed by the catalogue, this operation returns the associated meta information type. The meta information type can include meta information defined in an OAS_MI or could include other meta information used in well known standards</td>
</tr>
<tr>
<td>createMetaInformation</td>
<td>This operation creates meta information in catalogue entries managed by the catalogue</td>
</tr>
<tr>
<td>setMetaInformation</td>
<td>This operation updates meta information in existing catalogue entries</td>
</tr>
<tr>
<td>deleteMetaInformation</td>
<td>This operation deletes meta information from the catalogue</td>
</tr>
<tr>
<td>collectMetaInformation</td>
<td>This operation given one reference to a source of meta information, collects meta information for the discovery purpose using the means from other services. This operation can be called directly in a synchronous way or asynchronously using the interface for asynchronous operation support</td>
</tr>
<tr>
<td>collectMetaInformationPeriodic</td>
<td>This operation gives the possibility for periodic collection of meta information from the given source. It should only be called asynchronously</td>
</tr>
<tr>
<td>getNavigationRoots</td>
<td>This operation returns the features that can be used to start navigation inside the catalogue. If none is returned, no navigation will be possible</td>
</tr>
<tr>
<td>getNavigationEdges</td>
<td>This operation, given an existing node in the catalogue, returns all relationships that start from this node to other ones. Each relationship is qualified by the kind of relationship</td>
</tr>
<tr>
<td>improveQuery</td>
<td>This operation returns semantically connected keywords related to a given search request</td>
</tr>
</tbody>
</table>

*Table 16 - ORCHESTRA Catalogue Service operations*
A 'read-only' CSW ISO AP Catalogue Service has just to provide operations labelled as 'CSW'. In addition, a transactional CSW Catalogue Service needs to provide operations labelled as 'CSWT'. The additional management functions, providing a standardized interface for the active management (transaction by a push-model) or the passive management (harvesting pull model) of metadata is mandated to be an optional part of the CSW ISO AP. The CSW ISO AP Transaction and Harvest operations are part of the manage use case of the Discovery Service. This offers ease of use and the technical possibility for Member States’ public authorities to link and manage (create, delete and update) their metadata of spatial resources to the MS Discovery Service repository. Both the Transaction and Harvest operations of the CSW ISO AP are mandatory in order to offer Member States and their national data providers a choice between the push or pull management model of Discovery Service.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Requirements</th>
<th>Operation Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>OGC_Service.GetCapabilities</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSW Discovery.DescribeRecord</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSW Discovery.GetDomain</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSW Discovery.GetRecords</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSW Discovery.GetRecordById</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSWT Manager.Harvest</td>
<td>Optional</td>
<td>Mandatory</td>
</tr>
<tr>
<td>CSWT Manager.Transaction</td>
<td>Optional</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>

Table 17 - CSW ISO AP operations requirements for INSPIRE Discovery Services
4.2 Inspire View Service

The INSPIRE Directive [2007/2/EC] asks Member States to create “view services making it possible, as a minimum, to display, navigate, zoom in/out, pan, or overlay viewable spatial datasets and to display legend information and any relevant content of metadata; […] Member States may allow a public authority supplying a service […] to apply charges where such charges secure the maintenance of spatial datasets and corresponding data services, especially in cases involving very large volumes of frequently updated data. Data made available through the view services referred […] may be in a form preventing their reuse for commercial purposes”.

In INSPIRE these services are referred to as View Services.

Summarizing, the goal of view service is to allow requests over geo-referenced data belonging to the themes covered by the INSPIRE Directive Annexes, over a spatiotemporal extension, and provide a visual representation of these data.

The following aspects of a view service should be considered:

- nature of the Metadata;
- availability of the Metadata;
- multiple datasets View Geometry (supported spatial reference systems);
- multiple datasets View Output Format (supported formats and their possible integration);
- temporal data dimension;
- legend availability and handling;
- restriction of access and e-commerce;
- multilingualism;
- relationship with client applications.

A relevant approach related to the View Service (specifications and implementation) is represented by the Orchestra Map & Diagram Service.

Also for the View Service, INSPIRE is providing a draft implementation.

4.2.1 OGC WMS - WebMapping Service

4.2.1.1 Overview

A Web Map Service (WMS) produces maps of spatially referenced data dynamically from geographic information. This International Standard defines a "map" to be a portrayal of geographic information as a digital image file suitable for display on a computer screen. A map is not the data itself. WMS-produced maps are generally rendered in a pictorial format such as PNG, GIF or JPEG, or occasionally as vector-based graphical elements in Scalable Vector Graphics (SVG) or Web Computer Graphics Metafile (WebCGM) formats. [OGC 03-109r1]

4.2.1.2 Specification

Current “OGC Web Map Service” interface recommendation paper version is 1.3.0 and can be found at: http://portal.opengeospatial.org/files/?artifact_id=4756.
This International Standard defines three operations:

a) one returns service-level metadata;
b) another one returns a map whose geographic and dimensional parameters are well-defined;
c) and an optional third operation returns information about particular features shown on a map.

HTTP supports two request methods: GET and POST. One or both of these methods may be offered by a server. In WMS the support for the GET method has to be considered mandatory, while the support for the POST method is optional.

The response to a Web Map Service request is always supposed to be a computer file that is transferred over the Internet from the server to the client. The file may contain text, or it may represent a map image. The type of the returned file shall be indicated by a MIME type string. The return value of a valid Service request shall correspond to the type requested in the FORMAT parameter and in the HTTP environment, the Content-type header of the response shall be exactly the MIME type given in the valid request.

As showed in the following table, three operations are defined for a Web Map Service:

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>OPERATION REQUIREMENT</th>
<th>GOAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>Mandatory</td>
<td>The goal of this operation is to obtain service metadata, which is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a machine-readable (and human-readable) description of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>server's information content and acceptable request parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>values</td>
</tr>
<tr>
<td>GetMap</td>
<td>Mandatory</td>
<td>This operation returns a map</td>
</tr>
<tr>
<td>GetFeatureInfo</td>
<td>Optional</td>
<td>This operation is designed to provide clients of a WMS with more</td>
</tr>
<tr>
<td></td>
<td></td>
<td>information about features in the pictures of maps that were</td>
</tr>
<tr>
<td></td>
<td></td>
<td>returned by previous Map requests</td>
</tr>
</tbody>
</table>

Table 18 – OGC WMS Operations

OGC studied an International Standard to define the implementation of the WMS on a distributed computing platform (DCP) comprising Internet hosts that support the Hypertext Transfer Protocol (HTTP) [OGC 06-042].

Styled Layer Descriptor Specification [OGC 05-078r4] added a new optional operation to the three described in the OGC WMS specifications: DescribeLayer, that returns the feature types of the layer, or layers, specified in a request and the attributes can be discovered with the DescribeFeatureType operation of a WFS interface.

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>OPERATION REQUIREMENT</th>
<th>GOAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DescribeLayer</td>
<td>Optional</td>
<td>This operation returns the feature types of the layer, or layers,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>specified in the request</td>
</tr>
</tbody>
</table>

Table 19 - New WMS Operation added by Styled Layer Description Specification
4.2.1.3 OWS Experiments Reports

The WMS specification [OGC 03-109r1] defines a number of parameters that should be used when calling a WMS service. These parameters are encoded using keyword/value pairs (KVP) for operation requests using HTTP Get.

As stated in “SOAP Experiment Report” [OWS 03-014], the main advantage of HTTP Get vocabulary is that a request is fully represented in single URL. However, the HTTP GET model has various shortcomings, including:

a) it does not provide an easy way of passing structured parameter values to a server;
b) Styled Layer Descriptor (SLD) WMS requests, which include an XML description of the styling, are difficult to encode directly and require that the request URL make reference to a separate SLD URL;
c) proposals to add a Filter parameter have been made more complex by the need to associate specific Filter expressions with specific Layers;
d) length limitations on HTTP URLs may preclude all of the desired parameters from being included in the request.

To overcome the limitations of the HTTP GET binding a HTTP POST binding was developed. This binding and its associated schemas are described in [OGC 02-017r1]. The binding specifies that an HTTP POST message to a WMS server should consist of an XML message. Using XML messages solves the issues discussed above because:

a) it allows additional structure in the request message, thereby by allowing additional functionality;
b) it removes size restrictions;
c) the comma-separated list of Layer names can be replaced by a sequence of XML elements, each of which is either a named or a user-defined Layer, and directly associating Style and Filter information within each Layer. The GetFeatureInfo operation, which includes most of a GetMap request, benefits in a similar way.

Unfortunately, testing during experiments revealed that the XML Schemas developed for the WMS HTTP POST bindings do not work correctly with existing Web Service toolkits. Issues that were encountered included:

a) version and service are defined as attributes on the top-level element (GetMap, GetFeatureInfo, GetCapabilities). This is a problem because some Web Service toolkits, such as Axis, do not support the use of attributes. Note that .NET does support such attributes, but only through the use of its metadata functionality.
b) The exceptions simple type defines an enumeration that includes two values, application/vnd.ogc.se+inimage and application/vnd.ogc.se+xml. Both Axis and .NET map these to constants defined on particular classes. However, these strings are not valid names in Java or C#. In the case of Axis, this will cause a compilation error while .NET mangles the name to make it valid. In either case, this is a critical issue.

It is deemed crucial to support the HTTP GET binding in any future WMS specifications that will be developed. The reasoning for this recommendation is that the HTTP GET binding:

a) provides backwards compatibility with previous versions of WMS;
b) provides a mechanism for clients that do not support SOAP, such as older Web Browsers, to call a WMS server.

4.2.1.4 Change proposals
Following the new W3C standards, OGC has proposed a change request to the proposed WMS specifications, focusing on SOAP [OGC 04-050r1]. The main underlined point is that a Web Map Service may support the “SOAP” protocol.

There are two steps in supporting SOAP:
- to add a definition of the SOAP binding to the WSDL description of the service
- to add a support for SOAP document/literal envelope

This should be the WSDL description of the SOAP binding:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://www.opengis.net/wms/soap"
xmlns:wms-soap="http://www.opengis.net/wms/soap"
xmlns:wms-req="http://www.opengis.net/wms/requests"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
<wsdl:documentation/>
</wsdl:documentation>
<!-- import WMS interface definitions -->
<wsdl:import namespace="http://www.opengis.net/wms/requests" location="./wms-xml-interfaces.wsdl"/>
<!-- ****************************************
Bindings
 **************************************** -->
<wsdl:binding name="WMS_SOAP_Binding" type="wms-req:WMS_XML_Port">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="GetCapabilities">
<soap:operation/>
<wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
<wsdl:fault name="exception">
<soap:fault name="exception" use="literal"/>
</wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetMap">
<soap:operation/>
<wsdl:input>
SOAP support shall be declared between the Capabilities of the services; this will allow the usage of SOAP independently of WSDL.
If SOAP is supported, the services shall advertise this capability.

**SOAP document/literal envelope:** a Web Map Service that support SOAP binding shall accept document/literal SOAP messages which contain a message as defined in the XML encoding for Web Map Service. It is not required for the Web Map Service to support any other feature or extension of the SOAP protocol:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <Operation>
      ...
    </Operation>
  </env:Body>
</env:Envelope>
```

If the result of the operation is an XML document, it shall be included in a SOAP envelope as well:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <Result>
      ...
    </Result>
  </env:Body>
</env:Envelope>
```

If the result is a binary data, the content of the *Body* is not specified and the binary is attached to the SOAP message using SOAP with attachments [SOAP-Attachments].

Adding support for SOAP document/literal binding is very important for the future of OGC WMS: the W3C and industry standard for Web services is WSDL/SOAP/UDDI. Adding SOAP support will
increase the potential user base of this specification, while leaving the WMS without support for these standards is a thread for its mid-term adoption.

4.2.2 Orchestra Map&Diagram Service

4.2.2.1 Overview

“The Map and Diagram Service is a service that dynamically portrays geographic and statistical data using style definitions and symbolisation rules. Its main task is to produce maps and diagrams from geographic data (vector or raster) and/or statistical data (e.g. census data, result of a statistical analysis) as digital image files suitable for display on a computer screen. This service is able to create maps and diagrams based not only on data hosted on the server, but also on data provided by external services (e.g. provided by Feature Access Services) or directly included in the request message as GML (sent as an optional part of the Styled Layer Definition)”. [ORCHESTRA D3.4.3]

The main output of this service is an image document.

