This tutorial structure is based on the OGC WMS http://www.opengeospatial.org/resource/cookbooks and the response to the RFQ OGC http://www.opengeospatial.org/projects/initiatives/geoss/ogc. CREAF commitment to OGC is to provide a tutorial with a layout similar to that of the WMS cookbook which will be converted in a future to a more generic OGC cookbook.

GEOSS in practice - tutorial: from the enviroGRIDS project. Maybe useful for us: http://www.envirogrids.net/index.php?option=com_jdownloads&Itemid=13&view=view.download&catid=30&cid=54

Final destination: http://wiki.ieee-earth.org/Documents/GEOSS_Tutorials/GEOSS_Provider_Tutorials/Web_Feature_Service_Tutorial_for_GEOSS_Providers

CREAF and the University of Geneva (enviroGRIDS) are committed to the OGC to deliver a draft of this tutorial by the end of October.

Providing some related video support to this tutorial will be done soon

Title Page:
WFS tutorial
For data providers

Publisher Page (same for all tutorials, except for revision information)
- Published for GEOSS
- Found at the GEOSS BPW (comments are welcomed)
- Searched from the GWP (?)
- Revision information: v:0.1

Contributor Page (same for all tutorials, except for actual contributors)
- Gregory Giuliani, enviroGRIDS, gregory.giuliani@unepgrid.ch
- Joan Maso, CREAF, joan.maso@uab.es
- Anna Riverola, UAB, anna.reverola@uab.es

Table of Contents

1 GEOSS Introduction (same for all tutorials)

This tutorial is published as a documentation resource for the Global Earth Observation System of Systems (GEOSS). The purpose of this tutorial is to assist the GEOSS providers and GEOSS users in understanding the various ways in which GEOSS can be used to provision, discover, access, and use Earth observation data and services. The contents of this tutorial are provided, for the common good, under the Creative Commons Attribution 3.0 Unported License. This license is referenced at the bottom of the page, and applies to the entire tutorial. Briefly, the Creative Commons License allows the copying and sharing of this content, as long as proper attribution is made. Proper attribution shall include the list of authors (shown in the Tutorial Author table below) and the fact that the tutorial is for the benefit of the GEOSS. An example attribution is: "The GEOSS MyService Tutorial authored by Author 1, Author 2, etc., available from the GEOSS Best Practices Wiki.", where MyService is replaced by the actual name of the tutorial being reused or shared.

Any material that appears directly in any section of this tutorial that originated in a different work will be annotated with a reference identifier. This reference identifier can be found in the Bibliography section of the tutorial, along with the details of the reference. This bibliographic reference will serve as attribution for the material. It can be inferred from the existence of this attribution that the material being referenced and attributed is either in the public domain, offered through a license such as Creative Commons Zero or Creative Commons with Attribution, or has been used under explicit permission from the copyright owner of the material.

This tutorial can be found directly at the GEOSS Best Practices Wiki (BPW) (http://wiki.ieee-earth.org). It can also be found indirectly via a search at the Geo Web Portal (GWP)(http://www.geoportal.org). If registered at the BPW, and logged in, a reader of this tutorial can post comments on each page of the tutorial. These comments can serve to point out ways in which this tutorial can be improved upon.

1.1 Overview of GEOSS

The Global Earth Observation System of Systems will provide decision-support tools to a wide variety of users. As with the Internet, GEOSS will be a global and flexible network of content providers allowing decision makers to access an extraordinary range of information at their desk.
This ‘system of systems’ will proactively link together existing and planned observing

systems around the world and support the development of new systems where gaps currently exist. It will promote common technical standards so that data from the thousands of different instruments can be combined into coherent data sets. The ‘GEOPortal’ offers a single Internet access point for users seeking data, imagery and analytical software packages relevant to all parts of the globe. It connects users to existing data bases and portals and provides reliable, up-to-date and user friendly information – vital for the work of decision makers, planners and emergency managers. For users with limited or no access to the Internet, similar information is available via the ‘GEONETCast’ network of telecommunication satellites.
The Global Earth Observation System of Systems is simultaneously addressing nine areas of critical importance to people and society. It aims to empower the international community to protect itself against natural and human-induced disasters, understand the environmental sources of health hazards, manage energy resources, respond to climate change and its impacts, safeguard water resources, improve weather forecasts, manage ecosystems, promote sustainable agriculture and conserve biodiversity. GEOSS coordinates a multitude of complex and interrelated issues simultaneously. This cross-cutting approach avoids unnecessary duplication, encourages synergies between systems and ensures substantial economic, societal and environmental benefits.

1.2 Explanation of tutorials, in general

The AIP-4 tutorials aid data providers in deploying synchronous on-line access for their data. This activity was done in coordination with the Standards and Interoperability Forum (SIF), which managed the outline of tutorial topics. Data Access procedures, as recommended by the DSTF and GCI CT, were followed in AIP-4. An emphasis was placed on planning for multi-year persistence of the online data access and also on software, techniques and standards that make this possible.

2 Introduction to the topic

This chapter establishes the background information of web feature system interoperability, and the OpenGIS® WFS Standard.

How do the resources and the developed WFS service contribute to the GEOSS?
How to follow this tutorial in order to publish other Earth Observation data through WFS interface?
Things needed to be considered when providing robust WFS interface in an operation environment.
How to use the example data for GEOSS SBA application?

2.1 Discussion of the tutorial use

This chapter presents the technologies and the tools for WFS client and server development. WFS standard, version 2.0, is directly referenced to the corresponding "WFS requests" specification sections (GetCapabilities, DescribeFeature, GetFeature, etc) regarding the information needed when implementing the WFS interface.

Furthermore, this chapter also explains how to work with GML, and the concept of a GML application schema. It also shows you how to design your feature types by using UML class modeling, and how to convert it automatically to a GML application schema by using ShapeChange application (a java application that takes a UML model in XMI and converts it to a GML application schema).