Maps and diagrams represent a visualisation of the data; they are not the data itself, thus they are generally rendered in a pictorial format such as PNG (recommended as default format), GIF or JPEG. Portrayal in vector-based graphical elements as Scalable Vector Graphics (SVG) or Web Computer Graphics Metafile (WebCGM) is also allowed if it is acceptable for the service provider.

4.2.2.2 Specification

Current “Map&Diagram Service” specification version by ORCHESTRA is 3.0/2.2 and can be found at the following URL:


The interface specifies the following operations:

- getCapabilities
- getMap
- getDiagram
- getFeatureInfo
- getLegendGraphic
- getLayerDescription
- getStyle

It may be possible that the mapping of this interface to SOAP may differ from the future SOAP mappings defined for OGC Web Services. However, as the SOAP mapping defined in the above mentioned document leaves several things unspecified, the SOAP interface of the Map and Diagram Service goes further and specifies the input and output types for each operation messages as UML and XML Schema types.

Another issue linked with the SOAP platform mapping of the interface lies in the return of the image document. The image document can be attached to the response message (e.g. SOAP with attachments) or it can be sent as reference (URI to the rendered image document on the server). Our recommended approach is to send the URI of the rendered image to overcome the problems and limitations of the SOAP attachments.
4.2.2.3 Implementation

The Map and Diagram Service has been designed as an evolution of (and keeping backward compatibility with) the OGC WMS standard [OGC 03-109r1].


The Web service specifications presented in this document are compliant to WSDL Version 1.1 with SOAP Version 1.1 HTTP binding.

The accepted binding are:
- SOAP Version 1.1 (with attachments) HTTP binding (mandatory)
- HTTP using GET method (optional – for backward compatibility with OGC WMS standard)

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCapabilities</td>
<td>It returns a Capabilities document</td>
</tr>
<tr>
<td>getMap</td>
<td>It returns a map of spatially referenced geographic and thematic information as an image document</td>
</tr>
<tr>
<td>getDiagram</td>
<td>It returns a diagram representation of tabular data as an image document</td>
</tr>
<tr>
<td>getLayerLegend</td>
<td>It returns a legend symbol (corresponding to a layer) as an image document</td>
</tr>
<tr>
<td>getFeatureInfo</td>
<td>It returns information about the features rendered in a certain point of a map or diagram layer</td>
</tr>
<tr>
<td>setLayer</td>
<td>It stores a new data layer on the server</td>
</tr>
<tr>
<td>deleteLayer</td>
<td>It removes an existing data layer from the server</td>
</tr>
<tr>
<td>setStyle</td>
<td>It stores a new style on the server</td>
</tr>
<tr>
<td>deleteStyle</td>
<td>It removes an existing style from the server</td>
</tr>
<tr>
<td>getLayerDescription</td>
<td>It returns a layer description document containing schema information for a layer</td>
</tr>
</tbody>
</table>

Table 20 - Summary of Service Operations

4.2.3 Inspire proposed Implementation

Many factors encourage the adoption of a standards-based approach for the View Services Implementing Rules. First of all, the INSPIRE Directive requires a standards-based approach when possible.

Furthermore, the standards-based approach is particularly applicable to the INSPIRE context considering the following:

a) the standardization activity in the area of viewing maps has reached a very good level of maturity;

b) in Europe the INSPIRE reference materials provided by the INSPIRE community for the establishment of the Implementing Rules show a general endorsement of this international standard by the different European actors of the geographic information domain. Many INSPIRE LMOs and SDICs have already adopted the OGC standard.

A standards-based approach is certainly the best answer to the high-level requirements of INSPIRE as it ensures the conformance to the European and International Standards.
The Network Services Drafting Team has made a proposal for the technical content of the INSPIRE Implementing Rules for View Services; such proposal is available and described in “Draft Implementing Rules for Discovery and View Services (IR1)” [INSPIRE D3.7]. In accordance with this document, the INSPIRE View Service operations shall follow those of the EN ISO 19128 international standard [OGC 03-109r1].

The View Service shall provide the following functions:

<table>
<thead>
<tr>
<th>ISO WMS operations</th>
<th>ISO WMS operation requirements</th>
<th>INSPIRE WMS operation requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>mandatory</td>
<td>mandatory</td>
</tr>
<tr>
<td>GetMap</td>
<td>mandatory</td>
<td>mandatory</td>
</tr>
<tr>
<td>GetFeatureInfo</td>
<td>optional</td>
<td>optional</td>
</tr>
</tbody>
</table>

*Table 21 - View Service operations*

These three operations shall use parameters defined in the ISO 19128 WMS standard.
4.3 Inspire Transformation Services

The INSPIRE Directive asks “Member States to establish and operate a network of … transformation services, enabling spatial data sets to be transformed with a view to achieving interoperability”.

Generalisation services could be classified as transformation services too.

Following [INSPIRE D3.5], the transformation services shall be combined with the other services such in a way to enable all those services to operate in conformity with the Implementing Rules laying down the following:
   a) harmonised spatial data specifications;
   b) arrangements for the exchange of spatial data.

This set of services is the one with the widest range of potential interpretation, as it can go from the change of coordinate systems, to schema mapping, encoding or generalisation.

Proposed Assumptions
The allowed transformations would be from a set of National SRS to the harmonised spatial data theme “coordinate reference systems” and from the harmonised spatial data theme “coordinate reference systems” to the same set of National SRS. An example for the latter is to help an end user to use a specific tool, working directly on the original data set, using the original SRS.

Up to now the meaning of INSPIRE Transformation Services is not completely clear and under discussion. The actual understanding addresses Web services that can be service-chained for schema transformation and coordinate transformation. These services are offered as “translating download service”, where the translation is configured in the service and the client has (nearly) no control over the translation.

In this chapter, four services will be analyzed, that can fit in the Inspire Transformation Services domain: OGC Web Coordinates Transforming Services, OGC Web Processing Service, ORCHESTRA Schema Mapping Service and ORCHESTRA Processing Service. Each of these services focuses or covers one of the neuralgic aspects of the Inspire Transformation Services.

4.3.1 OGC WCTS - Web Coordinates Transforming Services

The OGC WCTS Implementation Specification [OGC 01-009] provides interfaces for general positioning, coordinates systems and coordinates transformations that can handle 2D and 3D coordinates, as well as 4D, 5D, etc.

The interfaces for Coordinate Transformation Services are split into three packages:
   a) Positioning (PT);
   b) Coordinate Systems (CS);
   c) Coordinate Transformations (CT).

OGC specification refers to packages that have been profiled for COM, CORBA and Java and then focuses on common data structures and typical mathematical transformations. There are no specifications for a possible service-based implementation. In this sense this OGC service differs from the ones described in the previous chapters.

4.3.2 Orchestra Schema Mapping Service
4.3.2.1 Overview
As stated in [ORCHESTRA SMS SPEC], the ORCHESTRA Schema Mapping Service (SMS) provides functionalities that are related to the mapping of features from a source into a target schema.

The main (and hence mandatory) functionality of this interface is to execute a schema mapping. We consider a schema mapping to be “the definition of an automated transformation of each instance of a data structure $A$ into an instance of a data structure $B$ that preserves the intended meaning of the original information” (Doerr, 2004). This is the main difference with the Processing Service, which also transforms data from one schema to another one, but usually changes the semantics of the information.

A schema mapping is described by
- an identifier that is unique to the Schema Mapping Service instance;
- descriptions of the source and target feature types;
- the schema mapping language used to describe the mapping;
- a reference to the actual mapping.

The Schema Mapping Service can be used to
a) directly map from one application schema to another one;
b) map from an application schema to a common (or community) schema (or vice versa). This situation can be used to perform an indirect mapping between two application schemas through the community schema.

The features mapping might also require that several feature collections are combined. In order to support this possibility, an (optional) concatenation operation is also included in the service interface. The description of the schema mapping is required as an input, because it is outside the scope of the Schema Mapping Service to automatically derive a mapping between two application schemas.

4.3.2.2 Specification and implementation
The Web service specifications presented in [ORCHESTRA IS SMS] are compliant to WSDL Version 1.1 with SOAP Version 1.1 HTTP binding; WSDL used version is 1.1.

The message style that shall be used is Document-Literal non-wrapped; this choice has the consequence that each operation can take only one single input parameter (an XML element).

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCapabilities</td>
<td>This mandatory operation informs the client of the capabilities of an ORCHESTRA Service Instance (OSI)</td>
</tr>
<tr>
<td>mapFeatures</td>
<td>This operation maps a feature collection from a local to a target schema using a predefined mapping rule</td>
</tr>
<tr>
<td>concatFeatures</td>
<td>Not implemented</td>
</tr>
<tr>
<td>createMapping</td>
<td>This operation registers in the repository a mapping between a source feature descriptor and a target feature descriptor, for a specific schema mapping language</td>
</tr>
<tr>
<td>deleteMapping</td>
<td>This operation removes permanently zero or more mappings identified by the specified query</td>
</tr>
<tr>
<td>getMapping</td>
<td>This operation returns a collection of mapping descriptors identified by the specified query. If no query is specified all registered mappings are returned</td>
</tr>
</tbody>
</table>
The `setMapping` operation is optional in the abstract specification and is not implemented.

The `createMapping` and the `mapFeatures` operations contain multiple input parameters, which are represented as multiple parts messages in the WSDL. The SOAP protocol allows multiple parts messages only if SOAP message style is defined as “RPC”, which is not the case of the SMS: this service is using “Document-Literal non-wrapped” style message which only allows single part messages. Therefore, the input parameters are wrapped into one single request type.
4.3.3 Other Relevant OGC Services

OGC Web Processing Service (and Orchestra Processing Service) should also be considered as relevant for the Inspire Transformation Services, as the Network Services Draft Team is currently proposing to use it for both schema and coordinate transformation. Anyway, due to the current state of play and to the nature of the WPS, these services are reported in this document in the Inspire Invoke Services section.
4.4 Inspire Invoke Services

The INSPIRE Directive asks “Member States to establish and operate a network of […] services allowing spatial data services to be invoked”.

The Invoke Spatial Data Services service supports invoking individual (spatial) services, as well as combinations of individual (spatial) services, both synchronous and asynchronous, in service chains through a (Web) service orchestration engine a.k.a. “workflow engine”. The service chains are expressed in a standard (e.g. XML-based) notation that can be consumed by commercial, as well as open-source, orchestration engines from multiple sources.

The Invoke Spatial Data Services service allows also the definition of both the data inputs and data outputs expected by the spatial services and the external Web service interface of the service chain.

For spatial data services available on the Internet, the Invoke Spatial Data Services service will enable a user or a client application to run them without requiring the availability of a GIS. This requires that a client application can discover the service, bind to it and invoke it. The orchestration/combination of Spatial Data Services with other services will require to precisely defining the interactions between them.

OGC Web Processing Service has the same goal of Inspire Invoke Services, so it will be analysed in the following of this chapter, together with its ORCHESTRA implementation.

4.4.1 OGC WPS - Web Processing Service

4.4.1.1 Overview

A WPS provides client access across a network to pre-programmed calculations and/or computation models that operate on spatially referenced data. The calculation can be extremely simple (as subtracting one set of spatially referenced numbers from another), or highly complex (as a global climate change model), with any number of data inputs and outputs.

The data required by the service can be delivered across a network, or available at the server. This data can use image data formats or data exchange standards such as Geography Mark-up Language (GML).

4.4.1.2 Specification

[OGC 05-007r7] specifies that the platform-neutral data needed as input or output for the services could be encoded in many alternative ways: each appropriate to one or more specific DCPs. Possible choices are HTTP GET transfer of operations requests (using KVP encoding) and HTTP POST transfer of operations requests (using XML or KVP encoding), but the same operation requests and responses (and other data) could be encoded for other specific computing platforms, including SOAP/WSDL.

The WPS interface specifies three operations that can be requested by a client and performed by a WPS server; all of them should be mandatorily implemented by all WPS servers:

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetCapabilities</td>
<td>This operation allows a client to request and receive back service metadata (or Capabilities) documents that describe the abilities of the specific server implementation. The GetCapabilities operation provides the names and general descriptions of each of the processes offered by a WPS instance. This operation also supports negotiation of the specification version being used for client-server interactions.</td>
</tr>
<tr>
<td>DescribeProcess</td>
<td>This operation allows a client to request and receive back detailed information about the processes that can be run on the service instance, including the inputs required, their allowable formats, and the outputs that can be produced.</td>
</tr>
<tr>
<td>Execute</td>
<td>This operation allows a client to run a specified process implemented by the WPS, using provided input parameter values and returning the outputs produced.</td>
</tr>
</tbody>
</table>

These operations have many similarities to other OGC Web Services, including WMS, WFS and WCS.

WPS specification describes a generic interface, in that it does not identify any specific process that is supported. Instead, each implementation of WPS should define the processes that are supported, as well as their associated inputs and outputs. WPS can be thought of as an abstract model of a Web service, for which profiles need to be developed, to support use, and standardized, to support interoperability.

### 4.4.1.3 Implementation

WPS discovery and binding mechanisms follow the OGC model set by WMS and WFS, in that WPS defines a GetCapabilities operation and requests are based on HTTP GET and POST. WPS does more than just describing the service interface. It specifies a request/response interface that defines how to:

- encode requests for process execution;
- encode responses from process execution;
- embed data and metadata in process execution inputs/outputs;
- reference Web-accessible data inputs/outputs;
- support long-running processes;
- return process status information;
- return processing errors;
- request storage of process outputs.

WPS allows for the provision of input data in two different methods. **Data can either be embedded in the Execute request, or referenced as a Web accessible resource.** In the former approach, WPS acts as a stand-alone service. In the latter fashion, WPS acts as middleware service for data, by obtaining data from an external resource in order to run a process on the local implementation. WPS allows existing software interfaces to be wrapped up and presented to the network as Web services. Implementations of WPS can thus be considered middleware for software.

WPS is compatible with both WSDL and SOAP: SOAP can be used to package WPS requests and responses.

WPS describes a message exchange mechanism that can be used if SOAP is not required (for security such as encryption or authentication), but it goes beyond SOAP, specifying what the payload should look like. Elements that are common to all payloads have been generalized in the WPS specification.
The use of SOAP to wrap WPS requests offers the ability to add security certificates, as well as encryption to Web-based geo-processing transactions.

The encoding of operation requests shall use HTTP GET with KVP encoding and HTTP POST with XML encoding as specified in Clause 11 of [OGC 06-121r3]. Table 24 summarizes the three Service operations and their encoding methods defined in the specification (encodings reported in bold are mandatory).

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>REQUESTED ENCODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET-KVP and POST-XML</td>
</tr>
<tr>
<td>DescribeProcess</td>
<td>GET-KVP and POST-XML</td>
</tr>
<tr>
<td>Execute</td>
<td>POST-XML and GET-KVP</td>
</tr>
</tbody>
</table>

Table 24 - Operations request encoding

The outputs can be returned in the form of an XML response document, either embedded within the response document, or stored as Web accessible resources.

WPS requests and responses encoded in SOAP shall use SOAP Document-style encoding (also called message-style or Document-Literal encoding), as described in [OGC 06-094], for the following operations:

- GetCapabilities request
- GetCapabilities response
- DescribeProcess request
- DescribeProcess response
- Execute response, when status=true

An example for a DescribeProcess request follows:

```xml
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsi=http://www.w3.org/2001/XMLSchema
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <DescribeProcess service="WPS" version="1.0.0" xmlns="http://www.opengeospatial.net/wps
xmlns:ows="http://www.opengeospatial.net/ows" xmlns:xlink="http://www.w3.org/1999/xlink
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://www.opengeospatial.net/wps ..\wpsDescribeProcess.xsd">
      <ows:Identifier>intersection</ows:Identifier>
      <ows:Identifier>union</ows:Identifier>
    </DescribeProcess>
  </soap:Body>
</soap:Envelope>
```

WPS execute requests encoded in SOAP shall be encoded as follows:

- the process name shall be turned into an element in the SOAP body by pre-pending the text "ExecuteProcess_";
• each input and output shall be encoded as an element in the SOAP body by using the Identifier as the name of the element.

SOAP requests to execute a process when RawDataOutput is requested shall generate a SOAP error.

4.4.1.4 Change requests/proposals
No change requests/proposals have been submitted.

4.4.2 Orchestra Processing Service
ORCHESTRA has followed all the OGC specification for the Web Processing Service, producing a WSDL implementing the pre-defined operations, using SOAP as transportation protocol.

All the three operations have been implemented:

<table>
<thead>
<tr>
<th>ORCHESTRA PROCESSING SERVICE</th>
<th>OGC WPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>execute</td>
<td>Execute</td>
</tr>
<tr>
<td>getProcessDescription</td>
<td>DescribeProcess</td>
</tr>
<tr>
<td>getCapabilities</td>
<td>GetCapabilities</td>
</tr>
</tbody>
</table>

Table 25 - ORCHESTRA PS - OGC WPS Operations mapping

The used encoding is Document-Literal for each operation, as specified in [OGC 05-007r7]. Complex data for services input or output can be included in the SOAP messages or can be referenced by URLs.
4.5 Inspire Download Service

The INSPIRE Directive asks Member States “to establish and operate a network” of “download services, enabling copies of spatial data sets, or parts of such sets, to be downloaded and, where practicable, accessed directly”.

A download service supports:
- download of a complete dataset or datasets;
- download of a part of a dataset or datasets;
- where practicable, provide direct access to complete datasets or parts of datasets;
- gazetteer-like services are also covered by a type of download service.

It is worth to note that either the conceptual, or application, or logical schema of the local or national spatial data set may and will often differ from the INSPIRE harmonised specification of the spatial object types in the data specification. In this case a download service will transform queries and data between the conceptual or application schema of the spatial dataset and the harmonized schema on-the-fly. Search criteria need to support searching based on spatial and temporal extents.

OGC provides a couple of services that allow having access to two different kinds of geospatial information: the Web Coverage Service and the Web Feature Service. These services and their possible correlated ORCHESTRA implementation will be analyzed in the following sections.

4.5.1 OGC WCS - Web Coverage Service

4.5.1.1 Overview

The Web Coverage Service (WCS) supports electronic retrieval of geospatial data as "coverages" – that is, digital geospatial information representing space-varying phenomena.

A WCS provides access to potentially detailed and rich sets of geospatial information, in forms that are useful for client-side rendering, multi-valued coverages and input into scientific models and other clients.

The WCS may be compared to the OGC Web Map Service (WMS) and the OGC Web Feature Service (WFS): like them it allows clients to choose portions of a server's information holdings based on spatial constraints and other criteria. However, unlike the WMS [OGC 06-042], which portrays spatial data to return static maps (rendered as pictures by the server), the Web Coverage Service provides available data together with their detailed descriptions, defines a rich syntax for requests against these data and returns data with their original semantics (instead of pictures), which may be interpreted, extrapolated, etc. – and not just portrayed.

Unlike WFS [OGC 04-094], which returns discrete geospatial features, the Web Coverage Service returns coverages representing space-varying phenomena that relate a space-temporal domain to a (possibly multidimensional) range of properties.

4.5.1.2 Specification

As specified by [OGC 07-067r5], the Web Coverage Service provides three operations:

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>This operation returns an XML document describing the service and brief descriptions of the coverages that clients may request</td>
</tr>
</tbody>
</table>
DescribeCoverage
This operation lets clients request a full description of one or more coverages served by a particular WCS server. The server then responds with an XML document that fully describes the identified coverages.

GetCoverage
This operation returns a coverage (that is, values or properties of a set of geographic locations), encoded in a well-known coverage format.

### Table 26 - WCS operations

<table>
<thead>
<tr>
<th>Description</th>
<th>HTTP Binding</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>DescribeCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>GetCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
</tbody>
</table>

### 4.5.1.3 Implementation

All WCS operation requests may use HTTP GET with KVP encoding, or use HTTP POST with XML, as specified in Clause 11 of [OGC 06-121r3].

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>HTTP BINDING</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>DescribeCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>GetCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
</tbody>
</table>

| Table 27 - WCS Operations Encodings |

**GetCapabilities**

The mandatory GetCapabilities operation allows WCS clients to retrieve service metadata from a WCS server. The response to a GetCapabilities request shall be an XML document containing service metadata about the server, usually including summary information about the data collections from which coverages may be requested.

All WCS servers shall implement HTTP GET transfer of the GetCapabilities operation request, using KVP encoding. Servers may also implement HTTP POST transfer of the GetCapabilities operation request, using XML encoding only.

When the HTTP POST connect point URL is different for different encodings of the operation requests, this <ows:Constraint> element shall be included in each <Post> element. When the connect point URL is the same for all encodings of all operation requests, this <ows:Constraint> element should be included in the <OperationsMetadata> element.

When a WCS server encounters an error while performing a GetCapabilities operation, it shall return an exception report message as specified in Clause 8 of [OGC 06-121r3].

**KVP encoding**

The KVP encoding of a WCS GetCapabilities operation request shall be as specified in Table 5 in Subclause 7.2.2 of [OGC 06-121r3].

**XML encoding**

The XML Schema fragment for encoding a WCS GetCapabilities operation request extends <ows:GetCapabilitiesType> from [OGC 06-121r3]:

```xml
<element name="GetCapabilities">
  <annotation>
    <documentation>
```

### Table 26 - WCS operations

<table>
<thead>
<tr>
<th>Description</th>
<th>HTTP Binding</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>DescribeCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
<tr>
<td>GetCoverage</td>
<td>GET-KVP or POST-XML</td>
</tr>
</tbody>
</table>

---

Page 41 of 75
Request to a WCS to perform the GetCapabilities operation. This operation allows a client to retrieve a Capabilities XML document providing metadata for the specific WCS server. In this XML encoding, no "request" parameter is included, since the element name specifies the specific operation.

DescribeCoverage / GetCoverage

WCS servers may implement either KVP encoding for HTTP GET transfer, or HTTP POST using XML transfer of DescribeCoverage and GetCoverage operations requests.

When a WCS server encounters an error while performing a DescribeCoverage operation, it shall return an exception report message as specified in Subclause 7.4 of [OGC 06-121r3].

4.5.1.4 Change requests/proposals

The following indications were added to the OGC WCS specification [OGC 07-067r5], after that a change proposal [OGC 06-085r2] had been submitted by OWS.

All compliant WCS servers may implement SOAP 1.2 transfer of all WCS operation requests and responses. When SOAP is implemented, the SOAP Request-Response message exchange pattern shall be used with the HTTP POST binding.

If a WCS server implements SOAP version 1.2 transfer of an operation request, it shall also implement SOAP transfer of the corresponding operation response.

For SOAP transfer, each XML-encoded operation request or response shall be encapsulated in the body of a SOAP envelope, which shall contain only one body and only this request or response in that body.

A WCS server shall return operation responses and error messages using only SOAP transfer when the operation request is sent using SOAP; the proposed implementation of the SOAP protocol uses document/literal encoding.

All compliant WCS servers shall specify the URLs to which SOAP operation requests may be sent, within the OperationsMetadata section of a service metadata (Capabilities) XML document. When an error is detected while processing an operation request encoded in a SOAP envelope, the WCS server shall generate a SOAP response message where the content of the <Body> element is a <Fault> element containing an <ExceptionReport> element.

This shall be done using the following XML fragment:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```
The `<Code>` element shall have the Value “soap:server” indicating that this is a server exception.

The `<Reason>` element shall have the Text “Server exception was encountered.” This fixed string is used since the details of the exception shall be specified in the `<Detail>` element using an `<ows:ExceptionReport>` element as specified in OWS Common [OGC 06-121r3].

All WCS servers shall specify the encodings that may be sent using HTTP POST transfer of operation requests. Specifically, an `<ows:Constraint>` element shall be included, with `PostEncoding` as the value of the name attribute and specifying different allowed values for each allowed encoding:

a) the value “SOAP” shall indicate that SOAP encoding is allowed;

b) the value “XML” shall indicate that XML encoding is allowed (without SOAP message encapsulation).

The value “KVP” is not used, since this WCS standard does not allow KVP encodings of operation requests to be transferred using HTTP POST.