This tutorial provides the necessary information for a data provider to publish its vector data though a WFS server. Some examples, illustrated with real data, are presented in a step by step description with open source and proprietary server technologies. Hopefully, these examples can be easily adapted to the provider data particularities. In addition, this tutorial presents some information regarding the WFS standard and its operation. This is illustrated with examples, which can be used by a technology developer as an easy way to have a first contact with a particular specification, before reading the hard details of the standard document itself. It is also thought to be used by teachers that want to present the WFS standard to their pupils using existing examples. A WFS user is supposed to access the data, to download the data, and to use visualization tools, some examples of these practices can also be seen here.

2.2 How interoperability is improved through the use of this server/service

2.2.1 Sharing features interoperability problems

A geographic feature is an abstraction of a real world phenomenon that is associated with a location relative to the Earth. A digital representation of the real world can be thought of as a set of features. A feature is a container for a set of values that characterizes that particular feature. Feature values are formalized in sets of *properties* where each property can be thought of as a {name, type, value} triplet.
Real world features are classified into *feature types* which can be characterized by a set of common properties. The number of properties a feature may have, together with their names and types, are determined by the *feature type* definition. Geographic features with geometry are those who have at least one geometric type property and one geometric value.

2.2.1.1 Sharing data model problem

Fortunately, many features of the real world can be classified into a single feature type and thus share the same set of property names and types. The process of choosing and defining feature types that build a feature collection are called data model definition.

Currently, almost each GIS software vendor has its own data encoding. In many GIS softwares, the definition of the data model starts by distributing the different feature types into different layers. Each layer can only contain a particular feature type with a particular kind of geometry (e.g. point, line or polygon) and defines a table of attributes that a feature can have. One of the most common formats that use this approach is the shape file, which can only contain a feature type with its associated geometric property in a binary file (.shp); the non geometric attributes will be contained in a table file (.dbf) as well as in a data indexing file (.shx). In this approach, the way features are going to be visualized (symbolized) is not included in the data model and it is left at the user discretion.
In this case, the data model is described by the number of layers, the use of points, lines, polygons etc, and by the data table field type's definition.

On the other hand, many CAD software vendors use another approach. Even though the feature types are also classified into layers, now the properties are represented by symbolization properties like colors, line width, line style, etc. Commonly, all feature types are stored in a single file (such as .dxf or .dgn).
In this case, the data model is described in a separated document where layering and symbolization are related to feature attribute types.

Other models are also possible, such as the geodatabase model where features are associated to records from tables from a database. Each feature type can be recorded in a different table and each property is stored with a date field. Spatial data can be both stored in a specific kind of data fields (geospatial fields) or as a sequence of coordinates in separate tables (in numerical fields).

In this heterogeneous situation, there is a need for a unified feature type definition and encoding. There is also the need for a data service that could inform and communicate the feature type's definition. The WFS services define the DescribeFeature operation as a way to request feature types, and these can be shared using GML application schemas.

2.2.1.2 Sharing feature collection problem

A feature collection is a sequence of different types of features. In the scenarios previously described, feature collections can be expressed in different data formats by different software vendors. Fortunately, features can be stored in files which in turn can be redistributed by web accessible folders or by ftp. For example, the Circum-Arctic Map of Permafrost and Ground-Ice Conditions (http://nsidc.org/data/ggd318.html) can be accessed by ftp via ftp://sidads.colorado.edu/pub/DATASETS/fgdc/ggd318_map_circumarctic in shapefiles.

Nevertheless, this approach does not offer the user the possibility to control the data format used, and it is completely up to the producers who offer their own favorite format and in some cases other popular formats. This usually leads to the coexistence of several different incompatible data formats. Furthermore, some geospatial information can generate large files which can be difficult to transmit. Therefore, to be able to extract only a subset of elements from the original file would be an asset.

In this heterogeneous situation, it is detected that a common way to request a subset of a feature data from some feature types is needed. Therefore, the WFS services define the GetFeature operation as a way to request subsets of feature collections, being the features shared in GML encoded files.

2.2.1.3 Editing a feature collection problem

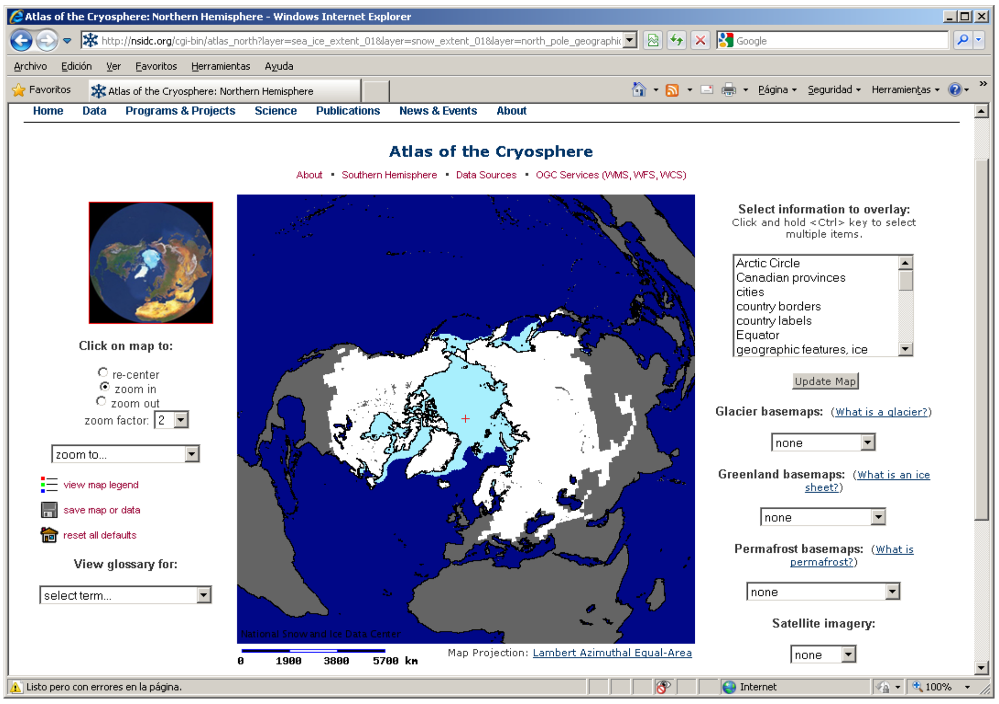
Feature collections aren't static. Sometimes, the representation of a feature is not accurate enough, and later, a better representation becomes available; or sometimes the real world changes and their feature representations have to be updated accordingly. In our previous example, where data was held in web accessible folders or by an ftp file, the simplest approach would be to update the dataset and replace the whole file with the updated version. This is only possible when the changes are made in a centralized way, where there is only one person responsible for the management of the dataset to be updated. In other cases, different independent actors maintain a single dataset from different places. In the worse scenario, each actor could use a different format and a different set of tools to do it.