In the `GetCoverage` operation, when the `store` parameter is absent or has the value “false”, the server shall transfer the complete `GetCoverage` response to the client, either as a MIME multipart message (for KVP or XML requests), or as a SOAP message with attachments (for SOAP requests). The Coverages shall reference the other parts of the MIME multi-part message (or SOAP attachments). When the `store` parameter has the value “true”, the body of the SOAP envelope response shall contain one XML encoded Coverages element with one or more references to URLs of files containing coverage content and metadata.

When the `store` parameter is absent or has the value “false”, responses to `GetCoverage` SOAP requests shall be encoded as SOAP with Attachments as defined in [W3C Note] (but using SOAP 1.2 rather than SOAP 1.1). These responses shall consist of MIME multipart messages, with the Coverages element contained within a SOAP 1.2 envelope.

Another change request was posted by OWS, specifying the morphology of the possible SOAP binding for WCS in a better way [04-049r1]. WCS may support the “SOAP” protocol and **the implementation of the SOAP protocol shall use Document-Literal encoding.**
4.5.2 OGC WFS - Web Feature Service

4.5.2.1 Overview
As [OGC 04-094] declares, the OGC Web Map Service allows a client to overlay map images for display, served from multiple Web Map Services on the Internet. In a similar fashion, the OGC Web Feature Service allows a client to retrieve and update geospatial data encoded in Geography Markup Language (GML) from multiple Web Feature Services.

The requirements for a Web Feature Service are:
- the interfaces must be defined in XML;
- GML must be used to express features within the interface;
- at a minimum a WFS must be able to present features using GML;
- the predicate or filter language will be defined in XML and be derived from CQL, as defined in the OpenGIS Catalogue Interface Implementation Specification;
- the data-store used to store geographic features should be opaque to client applications and their only view of the data should be through the WFS interface;
- the use of a subset of XPath expressions for referencing properties.

The WFS operations support INSERT, UPDATE, DELETE, LOCK, QUERY and DISCOVERY operations on geographic features using HTTP as the distributed computing platform.

4.5.2.2 Specification
To support transaction and query processing, the following operations are defined:

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>A Web feature service must be able to describe its capabilities. Specifically, it must indicate which feature types it can service and what operations are supported on each feature type</td>
</tr>
<tr>
<td>DescribeFeatureType</td>
<td>A Web feature service must be able, upon request, to describe the structure of any feature type it can service.</td>
</tr>
<tr>
<td>GetFeature</td>
<td>A Web feature service must be able to service a request to retrieve feature instances. In addition, the client should be able to specify which feature properties to fetch and should be able to constrain the query spatially and non-spatially</td>
</tr>
<tr>
<td>GetGmlObject</td>
<td>A Web feature service may be able to service a request to retrieve element instances by traversing XLinks that refer to their XML IDs. In addition, the client should be able to specify whether nested XLinks embedded in returned element data should also be retrieved</td>
</tr>
<tr>
<td>Transaction</td>
<td>A Web feature service may be able to service transaction requests. A transaction request is composed of operations that modify features; that is create, update, and delete operations on geographic features</td>
</tr>
<tr>
<td>LockFeature</td>
<td>A Web feature service may be able to process a lock request on one or more instances of a feature type for the duration of a transaction. This ensures that serializable transactions are supported</td>
</tr>
</tbody>
</table>

Table 28 - WFS Operations
HTTP supports two request methods: GET and POST. One or both of these methods may be defined for a particular Web Feature Service and offered by a service instance.

In addition to, or instead of offering Web feature services using the HTTP protocol, a service provider may offer Web feature services using HTTPS.

### 4.5.2.3 Implementation

There are two methods of encoding WFS requests: the first one uses XML as the encoding language and is intended to be used with HTTP POST method. The second encoding uses keyword-value pairs (KVP) to encode the various parameters of a request and is intended to be used with HTTP GET.

Table 29 correlates WFS operations and their encoding semantics as defined in OGC specification [OGC 04-094].

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>REQUEST ENCODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>XML &amp; KVP</td>
</tr>
<tr>
<td>DescribeFeatureType</td>
<td>XML &amp; KVP</td>
</tr>
<tr>
<td>GetGmlObject</td>
<td>XML &amp; KVP</td>
</tr>
<tr>
<td>LockFeature</td>
<td>XML &amp; KVP</td>
</tr>
<tr>
<td>Transaction</td>
<td>XML &amp; limited KVP</td>
</tr>
</tbody>
</table>

Table 29 - WFS Operations Request Encoding

[OGC 04-094] mandates the use of GML for the XML encoding of the state of geographic features.

The following matrix correlated WFS request encodings with each of the supported HTTP methods (GET and POST). The value in each cell defines the expected MIME type for the encoding/request method combination. The value “Not Applicable” means that the encoding/request method combination is supported but a MIME type is not applicable. An empty cell indicates that the combination is not supported.

<table>
<thead>
<tr>
<th></th>
<th>HTTP GET</th>
<th>HTTP POST</th>
</tr>
</thead>
<tbody>
<tr>
<td>XML encoded requests</td>
<td>text/xml</td>
<td></td>
</tr>
<tr>
<td>KVP encoded requests</td>
<td>Not applicable</td>
<td>application/x-www-form-urlencoded</td>
</tr>
</tbody>
</table>

Table 30 - Request encoding and transport methods

When using the HTTP POST method, the content type for XML encoded WFS requests must be set to text/xml. When using the HTTP POST method, the content type for KVP encoded WFS requests must be set to application/x-www-form-urlencoded and the content of the document must be equivalent to the query string of an HTTP GET request.

When using the HTTP GET method and KVP encoded WFS requests, a MIME type is not applicable as the entire request is encoded in the URL as keyword-value pairs that follow the ‘?’ character. The combination of XML encoded requests and the HTTP GET method is not supported.

**getCapabilities**

Every Web Feature Service must support the KVP encoded form of the *GetCapabilities* request over HTTP GET so that a client can always know how to obtain a capabilities document.
The following section defines the XML encoding for a GetCapabilities request and defines the service metadata generated in response as an XML document that a Web Feature Service must generate to describe its capabilities.

Request

The GetCapabilities element is used to request a capabilities document from a Web Feature Service. It is defined by the following XML Schema fragment:

```
<xsd:element name="GetCapabilities" type="wfs:GetCapabilitiesType"/>
<xsd:complexType name="GetCapabilitiesType">
  <xsd:complexContent>
    <xsd:extension base="ows:GetCapabilitiesType">
      <xsd:attribute name="service" type="ows:ServiceType" use="optional" default="WFS"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

The base type, ows:GetCapabilitiesType, is defined in the OWS Common Implementation Specification [Murata et al., 2001].

Response

The root element of the response to a GetCapabilities request is the <WFS_Capabilities> element. It is partially defined be the following XML Schema fragment:

```
<xsd:element name="WFS_Capabilities" type="wfs:WFS_CapabilitiesType" substitutionGroup="ows:Capabilities"/>
<xsd:complexType name="WFS_CapabilitiesType">
  <xsd:complexContent>
    <xsd:extension base="ows:CapabilitiesBaseType">
      <xsd:sequence>
        <xsd:element ref="wfs:FeatureTypeList" minOccurs="0"/>
        <xsd:element ref="wfs:ServesGMLObjectTypeList" minOccurs="0"/>
        <xsd:element ref="ows:Filter_Capabilities"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

The base type, ows:CapabilitiesBaseType, is defined in the OWS Common Implementation Specification [Murata et al., 2001].

4.5.2.4 Change Proposals/Requests

OWS has posted a couple of change requests about SOAP usage in WFS. In [OGC 04-041] it’s underlined that the compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only “Document-Literal” or “RPC-Literal” bindings are employed. The former style
entails little more than wrapping a WFS message with a SOAP envelope. In the latter style, the child element of the <soap:Body> element consists of a wrapper element bearing either the name of the operation or the name of the operation suffixed with "Response"; the corresponding message parts appear in the guise of child accessory elements.

When specifying a binding for an interface, the WS-I Basic profile doesn't allow any operation to be omitted, even though the WSDL content model does. Compliance with the Basic profile would rule out the use of piecemeal bindings; that is, all operations within the interface must be bound to that protocol (e.g. HTTP GET).

One consequence of this is that it may be desirable to place operations that are commonly bound to the HTTP GET method within a separate interface.

Message encodings: KVP vs. XML

Most WFS messages admit two encoding styles: an XML representation (application/xml) based on an XML Schema type definition, or a key-value pair (KVP) representation that can be readily incorporated within a Request-URI (application/x-www-form-urlencoded). Unfortunately these usually have inconsistent content models so they must be defined separately (e.g. GetCapabilities KVP representation uses different names and adds the REQUEST parameter).

In light of this aspect, retrieval operations that are commonly bound to the GET method may also require alternative message definitions (e.g., GetCapabilitiesRequest-KVP).

[OGC 04-094] specification has included the following guidelines, after that OWS proposed them in [OGC 06-087]: a client may send WFS requests to a compatible server using the body of a SOAP envelope. The client simply encodes the WFS request as the content of the <soap:Body> element in the request message.

The WFS may then response by generating a SOAP message where the response to the client’s request is the content of the <soap:Body> element.

In the event that an exception is encountered while processing a WFS request encoded in a SOAP envelope, the Web Feature Service must generate a SOAP response message where the content of the <soap:Body> element is a <soap:Fault> element. The following skeleton XML fragment must be used when generating the <soap:Body> element in the event that a WFS encounters an exception:

```
<soap:Body>
  <soap:Fault>
    <soap:faultcode>soap:Server</soap:faultcode>
    <soap:faultstring>A server exception was encountered.</soap:faultstring>
    <soap:detail>
      <ows:ExceptionReport>
        ...
      </ows:ExceptionReport>
    </soap:detail>
  </soap:Fault>
</soap:Body>
```

The <soap:faultcode> element must have the content “soap:Server” indicating that this is a server exception. The <soap:faultstring> element must have the content “Server exception was encountered.”. This fixed string is used since the details of the exception will be specified in the
<soap:detail> element, using an <ows:ExceptionReport> element as defined in document [Murata et al., 2001].
The <soap:detail> element must contain an <ows:ExceptionReport> element detailing the specific exception that the server encountered.
The use of the soap:Header element has not been discussed in the change requests.

4.5.3 Orchestra Feature Access Service

4.5.3.1 Overview
The Feature Access Service (FAS) allows interoperable read and write access on feature instances available in an OSN. [ORCHESTRA FAS SPEC]
The FAS offers information about:
- the feature types it is capable to provide or support;
- the supported encoding(s) to transfer requested or submitted feature data;
- the query language and mechanism that is supported for filtered feature access.

The Feature Access Service allows queries to select certain features based on their type, certain attribute values and their spatial and temporal extent.
More on, any Feature Access Service may support the update of existing feature instances, the creation of new feature and the deletion of existing features and hence, in this case, it should also be transactional. It can also allow creation, updates and deletions of feature types.

4.5.3.2 Specification
The functionality of the Feature Access Service (FAS) is based on the WFS and WCS OGC implementation specifications. These specifications allow the retrieval of features and coverages respectively. At the abstract level, coverages and features are considered as ORCHESTRA features and thus an interface has been developed to access to both types.
The write functionalities of the WFS specification (which basically consist of a transactional operation) have been transferred into three operations setFeatures, createFeatures and deleteFeatures, as to follow the ORCHESTRA convention of operation functionality. Additionally, the objective was to put the “write behavior” of the WFS at the operation level in the interface.
Currently, in the OGC WFS specification, the write type of a given operation (i.e. insert, update, or delete) is specified as a parameter to a more generic operation (transaction operation).