When a decentralized and heterogeneous situation occurs, the need for a common way to sent updates of some features, while keeping the majority of them intact is of great importance. The WFS services define the Transaction operation in order to create, update or delete some features in a feature collection; it also uses GML encoded files to communicate which features are being affected by the operation.

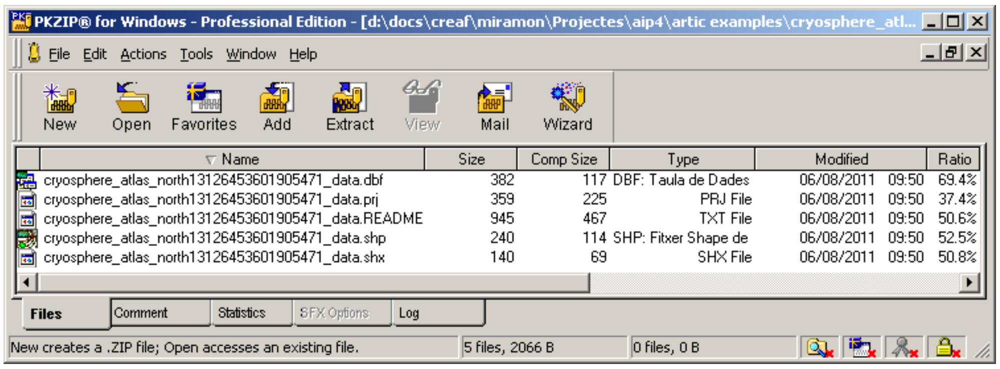
2.2.2 Data modeling needs and GML

2.2.2.1 Formal definition of a data model is possible only when the changes are made in a centralized way

The main format used in WFS is GML. GML requires a clear definition of the data model, as previously mentioned, it specifies the feature types and their properties. In this section of the tutorial we are going to use a very simple layer defining the position of 4 different pole locations that we have extracted from the "Atlas of the Cryosphere" web site (CSR record: https://geossregistries.info/geosspub/component_details_ns.jsp?compId=urn:uuid:07fda5dd-00a9-4c25-9668-dc598ca73fd7). The portal has the functionality of saving the data for a later use, and thus one can retrieve the desired layer in Shapefile or in GML format.


Atlas of the Cryosphere: Northern Hemisphere - Windows Internet Explorer
http://nsidc.org/cgi-bin/atlas_north?layer=sea_ice_extent_01&layer=snow_extent_01&layer=north_pole_geographic
Archivo Edición Ver Favoritos Herramientas Ayuda
Favoritos
Atlas of the Cryosphere: Northern Hemisphere
Home Data Programs & Projects Science Publications News & Events About
Atlas of the Cryosphere
About • Southern Hemisphere • Data Sources • OGC Services (WMS, WFS, WCS)
Click on map to:
re-center
zoom in
zoom out
zoom factor: 2
zoom to...
view map legend
save map or data
reset all defaults
View glossary for:
select term...
Select information to overlay:
Click and hold <Ctrl> key to select multiple items.
Arctic Circle
Canadian provinces
cities
country borders
country labels
Equator
geographic features, ice
Update Map
Glacier basemaps: (What is a glacier?)
none
Greenland basemaps: (What is an ice sheet?)
none
Permafrost basemaps: (What is permafrost?)
none
Satellite imagery:
none
National Snow and Ice Data Center
0 1900 3800 5700 km
Map Projection: Lambert Azimuthal Equal-Area
Listo pero con errores en la página.
Internet
100%

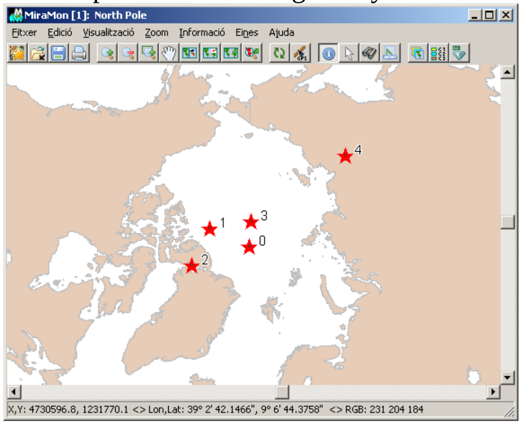
The shapefile format is downloaded in a ZIP file. The ZIP file contains 5 files.


PKZIP® for Windows - Professional Edition - [d:\docs\creaf\miramon\Projectes\aip4\artic examples\cryosphere_atl...
File Edit Actions Tools Window Help
New
Open
Favorites
Add
Extract
View
Mail
Wizard
Files
Comment
Statistics
SFX Options
Log

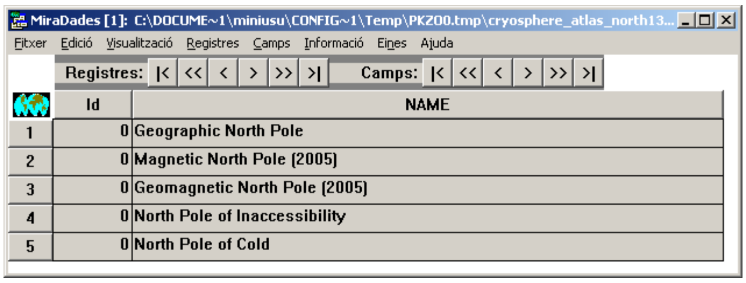
Name	Size	Comp Size	Type	Modified	Ratio
cryosphere_atlas_north1312645360190547 1_data.dbf	382	117	DBF: Taula de Dades	06/08/2011 09:50	69.4%
cryosphere_atlas_north1312645360190547 1_data.prj	359	225	PRJ File	06/08/2011 09:50	37.4%
cryosphere_atlas_north1312645360190547 1_data.README	945	467	TXT File	06/08/2011 09:50	50.6%
cryosphere_atlas_north1312645360190547 1_data.shp	240	114	SHP: Fitxer Shape de	06/08/2011 09:50	52.5%
cryosphere_atlas_north1312645360190547 1_data.shx	140	69	SHX File	06/08/2011 09:50	50.8%

New creates a .ZIP file; Open accesses an existing file. | 5 files, 2066 B | 0 files, 0 B

As a classical GIS file, a shapefile is geospatial-centric which means that entities are encoded as point files whose positions are being binary encoded in a .shp file.


MiraMon [1]: North Pole
Fitxer Edició Visualització Zoom Informació Eines Ajuda
4
1
3
0
2
X,Y: 4730596.8, 1231770.1 <> Lon,Lat: 39° 2' 42.1466", 9° 6' 44.3758" <> RGB: 231 204 184

These points are linked to a separate table file that contains alphanumeric attributes in a .dbf file.


MiraDades [1]: C:\DOCUME~1\miniusu\CONFIG~1\Temp\PKZ00.tmp\cryosphere_atlas_north13...
Fitxer Edició Visualització Registres Camps Informació Eines Ajuda
Registres: |< << < > >> >|
Camps: |< << < > >> >|

	Id	NAME
1	0	Geographic North Pole
2	0	Magnetic North Pole (2005)
3	0	Geomagnetic North Pole (2005)
4	0	North Pole of Inaccessibility
5	0	North Pole of Cold

The shapefile is implicitly defining a data model. In this example, there is only one feature type, let’s call it PoleLocationType. This type has 3 properties: one geometric property and two thematic properties. The geometric one is expressed by a point coordinate, and the thematic ones are represented as a number (Id) and as a name (description).

A GML file is a text file that can be seen in an XML editor (such as XMLSpy). This tutorial assumes that you are familiar with XML; if this is not the case, we recommend you to get started with the course available in the W3C school tutorial at http://www.w3schools.com/xml/default.asp.

This is how the position of the Geographic North Pole is expressed in GML.

```
<ogr:PoleLocation fid="F0">
    <ogr:geometryProperty>
        <gml:Point>
            <gml:coordinates>0,0</gml:coordinates>
        </gml:Point>
    </ogr:geometryProperty>
    <ogr:id>0</ogr:id>
    <ogr:name>Geographic North Pole</ogr:name>
</ogr: PoleLocation >
```

In GML format, objects are represented in upper camel case (all words start by a capital letter) and properties in lower camel case (all words start by a capital letter except the first one). Objects are composed by property elements, which in turn can be described by objects. So that, GML alternates objects and properties and this is clearly identifiable by the alternated use of upper camel cases and lower camel cases (the original GML file does not follow this notation and thus, it has been altered in this example).

A very common way of encoding a complete GML file is to pack features into feature collections. A feature collection is a list of features. This XML fragment shows the whole feature collection.

```
<ogr:FeatureCollection>
    <gml:featureMember>
        <ogr:PoleLocation fid="F0">
            […]
            <ogr:name>Geographic North Pole</ogr:name>
        </ogr:PoleLocation>
    </gml:featureMember>
    <gml:featureMember>
        <ogr:PoleLocation fid="F1">
            […]
        <ogr:name>Magnetic North Pole (2005)</ogr:name>
        </ogr:PoleLocation>
    </gml:featureMember>
    <gml:featureMember>
        <ogr:PoleLocation fid="F2">
            […]
        <ogr:name> Geomagnetic North Pole (2005)</ogr:name>
        </ogr:PoleLocation>
    </gml:featureMember>
    <gml:featureMember>
        <ogr:PoleLocation fid="F3">
            […]
        <ogr:name>North Pole of Inaccessibility</ogr:name>
        </ogr:PoleLocation>
    </gml:featureMember>
    <gml:featureMember>
```

```
            <ogr:PoleLocation fid="F4">
                [...]
            <ogr:name> North Pole of Cold </ogr:name>
            </ogr:PoleLocation>
      </gml:featureMember>
</ogr:FeatureCollection>
```

XML files can be validated to a XML schema (XSD) that ensures that a particular XML file only contains data types that are defined in the XML schema. GML uses this mechanism to define the data model in a GML application schema and thus, forcing GML files to strictly follow the expected data model.

2.2.2.2 Practical definition in GML application schemas

Because of what it has just been explained, you must define your own data model as a GML application schema (or reuse a previously created data model) and connect your GML file to it. By creating your GML application schema, you also create you namespace. Please do not be scared by this last word; a namespace is only the group of elements and types that define your model. You will also need other preexisting namespaces like the GML namespace and the XML Schema instance namespace. It is now when things can get complicated. A namespace receives 3 different names in an XML file:

- A short name of typically 2 or 3 characters, which is used to qualify elements in XML.
- A long name in the form of a URI. This is a thing that resembles an http URL, but it is not. Since it is only a long name that uses the URL structure to be sure that it is unique in the whole world.
- The real URL position of the XSD file (or files) that defines the namespace.

To be practical, in the XML file, you could define a default namespace that has no short name and is generally assigned to the most used namespace (this is not the case in the following example).

The root elements of a typical GML file is FeatureCollection. To connect these elements to the right namespaces, you must use the attributes of this element:

```
<ogr:FeatureCollection
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:gml="http://www.opengis.net/gml"
            xmlns:ogr="http://ogr.maptools.org/"
            xsi:schemaLocation="http://ogr.maptools.org/
                        cryosphere_atlas_north_pole_location.xsd">
```

In this GML file you have 3 namespaces:

short	long name (URI)	URL
xsi	http://www.w3.org/2001/XMLSchema-instance	not needed
gml	http://www.opengis.net/gml	still unknown
ogr	http://ogr.maptools.org/	cryosphere_atlas_north_pole_location.xsd

Our own namespace is called "ogr" as the short name and http://ogr.maptools.org/ as the long name, which can be found at:

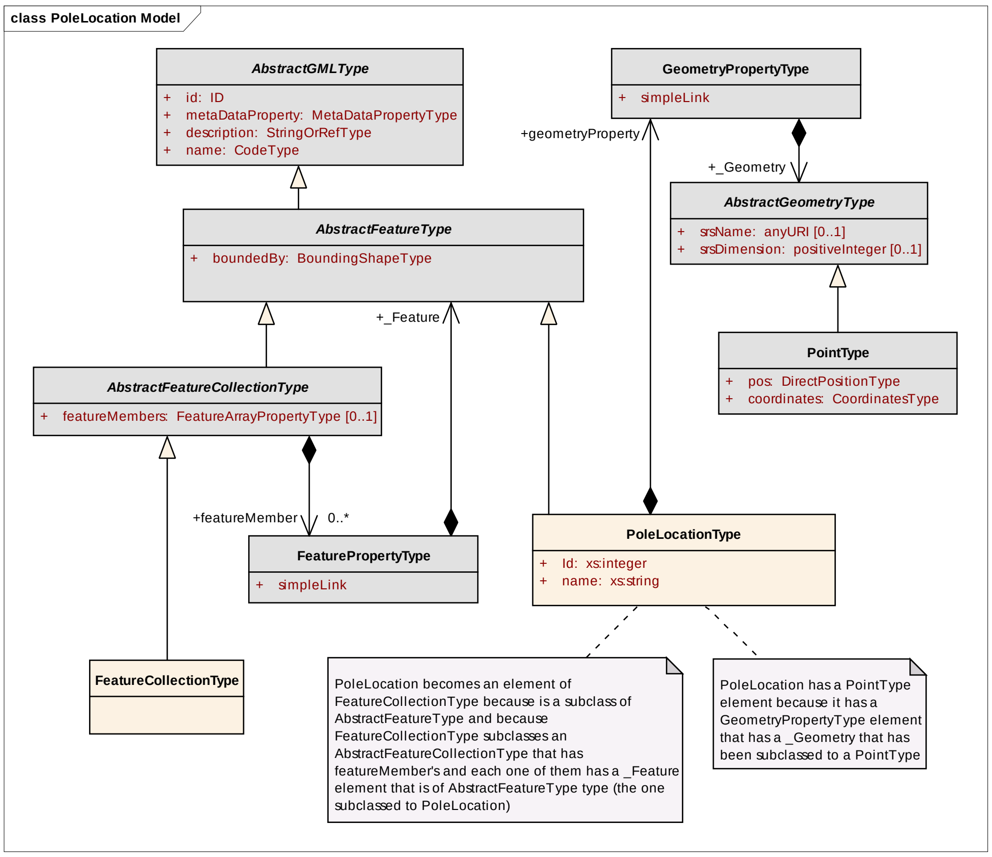
cryosphere_atlas_north_pole_location.xsd. This is the file that will define your namespace. The root element of your schema defines that the taretNamespace (the namespace that will be defined) is http://ogr.maptools.org/. The first child element in root is the "import" element that sets the location for the gml namespace to a real URL: http://schemas.opengis.net/gml/2.1.2.1/feature.xsd.

```
<xs:schema xmlns:ogr="http://ogr.maptools.org/"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:gml="http://www.opengis.net/gml"
        targetNamespace="http://ogr.maptools.org/"
        elementFormDefault="qualified" version="1.0">
        <xs:import namespace="http://www.opengis.net/gml"
                schemaLocation="http://schemas.opengis.net/gml/2.1.2.1/feature.xsd"/>
```

Formatat: alemany (Alemanya)

Now we are ready to define the Atlas of the Cryosphere North Pole Location data model. To do that we only have to define 2 data types: one data type for a FeatureCollection and another for features (these are called PoleLocations). This could be done in a simple way by using XML schema types, but it is not recommended, because we have to ensure both data types are derived from GML types, so instead we can use the geometric types that GML provides (particularly, we need gml:Point in our example).

The following UML class model illustrates how FeatureCollection (from the FeatureCollectionType) and PoleLocation (from PoleLocationType) are derived from GML types (which in turn are also derived from other more primitive GML types). GML types are represented in grey in the UML model illustrations.



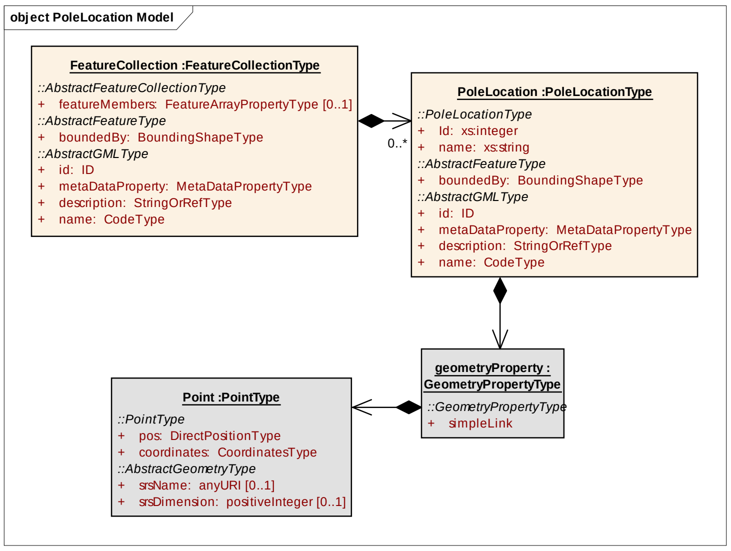
PoleLocation becomes an element of FeatureCollectionType because is a subclass of AbstractFeatureType and because FeatureCollectionType subclasses an AbstractFeatureCollectionType that has featureMember's and each one of them has a _Feature element that is of AbstractFeatureType type (the one subclassed to PoleLocation)

PoleLocation has a PointType element because it has a GeometryPropertyType element that has a _Geometry that has been subclassed to a PointType

This figure shows the inheritance model of the data types (classes and subclasses)

A FeatureCollectionType is a subclass of AbstractFeatureCollectionType which is a subclass of AbstractFeatureType that in turn is a subclass of AbstractGMLType. In practice this means that a FeatureCollection inherits all elements and attributes of the previous ones.
PoleLocation becomes an element of FeatureCollection because it is a subclass of AbstractFeatureType, and a FeatureCollection is composed by a _Feature element or some element subclass of AbstractFeatureType type (such as PoleLocation).
PoleLocation has a Point element because it has a GeometryPropertyType element that has a _Geometry that has been subclassed to a PointType.