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>NAME DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFeatureTypes</td>
<td>This operation gets a description (the schema) of given feature types serviced by a Feature Access Service instance</td>
</tr>
<tr>
<td>setFeatureTypes</td>
<td>This operation updates existing Feature Types</td>
</tr>
<tr>
<td>createFeatureTypes</td>
<td>This operation creates new Feature Types</td>
</tr>
<tr>
<td>deleteFeatureTypes</td>
<td>This operation deletes existing Feature Types</td>
</tr>
<tr>
<td>getFeatures</td>
<td>This operation retrieves features and their attributes</td>
</tr>
<tr>
<td>setFeatures</td>
<td>This operation updates existing features</td>
</tr>
<tr>
<td>createFeatures</td>
<td>This operation creates new features</td>
</tr>
<tr>
<td>deleteFeatures</td>
<td>This operation deletes existing features</td>
</tr>
</tbody>
</table>

Table 31 - WFS Summary of additional Service Operations
4.5.3.3 Implementation

As specified in [ORCHESTRA FAS IS], the implementation of the Feature Access Service is compliant to WSDL Version 1.1 with SOAP Version 1.1 HTTP binding.

The possible bindings are:
- SOAP Version 1.1 HTTP binding (mandatory)
- HTTP using POST method and XML encoding (optional)
- HTTP using GET method with KVP encoding (optional)

<table>
<thead>
<tr>
<th>OPERATION</th>
<th>REQUIRED ENCODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFeatureTypes</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>setFeatureTypes</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>createFeatureTypes</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>deleteFeatureTypes</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>getFeatures</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>setFeatures</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>createFeatures</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
<tr>
<td>deleteFeatures</td>
<td>SOAP 1.1 (POST+XML or GET+KVP)</td>
</tr>
</tbody>
</table>

Table 32 - FAS Operations Encodings

All the operations described in the specification document [ORCHESTRA FAS SPEC] have been implemented.
5. Critical Review

In this chapter a multi-dimensional analysis will be performed over all the Inspire services, focusing on their SOAP implementation choices/suggestions. Firstly, such options are analysed for each of the Inspire Services, including choices undertaken by relevant OGC and ORCHESTRA services. After that, some common issues related to the use of SOAP have been identified and analysed (such as SOAP header, exceptions, data encoding and binary data management).

N.B.: in the next tables, red sentences are not reported from the native service specification, but derive from OWS change requests/proposals.

5.1 Inspire Discovery Service

5.1.1 Analysis of OGC-CWS implementations

Following the OGC CSW specifications, three application profiles have been produced. Their implementation choices are quite homogeneous and do not have substantial differences between them, but considering SOAP binding as mandatory or not. OGC specifications of CSW already foresee the importance of SOAP binding and they explicitly support this protocol as an optional choice, beside the standard HTTP GET/POST methods. This choice has influenced the OGC Application Profiles as much as both ISO METADATA AP and AP for EO Products consider SOAP support mandatory, while ebRIM recommendations follow the specifications guidelines and allow freedom in the use of SOAP. As ebRIM AP is the profile that has the more generic view about protocol bindings, it requires a proper use of the Content-Type message header to specify the kind of content housed in the message.

In case SOAP is supported, all the examined specifications state that SOAP messages must be transferred using HTTP(S) POST.

Some minor differences, instead, arise on the SOAP data encoding: while the OGC specifications do not assert anything about it, as much as ebRIM AP, ISO METADATA AP calls for a Document/Literal style, while EO AP is more lenient and suggests just a Document style, without specifying anything about parameters encoding.

All AP specifications (OGC CSW doesn’t deal with this issue) agree on putting response and request messages inside the SOAP <Body> element, as much as the specifications dealing with the “error handling” topic agree on using a <soap:Fault> element in the SOAP Body when an exception is raised.

OWS ISO METADATA AP follows quite strictly the HTTP GET/POST choices for the non-SOAP implementation and for parameter data encodings, while the other profiles are vaguer on this topic and follow in a less severe way the OGC specifications.

The following table summarises the analysis:
<table>
<thead>
<tr>
<th>Supported bindings</th>
<th>OGC IMPL SPECIFICATIONS</th>
<th>OWS ISO METADATA AP</th>
<th>OWS ebRIM AP</th>
<th>AP for EO Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Both HTTP GET/POST and SOAP are supported.</td>
<td>SOAP encoding of operation requests are eventually transferred using HTTP POST (except getCapabilities based on GET+KVP)</td>
<td>All operations must support embedding of requests and responses in SOAP 1.2 messages via HTTP/POST</td>
<td>SOAP 1.2 messaging framework may be supported</td>
<td>SOAP over HTTP or HTTPS is used and mandatory</td>
</tr>
<tr>
<td>Operations specific protocol and parameters encoding</td>
<td>(in brackets optional choices)</td>
<td>(in brackets optional choices)</td>
<td>(in brackets optional choices, no specifications for parameters data encoding)</td>
<td>(only two CSW operations have been mapped; it is not anticipated to require a KVP encoding)</td>
</tr>
<tr>
<td>GetCapabilities:</td>
<td>GET + KVP (POST + XML)</td>
<td>GET + KVP (POST + XML)</td>
<td>GET (POST)</td>
<td>n/a</td>
</tr>
<tr>
<td>DescribeRecord:</td>
<td>POST + XML (GET + KVP)</td>
<td>POST + XML (GET + KVP)</td>
<td>POST (GET)</td>
<td>n/a</td>
</tr>
<tr>
<td>GetDomain:</td>
<td>POST + XML (GET + KVP)</td>
<td>POST + XML (GET + KVP)</td>
<td>GET (POST)</td>
<td>n/a</td>
</tr>
<tr>
<td>GetRecords:</td>
<td>POST + XML (GET + KVP)</td>
<td>POST + XML (GET + KVP)</td>
<td>GET (POST)</td>
<td>POST + XML (“SearchRequest”)</td>
</tr>
<tr>
<td>GetRecordById:</td>
<td>GET + KVP (POST + XML)</td>
<td>GET + KVP and POST + XML</td>
<td>GET (POST)</td>
<td>POST + XML (“PresentRequest”)</td>
</tr>
<tr>
<td>Harvest:</td>
<td>POST + XML (GET + KVP)</td>
<td>POST + XML (POST + KVP)</td>
<td>GET</td>
<td>n/a</td>
</tr>
<tr>
<td>Transaction</td>
<td>POST + XML</td>
<td>POST + XML</td>
<td>POST</td>
<td>n/a</td>
</tr>
<tr>
<td>SOAP message structure</td>
<td>Messages are stored in the Body of SOAP envelope (spec added as consequence of change proposed by [OGC 06-089])</td>
<td>Messages are stored in the Body of SOAP envelope</td>
<td>Responses are XML-encoded</td>
<td>not defined</td>
</tr>
<tr>
<td>SOAP Errors handling</td>
<td>In SOAP, exceptions are stored in the SOAP envelope Body as soap:Fault element</td>
<td>no remarks</td>
<td>In SOAP, exceptions are stored in the SOAP envelope Body as soap:Fault element.</td>
<td>not defined</td>
</tr>
<tr>
<td></td>
<td>SOAP Data encoding</td>
<td>Security</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------</td>
<td>--------------------</td>
<td>---------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td></td>
<td><em>not defined</em></td>
<td><em>not defined</em></td>
<td>Document/Literal style</td>
<td><em>not defined</em></td>
</tr>
<tr>
<td>SOAP style should be</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>&quot;Document&quot;</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 33 – Analysis of OGC-CWS profiles*
5.1.1.1 Summary of OWS CSW CHANGE PROPOSALS/REQUESTS

Some change proposals have been requested [OWS 06-089], to align the OGC CSW specifications with some choices taken by the examined Application Profiles.

Some of changes are related to the use of the SOAP **Body** as storage of CSW requests, and SOAP **Fault** for error handling: these have already been incorporated in the OGC-CWS specifications [07-006r1].

Other changes are linked with the necessity of making available a functionality by which the server can indicate if SOAP requests **can be managed using HTTP POST or not**.

Finally, another request is about SOAP **data encoding**, a topic that was not treated in 2.0.2 specifications. The change proposal leaves the choice open between *Document-Literal* and *RPC-Literal* encoding. [04-042]

It should be noticed that the EO AP draft implementation rules define only *Document* encoding style.

5.1.2 ORCHESTRA Catalogue Service

This implementation specification is based on SOAP 1.1 HTTP binding and *Document-Literal* SOAP encoding.

The most relevant remark is that the version proposed for SOAP is still 1.1, while current standard is 1.2.

Another difference is due to the fact that ORCHESTRA Catalogue service is defining more operations than OGC-CSW, as shown:

<table>
<thead>
<tr>
<th>OGC-CSW OPERATION</th>
<th>ORCHESTRA CATALOGUE OPERATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>GetCapabilities</td>
</tr>
<tr>
<td>GetRecords</td>
<td>search</td>
</tr>
<tr>
<td>GetRecordById</td>
<td>getMetaInformation</td>
</tr>
<tr>
<td>GetDomain</td>
<td>getQueryDomain</td>
</tr>
<tr>
<td>DescribeRecord</td>
<td>getMetaInformationType</td>
</tr>
<tr>
<td>Transaction (insert)</td>
<td>createMetaInformation</td>
</tr>
<tr>
<td>Transaction (update)</td>
<td>setMetaInformation</td>
</tr>
<tr>
<td>Transaction (delete)</td>
<td>deleteMetaInformation</td>
</tr>
<tr>
<td>Harvest</td>
<td>collectMetaInformation</td>
</tr>
<tr>
<td>- collectMetaInformationPeriodic</td>
<td></td>
</tr>
<tr>
<td>- getNavigationRoots</td>
<td></td>
</tr>
<tr>
<td>- getNavigationEdges</td>
<td></td>
</tr>
<tr>
<td>- improveQuery</td>
<td></td>
</tr>
</tbody>
</table>

*Table 34 – OGC-WMS vs. ORCHESTRA Map&Diagram operations*

5.1.3 INSPIRE Proposed Implementation
INSPIRE implementation rules have asserted that the **INSPIRE Catalogue Service should follow CSW ISO AP specifications**, considered as a good standard. This implies:

- use of SOAP 1.2 binding on HTTP POST;
- encoding style as `Document/Literal` (considered the most fitting one for geo-information context in most of OGC deliverables and discussion papers).

Beside SOAP, previous HTTP GET and POST bindings with KVP or XML parameters are anyway maintained for coherence with servers that do not support SOAP.

5.2 **Inspire View Service**

OGC WMS specification is less detailed regarding the use of SOAP than the OGC CSW one; moreover there are no Application Profiles or further specification refinements for such service type. The analysis is thus performed amongst OGC WMS specification, OWS change proposals (as consequence of OWS experimentation [OWS 03-014]) and the ORCHESTRA “Map & Diagram” service.

ORCHESTRA specifications of Map&Diagram Service already consider the use of SOAP binding, even if it does not provide many details on SOAP conformity choices or recommendations. OGC WMS Specifications, instead, do not consider the possibility of using SOAP, while they just support HTTP GET as compulsory binding (KVP parameter encoding) and POST as optional. That’s why a change proposal has been redacted to fix this lack.

Also for this service ORCHESTRA refers to SOAP 1.1 (while latest one is 1.2).

It should also be noticed that ORCHESTRA is defining more operations than OGC, as shown below:

<table>
<thead>
<tr>
<th><strong>OGC-WMS OPERATION</strong></th>
<th><strong>ORCHESTRA MAP &amp; DIAGRAM OPERATION</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCapabilities</td>
<td>getCapabilities</td>
</tr>
<tr>
<td>GetMap</td>
<td>getMap</td>
</tr>
<tr>
<td>GetMap</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>getDiagram</td>
</tr>
<tr>
<td></td>
<td>getLayerLegend</td>
</tr>
<tr>
<td>GetFeatureInfo</td>
<td>getFeatureInfo</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>DescribeLayer</td>
<td>getLayerDescription</td>
</tr>
</tbody>
</table>

Table 35 – OGC-WMS vs. ORCHESTRA Map&Diagram operations

Regarding message structure, the only recommendation OGC specifications provide is that the `Content-type` header of the response shall be exactly the MIME type given in the valid request.

Taking into consideration the change request and ORCHESTRA approaches, we can find a substantial difference: both of them analyse the possible problem bound to the binary nature of the service response. OWS change proposal does not give options: if the result is a binary data, the content of the `Body` is not specified and the binary is attached to the SOAP message using SOAP
with attachments. On the other hand, ORCHESTRA leaves the choice to the implementer between SOAP with attachments or to send the file as a reference, but recommends this latter option. OWS change request is the only one to face the SOAP data encoding theme, stating that Document/Literal encoding shall be accepted.