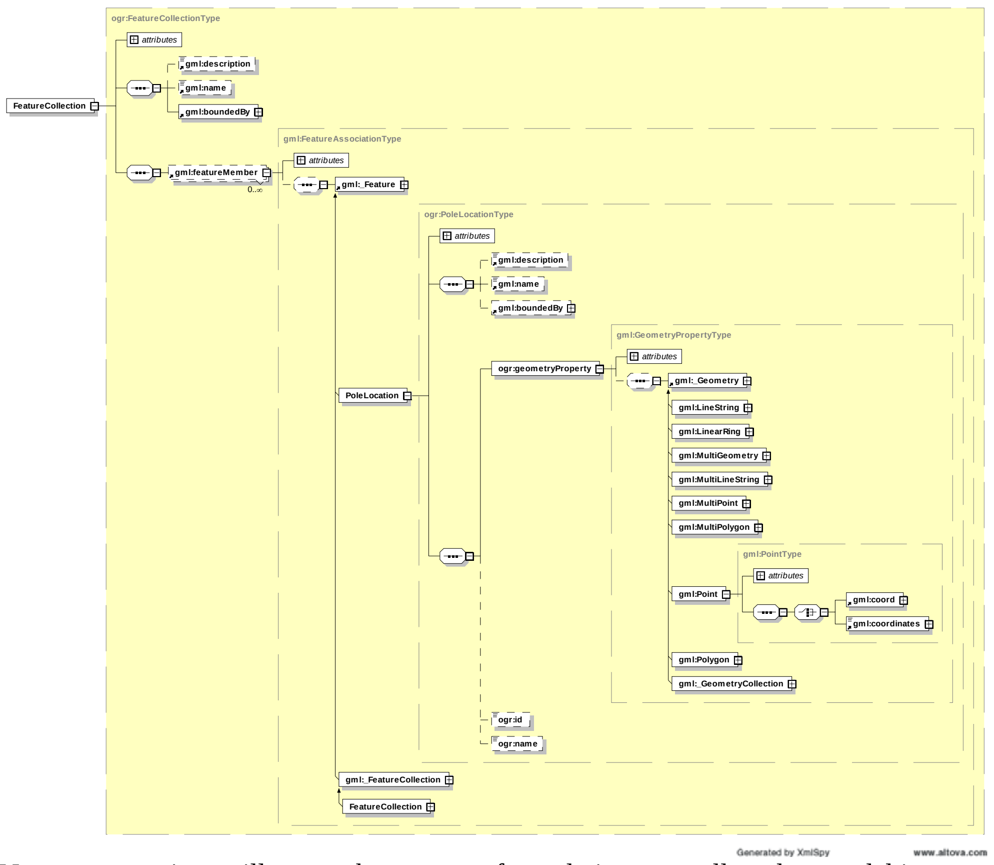
Fortunately, this inheritance mechanism happens automatically in XML, so the UML object model shows all attributes available for the 4 objects that are used in the definition of the feature collection.


object PoleLocation Model
FeatureCollection :FeatureCollectionType
::AbstractFeatureCollectionType
+ featureMembers: FeatureArrayPropertyType [0..1]
::AbstractFeatureType
+ boundedBy: BoundingShapeType
::AbstractGMLType
+ id: ID
+ metaDataProperty: MetaDataPropertyType
+ description: StringOrRefType
+ name: CodeType
0..*
PoleLocation :PoleLocationType
::PoleLocationType
+ Id: xs:integer
+ name: xs:string
::AbstractFeatureType
+ boundedBy: BoundingShapeType
::AbstractGMLType
+ id: ID
+ metaDataProperty: MetaDataPropertyType
+ description: StringOrRefType
+ name: CodeType
geometryProperty : GeometryPropertyType
::GeometryPropertyType
+ simpleLink
Point :PointType
::PointType
+ pos: DirectPositionType
+ coordinates: CoordinatesType
::AbstractGeometryType
+ srsName: anyURI [0..1]
+ srsDimension: positiveInteger [0..1]

This figure represents the feature instances of the types specified, as well as all the properties available from the different class dependencies nested.

The UML models presented here have been elaborated to illustrate the GML class dependencies and inheritance model. Latter we will present a simplified version of this model that is based on ISO types and on the way that this model can be automatically translated into XML.

XMLSpy offers us another view of the same model directly extracted from the XML Schema, so in this model more possibilities can be seen which are not shown in the UML diagram, although the main idea remains the same.



Now we are going to illustrate the process of translating manually a data model into a GML application schema. In order to do that, we have to do 4 things:

- Declare the root element FeatureCollection
- Define a type for the FeatureCollection
- Declare a PoleLocation element for our features
- Define a type for the PoleLocation

To declare the root element FeatureCollection is done by creating an element called substitutionGroup (subclass of) gml:_FeatureCollection.

```
<xs:element name="FeatureCollection" type="ogr:FeatureCollectionType"
            substitutionGroup="gml:_FeatureCollection"/>
```

The definition of a type for the FeatureCollection is done by extending gml:AbstractFeatureCollectionType. It adds to it a two additional attribute lockId and scope.

```
<xs:complexType name="FeatureCollectionType">
    <xs:complexContent>
        <xs:extension base="gml:AbstractFeatureCollectionType">
            <xs:attribute name="lockId" type="xs:string" use="optional"/>
            <xs:attribute name="scope" type="xs:string" use="optional"/>
        </xs:extension>
    </xs:complexContent>
    </xs:complexType>
```

The declaration of a PoleLocation element for our features is done by creating an element that is substitutionGroup (subclass of) gml:_Feature.

```
    <xs:element name="PoleLocation" type="ogr:PoleLocationType"
                substitutionGroup="gml:_Feature"/>
```