The following table summarises the analysis:
<table>
<thead>
<tr>
<th>Supported bindings</th>
<th>OGC-WMS SPECIFICATIONS</th>
<th>OWS WMS CHANGE PROPOSALS</th>
<th>ORCHESTRA “MAP &amp; DIAGRAM”</th>
</tr>
</thead>
<tbody>
<tr>
<td>In WMS the support for the GET method has to be considered mandatory, while the support for the POST method is optional</td>
<td>WMS may support the SOAP protocol</td>
<td>It supports SOAP 1.1 via HTTP/POST</td>
<td></td>
</tr>
<tr>
<td>Operations specific protocol and parameters encoding</td>
<td>Parameters are encoded using keyword/value pairs (KVP) for operation requests using HTTP GET</td>
<td>//</td>
<td>HTTP using GET method is optional</td>
</tr>
<tr>
<td>(SOAP) message structure</td>
<td></td>
<td>Requests and responses included in the SOAP message Body</td>
<td>The image document can be attached to the response message (e.g. SOAP with attachments) or it can be sent as reference (URI to the rendered image document on the server)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the result is a binary data, the content of the Body is not specified and the binary is attached to the SOAP message using SOAP with attachments</td>
<td>The ORCHESTRA recommended approach is to send the URI of the rendered image</td>
</tr>
<tr>
<td>SOAP Errors handling</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
</tr>
<tr>
<td>SOAP Data encoding</td>
<td>not defined</td>
<td>WMS that support SOAP binding shall accept document/literal SOAP messages</td>
<td>not defined</td>
</tr>
<tr>
<td>Security</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
</tr>
<tr>
<td>Other Topics</td>
<td>//</td>
<td>If SOAP is supported, the services shall advertise this capability</td>
<td>//</td>
</tr>
</tbody>
</table>

*Table 36 – Analysis of Inspire View Service related specifications*
5.3 Inspire Transformation Services

Following the goals described in the Inspire directive, the Transformation Services should base on two different kinds of OGC services and on one service implemented by ORCHESTRA: OGC Web Coordinate Transformation Services, OGC Web Processing Service and ORCHESTRA Schema Mapping Service.

The goals of these three kinds of services are very different, but combined together they should cover the whole needs of Inspire Transform Service:

- **OGC WCTS** offers the interfaces to transform geo-spatial data to perform transformations under a geo-spatial point of view, for example to convert them from a 3D coordinate system to a 2D one.
- **OGC WPS** allows invoking one or more generic manipulation algorithms on geo-spatial data, offering the possibility to invoke service chains too. These algorithms can be of different kinds like a simple subtraction of one set of spatially referenced numbers from another, or highly complex, as a global climate change model.
- **ORCHESTRA SMS**, instead, is very different compared with the OGC Processing Service, which also transforms data from one schema to another one, but usually changes the semantics of the information. The Schema Mapping Services, rather, change the representation of the data, perhaps modifying the data structure by which they are represented, but preserve the data semantic.

OGC WCTS implementation specification document [OGC 01-009] does not provide an in depth description of the transport mechanism related to this service, but just focuses on the operations interfaces and their parameters: no approach with SOAP is specified and no possible issues or suggestions are proposed.

On the ORCHESTRA side, instead, the SOAP HTTP binding is as usual explicitly recommended even in SMS, but as usual proposing the 1.1 SOAP version and providing Web service specifications compliant to WSDL Version 1.1.

OGC WPS specification [OGC 05-007r7] states that the platform-neutral data needed as input or output for the services could be encoded in many alternative ways: it foresees the possibility of using SOAP as well, but constraints the use of GET with KVP and POST with XML pairs, as specified in Clause 11 of [OGC 06-121r3].

ORCHESTRA provided a WSDL of a faithful implementation of WPS services, where a SOAP binding with *Document-Literal non-wrapped* style has been used.

ORCHESTRA SMS documentation is very unambiguous about the message style that shall be used for SOAP data: *Document-Literal non-wrapped*, even if this choice has the consequence that each operation takes only one single input parameter (an XML element).

Conversely, WPS requests and responses encoded in SOAP shall use SOAP *Document-Literal encoding*, as described in [OGC 06-094], without taking into consideration the problem of wrapped or non-wrapped style.

For both OGC WPS specification and ORCHESTRA Processing Service WSDL, the values of complex data structures can be (either) directly encoded in the operation requests, or made available through a Web accessible URL.

The outputs can be returned in the form of an XML response document, either embedded within the response document, or stored as Web accessible resources.
<table>
<thead>
<tr>
<th>Supported bindings</th>
<th>OGC WCTS SPECIFICATION</th>
<th>ORCHESTRA SMS</th>
<th>OGC WPS SPECIFICATION</th>
<th>ORCHESTRA PROCESSING SERVICE WSDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>not defined</td>
<td></td>
<td>It supports SOAP 1.1 via HTTP POST</td>
<td>In WPS the support for the GET (KVP) and POST (XML) methods has to be considered mandatory.</td>
<td>SOAP</td>
</tr>
<tr>
<td>Operations specific protocol and parameters encoding</td>
<td>not defined</td>
<td>not defined</td>
<td>GetCapabilities</td>
<td>GET+KVP (POST+XML) //</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DescribeProcess</td>
<td>GET+KVP (POST+XML) //</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Execute</td>
<td>POST+XML (GET+KVP) //</td>
</tr>
<tr>
<td>(SOAP) message structure</td>
<td>not defined</td>
<td>not defined</td>
<td>Complex data can be directly encoded or referenced by URL</td>
<td>Complex data can be directly encoded or referenced by URL</td>
</tr>
<tr>
<td>SOAP Errors handling</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>//</td>
</tr>
<tr>
<td>SOAP Data encoding</td>
<td>not defined</td>
<td>Document-Literal non-wrapped</td>
<td>Document-Literal</td>
<td>Document-Literal non-wrapped</td>
</tr>
<tr>
<td>Security</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>//</td>
</tr>
<tr>
<td>Other Topics</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>//</td>
</tr>
</tbody>
</table>

*Table 37 - OGC and ORCHESTRA WCTS and WPS services summary*
5.4 Inspire Invoke Service

The mission of INSPIRE Invoke Services can be assimilated to the OGC WPS one, because this service can perform its duty invoking other services to provide its input, generating a sort of service chain.

The analysis of the OGC Web Processing Service has been already performed in the previous paragraph, but it will be repeated here after, without considering possible relationships with OGC WCTS and ORCHESTRA SMS.

The conformance between OGC WPS specification and ORCHESTRA WSDL implementation is quite total under the SOAP profile point of view.

Complex data can be accessed either by URL, or encoding their XML representation directly in the message; a Document-Literal style is applied, but while OGC didn’t retain important to specify whether a wrapped encoding should be used, it looks like ORCHESTRA followed the way already undertaken in SMS: non-wrapped style.

No other SOAP/transportation aspects have been marked in any of both the analysed documents.

<table>
<thead>
<tr>
<th>Supported bindings</th>
<th>OGC WPS SPECIFICATION</th>
<th>ORCHESTRA PROCESSING SERVICE WSDL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In WPS the support for the GET (KVP) and POST (XML) methods has to be considered mandatory.</td>
<td>SOAP</td>
</tr>
<tr>
<td></td>
<td>WPS may support SOAP</td>
<td></td>
</tr>
<tr>
<td>Operations specific protocol and parameters encoding</td>
<td>(in brackets optional choices)</td>
<td>No GET or POST encoding specified</td>
</tr>
<tr>
<td>GetCapabilities</td>
<td>GET+KVP (POST+XML)</td>
<td>//</td>
</tr>
<tr>
<td>DescribeProcess</td>
<td>GET+KVP (POST+XML)</td>
<td>//</td>
</tr>
<tr>
<td>Execute</td>
<td>POST+XML (GET+KVP)</td>
<td>//</td>
</tr>
<tr>
<td>(SOAP) message structure</td>
<td>Complex data can be directly encoded or referenced by URL</td>
<td>Complex data can be directly encoded or referenced by URL</td>
</tr>
<tr>
<td>SOAP Errors handling</td>
<td>not defined</td>
<td>//</td>
</tr>
<tr>
<td>SOAP Data encoding</td>
<td>Document-Literal</td>
<td>Document-Literal non-wrapped</td>
</tr>
<tr>
<td>Security</td>
<td>not defined</td>
<td>not defined</td>
</tr>
<tr>
<td>Other Topics</td>
<td>//</td>
<td>//</td>
</tr>
</tbody>
</table>

Table 38 - OGC WPS and ORCHESTRA PS service comparison
5.5 **Inspire Download Service**

INSPIRE Download Service allows the user to download and have access to a set of geospatial data or to a part of them.

Two OGC services and one ORCHESTRA service are taken into consideration for this goal: OGC Web Coverage Service, OGC Web Feature Service and ORCHESTRA Feature Access Service.

OGC WCS specification [OGC 07-067r5] approached only the use of HTTP GET with KVP encoding, or HTTP POST with XML encoding, as specified in Clause 11 of [OGC 06-121r3], but a change report [OGC 06-085r2] added the possibility for all compliant WCS servers to implement SOAP 1.2 transfer of all WCS operation requests and responses, over HTTP POST. Anyway all WCS servers shall implement HTTP GET transfer of the *GetCapabilities* operation request, using KVP encoding.

Also in OGC WFS specification [OGC 04-094] both HTTP GET and POST methods are allowed to be offered by a Web Feature Service instance and after that OWS proposed in [OGC 06-087] the possibility to send WFS requests to a compatible server using the body of a SOAP envelope, the two OGC services specifications have been lined up.

Both the services specifies that for SOAP transfer, each XML-encoded operation request or response shall be encapsulated in the body of a SOAP envelope, which shall contain only one body and this body should contain only that request or response.

In the [OGC 04-041] WFS change proposal, it’s underlined that the compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only Document-Literal or RPC-Literal bindings are employed, while WCS specifies the opportunity to use the Document-Literal encoding.

All compliant WCS servers shall specify the URLs to which SOAP operation requests may be sent, within the *OperationsMetadata* section of a service metadata (Capabilities) XML document.

Another aspect on which both the specifications agree is that when an error is detected, during the processing an operation request encoded in a SOAP envelope, the servers shall generate a SOAP response message where the content of the `<Body>` element is a `<Fault>` element. WCS specification adds that this `<Fault>` element must contain an `<ExceptionReport>` element.

WFS focuses also on the possibility of using SOAP on HTTPS to improve security for its messages and mandates the use of GML for the XML encoding of the state of geographic features.

ORCHESTRA provided an implementation of OGC WFS and WCS based on SOAP 1.1, with optional HTTP GET and POST bindings, but that does not enter in the details of peculiar aspects like data encoding or SOAP message structure.

<table>
<thead>
<tr>
<th>Supported bindings</th>
<th>OGC WCS SPEC</th>
<th>OGC WFS</th>
<th>ORCHESTRA FAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Both HTTP GET and POST (and SOAP) are supported</td>
<td>Both HTTP GET and POST (and SOAP) are supported</td>
<td>SOAP 1.1 is mandatory HTTP POST and GET are optional</td>
<td></td>
</tr>
<tr>
<td>Operations specific protocol and parameters encoding</td>
<td><em>(in brackets optional choices)</em></td>
<td><em>(in brackets optional choices)</em></td>
<td>SOAP 1.1 (POST + XML or GET + KVP)</td>
</tr>
<tr>
<td>GetCapabilities:</td>
<td>GET + KVP (POST + XML) (POST + SOAP)</td>
<td>GET + KVP (POST + XML) (POST + SOAP)</td>
<td>//</td>
</tr>
<tr>
<td>DescribeCoverage:</td>
<td>(GET + KVP or POST + XML) (POST + SOAP)</td>
<td>//</td>
<td>//</td>
</tr>
</tbody>
</table>
5.6 Common Issues

As stated in “OWS common change request: add SOAP encoding” [OWS 06-094r1], OWS understood that SOAP encoding is critical for the survival of OGC services, so SOAP encoding needs to be foreseen and implemented.

OWS specifications shall indicate that servers may optionally implement SOAP 1.2 transfer of all operation requests and responses, using the same XML encodings as specified for use with HTTP POST. When SOAP encoding is implemented, the SOAP Request-Response message exchange pattern shall be used with the HTTP POST binding.

The following **common rules** should be followed by all the OGC Web services.

For SOAP transfer, each XML-encoded operation request shall be encapsulated in the `<Body>` element of a SOAP envelope; in other words, the *SOAP-Body* shall be used only for transmitting the actual OWS service request.

When the operation request is sent using SOAP, an implementing server shall return operation responses and error messages using only SOAP transfer.

All compliant OWS servers shall specify the URLs to which SOAP-encoded operation requests may be sent, within the *OperationsMetadata* section of a service metadata (Capabilities) XML document.

---

<table>
<thead>
<tr>
<th><strong>SOAP message structure</strong></th>
<th><strong>SOAP Errors handling</strong></th>
<th><strong>SOAP Data encoding</strong></th>
<th><strong>Security</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages are stored in the <em>Body</em> of SOAP envelope</td>
<td>In SOAP, exceptions are stored in the SOAP envelope <em>Body as soap:Fault</em> element</td>
<td><em>Document-Literal</em></td>
<td>not defined</td>
</tr>
<tr>
<td>Binary data can be encoded as URL reference or with SOAP with Attachment in function of the parameter “store” of GetCoverage</td>
<td>In SOAP, exceptions are stored in the SOAP envelope <em>Body as soap:Fault</em> element</td>
<td>“<em>Document-Literal</em>” or “<em>RPC-Literal</em>”</td>
<td>WFS may be offered using HTTPS</td>
</tr>
<tr>
<td>It’s mandatory the use of GML for the XML encoding of the state of geographic features</td>
<td><em>soap:Header</em> use is not discussed</td>
<td>not defined</td>
<td>not defined</td>
</tr>
</tbody>
</table>

*Table 39 - OGC WCS, OGC WFS and ORCHESTRA FAS service comparison*
5.6.1 SOAP Header

The optional SOAP-Header shall be used for elective elements (NOT parameters), in order to invoke the service and could be targeted at any SOAP receiver in the SOAP message path and the same behaviour can be though even for the response messages. Actually no OGC service specification document or change proposal examines the SOAP Header theme, an aspect that instead should be analysed to provide a common use inside the Inspire domain. SOAP headers could be used for security, like data encryption, for documentation, providing descriptive comments on the content of the message, for policies, affecting behaviour of the targeted receiver, not tightly bound to message semantics, for adding message meta-data, or for “piggybacking”, providing information (potentially) unrelated to the message body.

5.6.2 Exception report SOAP encoding

If an error is detected while processing an operation request encoded in a SOAP envelope, the OWS server shall generate a SOAP response message where the content of the <Body> element is a <Fault> element containing an <ExceptionReport> element.

This shall be done using the following XML fragment:

```xml
<soap:Envelope xmlns:soap=http://www.w3.org/2003/05/soap-envelope xmlns:ows=http://www.opengis.net/ows/1.2>
  <soap:Body>
    <soap:Fault>
      <soap:Code>
        <soap:Value>soap:Server</soap:Value>
      </soap:Code>
      <soap:Reason>
        <soap:Text>A server exception was encountered.</soap:Text>
      </soap:Reason>
      <soap:Detail>
        <ows:ExceptionReport>
          ...
        </ows:ExceptionReport>
      </soap:Detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

The <Code> element shall have the Value “soap:server” indicating that this is a server exception. The Reason element shall have the Text “Server exception was encountered”. This fixed string is used since the details of the exception shall be specified in the Detail element using an ows:ExceptionReport element.

5.6.3 Protocol Binding

OWS provides a report [OGC 04-060r1] on the use of SOAP in various OGC services, focussing mainly on WMS, WFS and WCS. The conclusions is that XML bindings (e.g. SOAP) seem to be the way to go to; they should be used where possible and taken into account when defining new services; they offer all the flexibility
needed in terms of message complexity and permit to leverage the XML schema definitions of the
existing services. However, KVP bindings cannot be overlooked, as many existing services support only such
bindings, and we cannot expect all those services to be upgraded to SOAP. Since a unique abstract part cannot be defined for both KVP and XML bindings, specific message
types will have to be defined.

This leads to a situation where HTTP GET/POST and SOAP bindings will probably co-exist in parallel for a long time.

There is a common consensus within OGC to integrate existing Web service standards as much as possible, but incorporating those standards can on one hand certainly provide solutions for SOAP-based OGC services, but, on the other hand, it would disregard all HTTP GET and POST-based services, which currently are in the majority.

Another way to support HTTP GET and POST-based services would be to define an easy and generic approach to provide these services with a SOAP binding, which would enable a transformation of any HTTP GET and POST services to a SOAP service. Such a generic approach would even allow building generic wrapper components, which could be applied to any existing HTTP GET and POST service and to the according clients, so that the transformation to SOAP could be completely transparent to the client application.

The way to perform such a mapping is exhaustively described in [OGC 07-158] and [OGC 08-009r1].

In [OGC 08-009r1] there’s an important aspect to be remarked about the Capabilities documents. A Capabilities document contains for each operation a ‘DCPType’ node which defines the binding for this operation. The sub node ‘HTTP’ still remains, since also the SOAP binding will be available via HTTP, but the sub node under the HTTP node has to be changed from ‘Get’ or ‘Post’ to ‘SOAP’. The sub node under ‘Get’ or ‘Post’ contains the URL to the described operation. This URL has to be changed to the current SOAP wrapper URL.

5.6.3.1 getCapabilities
All OWS servers shall specify the encodings that may be sent using HTTP POST transfer of operation requests. Specifically, an <ows:Constraint> element shall be included, with “PostEncoding” as the value of the “name” attribute and specifying different allowed values for each allowed encoding:
   a) the value “SOAP” shall indicate that SOAP encoding is allowed;
   b) the value “XML” shall indicate that XML encoding is allowed (without SOAP message encapsulation);
   c) the value “KVP” shall indicate that KVP encoding is allowed.

If the HTTP POST connect point URL is different for different encodings of the operation requests, a
<ows:Constraint> element shall be included in each <Post> element; while if the connect point URL is the same for all encodings of all operation requests, the <ows:Constraint> element shall be included in the <OperationsMetadata> element.

5.6.4 Data Encoding
Data encoding is one of the aspects undertaken in all the OGC/ORCHESTRA services.
SOAP 1.2 specifications allow different styles and encodings: the “OWS common change request: add SOAP encoding” [OWS 06-094r1] suggests for the use of SOAP with OGC Web Services only the Document Literal-Wrapped style.

The compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only Document-Literal or RPC-Literal bindings are employed in the Web services implementation and this advice has been respected by all the specification and implementation documents analysed. The most widespread choice is Document-Literal, as probably it’s along the whole Internet, while RPC-Literal is considered a possible option just by CSW change request and by WFS specification.

The main point to be addressed appears to be the choice between Document-Literal Wrapped or non-Wrapped encoding. The largest part of the examined documents do not enter in this depth of detail, but ORCHESTRA appears to be more incline to a non-Wrapped choice, while the only time OGC expresses its opinion on this topic in a common discussion paper [06-094r1], it proposes a Wrapped style.

<table>
<thead>
<tr>
<th></th>
<th>OGC SPEC</th>
<th>ORCH.</th>
<th>INSPIRE DRAFTS</th>
<th>CHANGE REQ/PROP</th>
<th>Other0</th>
<th>Other1</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMS</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>Document-Literal [04-050r1]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WCTS</td>
<td>//</td>
<td></td>
<td>Document-Literal non-Wrapped [IS SMS]</td>
<td>//</td>
<td>//</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 1 - Document-RPC-Literal-Encoded space choices disposition
Specific documents report:

**Common**
- “A WSDL SOAP binding can be either an RPC style binding or a document style binding. Furthermore, A SOAP binding can have an encoded use or a literal use. Of the four resulting combinations, document/literal binding was chosen for OGC messages, because it is the one that matches best the way transport layers are designed in current OGC services (XML messages validated by OGC schemas) and because it allows for direct use of OGC schemas in the SOAP messaging scheme” [04-060r1].
- “OWS-2 pointed out document literal as the OGC binding of choice since it matches best the OGC Web Service world. This document follows this argumentation and **recommends Document/Literal style** for SOAP bindings” [08-009r1].
- “For the use of SOAP with OGC Web Services only the **Document Literal-Wrapped style** shall be used” [06-094r1].
- “Only SOAP messaging (via HTTP/POST) with **document/literal style** shall be used” [INSPIRE D3.7].

**CSW**
- “Compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only “document-literal” or “rpc-literal” bindings be employed (R2705). For the rpc-literal encoding style, the schema alone is not sufficient to validate the message body—the RPC rules must be known. Given the state of current SOAP tools, interoperability issues may arise” [04-042].
- “Apart from the GetCapabilities operation, all operations must support the embedding of requests and responses in SOAP messages. In this case only SOAP messaging (via HTTP/POST) with **document/literal style** has to be used” [05-057r3].
- “The message-based variant of SOAP will be used. This means that **style="document" are used** in the WSDL file as attribute of the <binding> element” [OGC 07-045].

**WMS**
- “This change **aims to add** support for SOAP document/literal binding” [04-050r1].

**WCTS**
- “The message style that **shall be** used is **document/literal non-wrapped**. This has the consequence that each operation takes only one single input parameter (an XML element)” [IS SMS].

**WPS**
- “WPS requests and responses encoded in SOAP **shall use SOAP document-style** encoding (also called message-style or **document-literal encoding**)” [05-007r7].
- “The used encoding is **Document-Literal for each operation**” [PRO WSDL].
• “The proposed implementation of the SOAP protocol uses document/literal encoding” [OGC 04-049r1]

WFS
• “Compliance with the WS-I Basic Profile v1.0 [WSI-Basic] requires that only “Document-Literal” or “RPC-Literal” bindings are employed” [OGC 04-041].

5.6.5 Binary Data

Another relevant topic to be analyzed is the way to transmit big binary data in SOAP messages. Many different solutions have been proposed, from MTOM and XOP to SOAP with Attachments, passing from uploading the file on a server and sending the reference by URL.

The topic has not been discussed in those services where large amount of binary data are not foreseen to be exchanged.

Anyway, according to the “OWS common change request: add SOAP encoding” [OWS 06-094r1], for binary data sent via a SOAP message, the Message Transportation Optimization Mechanism (MTOM) shall be used, in conjunction with XML-binary Optimized Packaging (XOP).

<table>
<thead>
<tr>
<th></th>
<th>OGC SPEC</th>
<th>ORCHESTRA</th>
<th>INSPIRE DRAFTS</th>
<th>CHANGE REQ/PROP</th>
<th>Var0</th>
<th>Var1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td></td>
<td>-</td>
<td>-</td>
<td>MTOM + XOP</td>
<td>SwA [04-060r1]</td>
<td>MTOM + XOP (+Fast Infoset) [08-009r1]</td>
</tr>
<tr>
<td>CSW</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>//</td>
</tr>
<tr>
<td>WMS</td>
<td>//</td>
<td>SwA</td>
<td>SwA</td>
<td>SwA</td>
<td>SwA [04-050r1]</td>
<td>-</td>
</tr>
<tr>
<td>WCTS</td>
<td>//</td>
<td>/ URL reference</td>
<td>/ URL reference</td>
<td>/ URL reference</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WPS</td>
<td>SwA / URL reference [05-007r7]</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WCS</td>
<td>SwA / URL reference [06-085r2]</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WFS</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>//</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 41 - Binary data management summary

Common
• “Using SOAP with attachments (SwA) seems the way to go, and WSDL documents created in the scope of OWS-2 were designed with that approach in mind. However, support for MIME messages in existing XML tools (e.g. BPEL) should be investigated” [04-060r1];

• “This document recommends OGC specifications to use the XOP specification as the basis for defining attachments as binary content when defining HTTP binding for their interfaces” [08-009r1];
• “this IPR recommends the use of **MTOP** (see section 6.3) and **XOP** (see section 6.4) to efficiently transfer (large) binary data with SOAP” [08-009r1];
• “To reduce the actual on-the-wire payload, **Fast Infoset** (see section 6.5) **should be considered** as a potential solution for future Web Services” [08-009r1];
• “For binary data send via a SOAP message, the Message Transportation Optimization Mechanism (**MTOM**) **shall be used in conjunction with** **XML-binary Optimized Packaging (XOP)”** [06-094r1].

**WMS**
• “If the result is a Binary, the content of the Body is not specified and the binary is attached to the SOAP message using **SOAP with attachment”** [04-050r1];
• “The image document can be attached to the response message (e.g. SOAP with attachments) or it can be sent as reference (URI to the rendered image document on the server). Our recommended approach is to send the URI of the rendered image to overcome the problems and limitations of the SOAP attachments” [ORCHESTRA D3.4.3].

**WPS**
• “The values of complex data structures can be (either) directly encoded in the operation requests, or made available through a Web accessible URL” [05-007r7];
• “The outputs can be returned in the form of an XML response document, either embedded within the response document, or stored as Web accessible resources” [05-007r7];
• “Data can either be embedded in the Execute request, or referenced as a web accessible resource” [05-007r7].

**WCS**
• “When the store parameter has the value “true”, the body of the SOAP envelope shall contain **one XML encoded Coverages element with one or more references to URLs of files containing coverage content and metadata”** [OGC 06-085r2].
• “When the store parameter is absent or has the value “false”, responses to **GetCoverage** SOAP requests shall be encoded as SOAP with Attachments as defined in [W3C Note] (but using SOAP 1.2 rather than SOAP 1.1). These responses shall consist of MIME multipart messages, with the Coverages element contained within a SOAP 1.2 envelope” [OGC 06-085r2].
6. Conclusions

Usually other systems interact with a Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Despite of these guidelines, the analysis performed in this document revealed that:

1. OGC services have been designed and proposed when SOAP was not a standard yet, so their specifications are not compliant with this protocol, so obviously a possible SOAP-based implementation is not taken into account.
2. Several change requests and common discussion papers have been proposed by OWS to propose a SOAP-based implementation of OGC services: version 1.2 has been considered the way to follow, but not all SOAP related issues have been analysed (i.e. data encoding, binary data transport, etc...).
3. The quantity and quality of SOAP-based experimentation and proposals is quite different when addressing the various OGC services. Indeed, OGC CSW (related to Inspire Discovery Service) has the richest set of documentation: this is due to the several Application Profiles already implemented or studied (by OWS and ORCHESTRA), that have already analysed most of the issues connected to the use of SOAP.
4. Other OGC reports ([OGC 03-014], [OGC 04-060r1] and [OGC 08-009r1]) are related to a general use of SOAP amongst OGC services. Anyway they do not cover in detail all the SOAP problematics, like the common use of the SOAP Header, or the choice between wrapped or non-wrapped data encoding style.

As a preliminary conclusion of the analysis, what could constitute the basis for the definition of a common framework for future Inspire services based on SOAP binding is as follows:

- SOAP v1.2 and WSDL v2.0
- Document-Literal wrapper data-encoding
- MTOM + XOP + Fast Infoset for binary data transport and XML compression
- HTTPS could be used as lower level protocol to improve security
- use of a common SOAP Header
- faults messages should be managed inside SOAP bodies

A consolidated and detailed definition of a proposal for an INSPIRE SOAP framework will be the object of a separate document.
## Annex A – Terms, Definitions and Abbreviations

<table>
<thead>
<tr>
<th>ACRONYM</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>Application Profile</td>
</tr>
<tr>
<td>CCSO</td>
<td>Computing and Communications Services Office</td>
</tr>
<tr>
<td>CQL</td>
<td>Continuous Query Language</td>
</tr>
<tr>
<td>DCP</td>
<td>Distributed Computing Platform</td>
</tr>
<tr>
<td>EO</td>
<td>Earth Observation</td>
</tr>
<tr>
<td>GML</td>
<td>Geography Mark-up Language</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>LMO</td>
<td>Legally Mandated Organisations</td>
</tr>
<tr>
<td>NS DT</td>
<td>Network Services Draft Team</td>
</tr>
<tr>
<td>OSN</td>
<td>Orchestra Service Network</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>SDIC</td>
<td>Spatial Data Interest Community</td>
</tr>
<tr>
<td>SLD</td>
<td>Styled Layer Descriptor</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
</tbody>
</table>

*Table 42 – Document Abbreviations Table*
7. Annex B – Inspire IR Reference

The listed documents are publicly available in the INSPIRE web site [http://inspire.jrc.it/reports.cfm](http://inspire.jrc.it/reports.cfm)

<table>
<thead>
<tr>
<th>REF.</th>
<th>AUTHOR(S)</th>
<th>TITLE</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSPIRE D3.5</td>
<td>NS DT</td>
<td>INSPIRE – Network Services Architecture</td>
<td>2007</td>
</tr>
<tr>
<td>INSPIRE D3.7</td>
<td>NS DT</td>
<td>Draft Implementing Rules for Discovery and View Services</td>
<td>2007</td>
</tr>
<tr>
<td>INSPIRE WPPA</td>
<td></td>
<td>INSPIRE Work Programme Preparatory Phase</td>
<td>2006</td>
</tr>
</tbody>
</table>

*Table 43 - Inspire IR References*
8. Annex C – Bibliography

The following table lists the sources referenced in the present document. All OGC change requests are publicly available at [http://www.opengeospatial.org/standards/cr](http://www.opengeospatial.org/standards/cr) All ORCHESTRA documents are publicly available at [http://www.eu-orchestra.org/documents.shtml](http://www.eu-orchestra.org/documents.shtml)

<table>
<thead>
<tr>
<th>REF.</th>
<th>AUTHOR(S)</th>
<th>TITLE</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Murata et al., 2001</td>
<td>Murata St. Laurent Kohn</td>
<td>XML Media Types</td>
<td>2001</td>
</tr>
<tr>
<td>OGC 01-009</td>
<td></td>
<td>OGC Implementation Specification: Coordinate Transformation Services</td>
<td>2001</td>
</tr>
<tr>
<td>OGC 02-017r1</td>
<td>J. de La Beaujardière</td>
<td>Web Map Service Implementation Specification Part 2: XML for Requests using HTTP POST</td>
<td>2002</td>
</tr>
<tr>
<td>OGC 03-109r1</td>
<td>J. de la Beaujardiere</td>
<td>OGC Web Map Service Interface</td>
<td>2004</td>
</tr>
<tr>
<td>OGC 04-041</td>
<td>R. Martell</td>
<td>OGC WFS 1.x change proposal (OWS-2): WSDL and SOAP usage</td>
<td>2004</td>
</tr>
<tr>
<td>OGC 04-042</td>
<td>R. Martell</td>
<td>OGC CSW 2.x change proposal: WSDL and SOAP usage</td>
<td>2004</td>
</tr>
<tr>
<td>OGC 04-049r1</td>
<td>P. Duschene J. Sonnet</td>
<td>OGC WCS Change Request: Support for WSDL &amp; SOAP</td>
<td>2005</td>
</tr>
<tr>
<td>OGC 04-050r1</td>
<td>P. Duschene J. Sonnet</td>
<td>OGC WMS Change Request: Support for WSDL &amp; SOAP</td>
<td>2005</td>
</tr>
<tr>
<td>OGC 04-094</td>
<td>P. A. Vretanos</td>
<td>Web Feature Service Implementation Specification</td>
<td>2005</td>
</tr>
<tr>
<td>OGC 05-007r7</td>
<td>P. Schut</td>
<td>OGC Web Processing Service</td>
<td>2007</td>
</tr>
<tr>
<td>OGC 05-057r3</td>
<td>J. Martin</td>
<td>Application Profile for EO Products</td>
<td>2005</td>
</tr>
<tr>
<td>OGC 05-078r4</td>
<td>M. Lupp</td>
<td>Styled Layer Descriptor profile of the Web Map Service Implementation Specification</td>
<td>2007</td>
</tr>
<tr>
<td>OGC 06-042</td>
<td>J. de la Beaujardiere</td>
<td>OGC Web Map Server Implementation Specification</td>
<td>2006</td>
</tr>
<tr>
<td>OGC 06-085r2</td>
<td>A. Whiteside</td>
<td>WCS change request: Add SOAP encoding</td>
<td>2006</td>
</tr>
<tr>
<td>OGC 06-087</td>
<td>A. Whiteside</td>
<td>WFS change request: Correct SOAP encoding</td>
<td>2006</td>
</tr>
<tr>
<td>OGC 06-089</td>
<td>A. Whiteside</td>
<td>CSW change request: Correct SOAP encoding</td>
<td>2006</td>
</tr>
<tr>
<td>OGC 06-094r1</td>
<td>A. Whiteside B. Schaeffer</td>
<td>OWS Common change request: Add SOAP encoding</td>
<td>2008</td>
</tr>
<tr>
<td>OGC 07-006r1</td>
<td>D. Nebert A. Whiteside P. Vretanos</td>
<td>OGC Catalogue Services Specification</td>
<td>2007</td>
</tr>
<tr>
<td>OGC 07-067r5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OGC 07-158</td>
<td>R. Gartmann B. Schäffer</td>
<td>Wrapping OGC HTTP-GET and -POST Services with SOAP</td>
<td>2008</td>
</tr>
<tr>
<td>ORCHESTRA IS-MDS</td>
<td>I. Iosifescu-Enescu M. Hugentobler</td>
<td>Implementation Specification of the OA Map and Diagram Service v.1.3/1.1</td>
<td>2007</td>
</tr>
<tr>
<td>ORCHESTRA D3.4.3</td>
<td>I. Iosifescu-Enescu</td>
<td>Specification of the Map and Diagram Service v.3.0/2.2</td>
<td>2007</td>
</tr>
<tr>
<td>Services</td>
<td>Authors</td>
<td>Description</td>
<td>Year</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>ORCHESTRA D3.4.4</td>
<td>M. Hugentobler</td>
<td>Specification of the Catalogue Service</td>
<td>2007</td>
</tr>
<tr>
<td>ORCH IS-MDS</td>
<td>Ioșifescu-Enescu M. Hugentobler</td>
<td>Implementation Specification of the OA Map and Diagram Service</td>
<td>2007</td>
</tr>
<tr>
<td>ORCH FAS SPEC</td>
<td>A. Friis-Christensen M. Lutz</td>
<td>Specification of the Feature Access Service</td>
<td>2006</td>
</tr>
<tr>
<td>PRO WSDL</td>
<td></td>
<td>Processing Service WSDL</td>
<td></td>
</tr>
<tr>
<td>RFC 2378</td>
<td>R. Hedberg P. Pomes</td>
<td>The CCSO Nameserver (Ph) Architecture</td>
<td>1998</td>
</tr>
<tr>
<td>RFC 2617</td>
<td>J. Franks et al.</td>
<td>HTTP Authentication: Basic and Digest Access Authentication</td>
<td>1999</td>
</tr>
<tr>
<td>RFC 3902</td>
<td>M. Baker M. Nottingham</td>
<td>The &quot;application/soap+xml&quot; media type</td>
<td>2004</td>
</tr>
<tr>
<td>SOAP-Attachments</td>
<td>H. F. Nielsen H. Ruellan</td>
<td>SOAP 1.2 Attachment Feature</td>
<td>2004</td>
</tr>
<tr>
<td>WSI-Basic</td>
<td>M. McIntosh et al.</td>
<td>Basic Security Profile Version 1.0</td>
<td>2007</td>
</tr>
</tbody>
</table>

Table 44 - Bibliography
Abstract
The goal of this document is to describe the status of SOAP HTTP binding of those services relevant for the Inspire Network Services, thus including OGC specifications (CSW, WMS, WFS, WPS, WCTS and WCS) and implementation (OWS3-5), as well as IST ORCHESTRA relevant services.
How to obtain EU publications

Our priced publications are available from EU Bookshop (http://bookshop.europa.eu), where you can place an order with the sales agent of your choice.

The Publications Office has a worldwide network of sales agents. You can obtain their contact details by sending a fax to (352) 29 29-42758.
The mission of the JRC is to provide customer-driven scientific and technical support for the conception, development, implementation and monitoring of EU policies. As a service of the European Commission, the JRC functions as a reference centre of science and technology for the Union. Close to the policy-making process, it serves the common interest of the Member States, while being independent of special interests, whether private or national.