To define a type for the PoleLocation is an essential step because it is going to define how our features will be (defines our feature types). It is done by defining a new type that extents gml:AbstractFeatureType.

```
    <xs:complexType name="PoleLocationType">
        <xs:complexContent>
            <xs:extension base="gml:AbstractFeatureType">
                <xs:sequence>
                    […]
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

The sequence of extended elements (and eventually attributes) define the properties of features (the only kind in this small example)

The first element added is a geometry property that uses a generic geometry property type: gml:GeometryPropertyType. This trick makes possible to use any of the generalized types of gml:GeometryPropertyType and particularly a gml:PointType (the one we are using). It is of best practice to directly use a type that adjusts best to our data model (i.e. gml:PointType in our case), but it is of common practice to use a more generic one such as gml:GeometryPropertyType

```
                    <xs:element name="geometryProperty"
type="gml:GeometryPropertyType" nillable="true"/>
```

Following the geometric element, we have two elements that are not GML types but common XML types (i.e. xs:integer and xs:string) restricted to follow convenient limitations:

```
                    <xs:element name="id" nillable="true" minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:integer">
                                        <xs:totalDigits value="6"/>
                                </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="name"nillable="true" minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:maxLength value="50"/>
```

```
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
```

2.2.2.3 GML application schemas from UML models

Many GML application schemas are elaborated directly in a XML editor like XMLSpy, but this requires the user to be XML proficiency. Another approach is to design them using ISO 19103 conformant UML model, and then the GML application schema can be created by following the rules given in the GML standard (Annex E of ISO 19136). In deed, conversion from simple UML models to GML application schemas is an important step to simplify the GML application schema creation. Some tools have automated the migration from UML to GML application schema, so one no longer has to worry about XML schemas. Fullmoon (https://www.seegrid.csiro.au/wiki/AppSchemas/ApplicationSchemaDesign and http://projects.arcs.org.au/trac/fullmoon/wiki/FullMoon) and ShapeChange (http://www.interactive-instruments.de/ugas/) are two open source software solutions that partially automate the process.
FullMoon provides a more complete environment, but the process of preparing the environment is quite complicated (even if you use the setup wizard, you have to follow several additional steps), so it is only recommended for complex projects maintained by specialists (www.fig.net/pub/fig2011/papers/ts03i/ts03i_kuczynska_chojka_5206.pdf).
ShapeChange is a Java based command line tool that has a manual setup process, which is not simple either, although it has a more limited number and clearer steps.

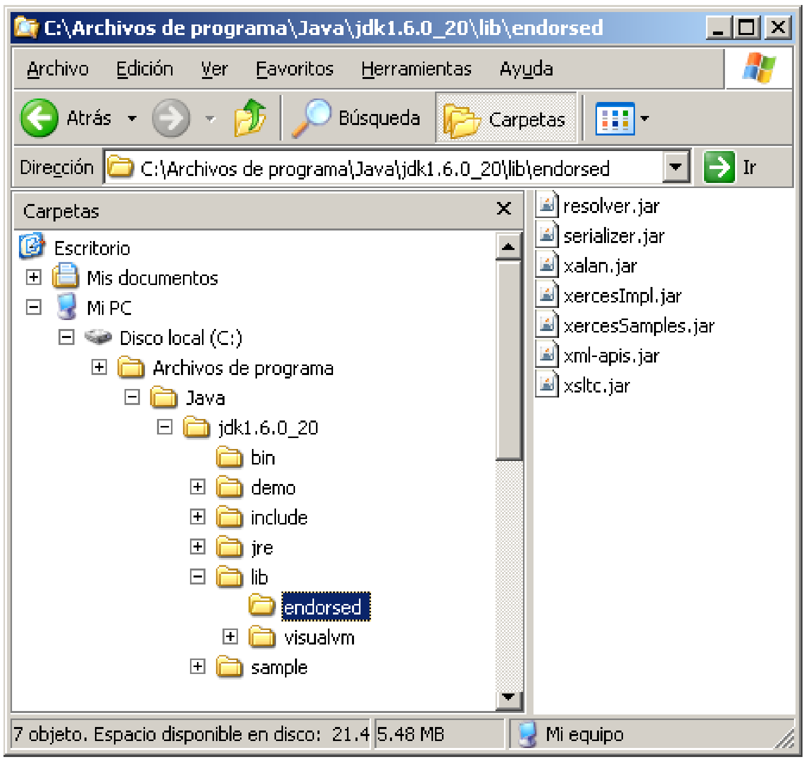
We are going to illustrate the use of ShapeChange v1.0rc to generate GML application schemas from UML models.

2.2.2.3.1 ShapeChange setup

Shapechange requires Java, Xerces and Xalan.

- Java JDK can be obtained from http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Xerces 2 for java can be obtained from http://xerces.apache.org/xerces2-j/
- Xalan for java http://xml.apache.org/xalan-j/

In this example we have downloaded: jdk-6u20-windows-i586.exe, Xerces-J-bin.2.11.0.zip and xalan-j_2_7_1-bin-2jars.zip. Once you setup the java JDK using the setup wizard, you have to uncompress Xerces-J-bin.2.11.0.zip and xalan-j_2_7_1-bin-2jars.zip and only copy the jar files to a new "endorsed" folder in your java "lib" folder (by default: "c:\Program Files\Java\jdk1.6.0_20\lib\endorsed" (some files are already duplicated in both zip files, and you just have to keep the newer one).



Dirección: C:\Archivos de programa\Java\jdk1.6.0_20\lib\endorsed

```
resolver.jar
serializer.jar
xalan.jar
xercesImpl.jar
xercesSamples.jar
xml-apis.jar
xsltc.jar
```

7 objeto. Espacio disponible en disco: 21.4 | 5.48 MB | Mi equipo

Then, you can proceed to install the ShapeChange tool by downloading it from http://www.interactive-instruments.de/fileadmin/gdi/dwnlds/ugas/ShapeChange-v1.0rc.zip and decompress it in a folder.

ShapeChange is prepared to generate GML 3.2 (ISO version) by default. In this example, we want to be compatible with WFS 1.1 and GML 3.1.1. These will require small modifications in the ShapeChange configuration. ShapeChange main configuration is stored in the ShapeChangeConfiguration.xml file from the "ShapeChange-1.0\config" folder. Actually, the file only contains pointers to three files that store respectively, aliases for UML stereotypes, URI namespaces and transformation mappings from UML attributes and types to GML elements. This file can be duplicated to support alternative configurations. In our case, we have cloned it to ShapeChangeConfigurationGML311.xml. Then we have edited it to point to two new configurations files:

```
<?xml version="1.0" encoding="UTF-8"?>
<ShapeChangeConfiguration xmlns:xi="http://www.w3.org/2001/XInclude"
xmlns="http://www.interactive-instruments.de/ShapeChange/Configuration"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.interactive-
instruments.de/ShapeChange/Configuration ShapeChangeConfiguration.xsd">
        <xi:include href="StandardAliases.xml"/>
        <xi:include href="StandardNamespacesGML311.xml"/>
        <xi:include href="StandardMapEntriesUML.xml"/>
</ShapeChangeConfiguration>
```

StandardNamespacesGML311.xml was created by cloning StandardNamespaces.xml and editing the GML namespace to point to the 3.1.1.

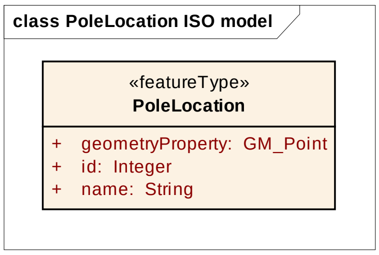
```
<Namespace nsabr="gml" ns="http://www.opengis.net/gml"
location="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd" />
```

Also, StandardMapEntriesUML.xml was created by cloning StandardMapEntries.xml and introducing a new line for convenience.

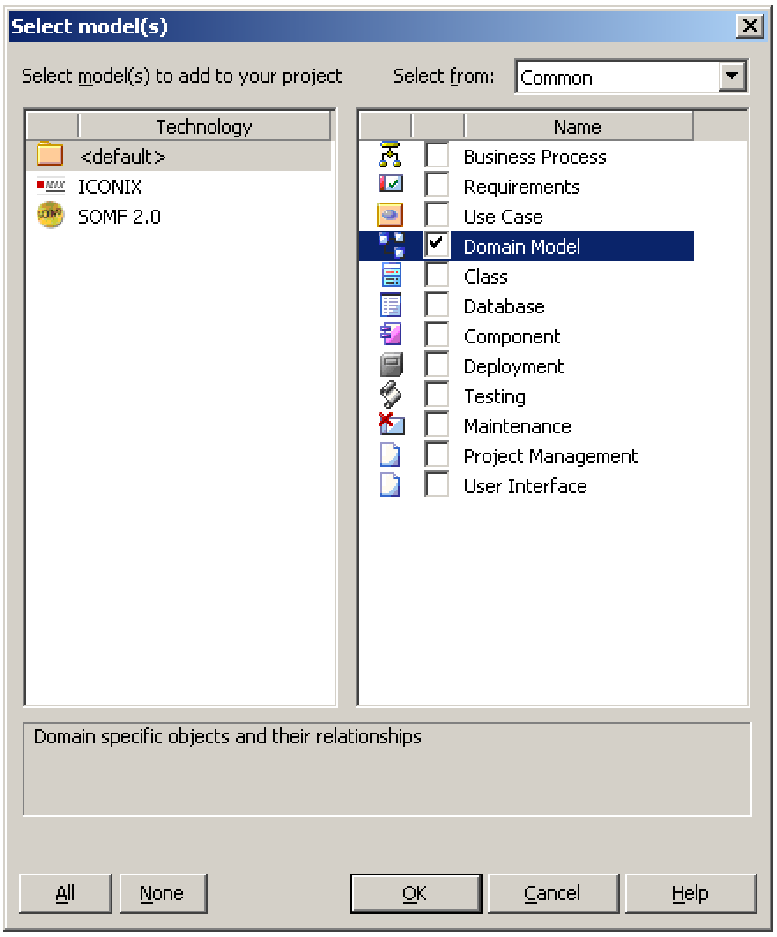
```
<MapEntry type="String" xsdEncodingRules="iso19136_2007
iso19136_2007_ShapeChange_1.0_Extensions" xmlPropertyType="string"
xmlType="string" />
```

2.2.2.3.2 UML model edition

We are going to create our UML model in Enterprise Architect v8.0. The UML model for our pole location example will look very simple, since we have only one feature type called PoleLocation with three properties.



In Enterprise Architect v8.0 we will create a new project and generate a Domain Model.



Select model(s)

Select model(s) to add to your project

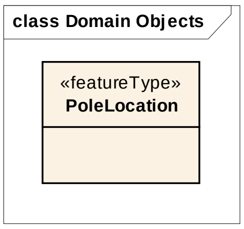
Select from: Common

Technology
<default>
ICONIX
SOMF 2.0

	Name
[ ]	Business Process
[ ]	Requirements
[ ]	Use Case
[✓]	Domain Model
[ ]	Class
[ ]	Database
[ ]	Component
[ ]	Deployment
[ ]	Testing
[ ]	Maintenance
[ ]	Project Management
[ ]	User Interface

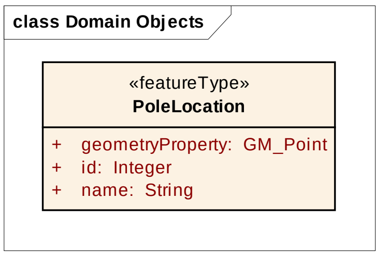
Domain specific objects and their relationships

We will remove Class1, Class2 and Class3 classes that EA creates automatically as examples, and then we will open the Domain Object diagram. We will create a new class by dragging and dropping a class from the toolbox in the Domain Object diagram. This class will be named PoleLocation and will have a "featureType" stereotype (featureType is not on the stereotypes drop down list, and it needs to be typed in). As a language we will select <none> and accept the dialog box.

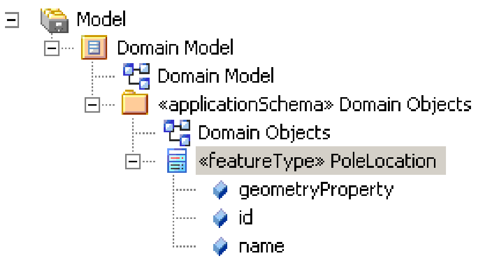


Now we are going to add attributes to the class: The first one will be named geometryProperty and will have the GM_Point type (this will be a public scope attribute). ShapeChange uses the geographical feature datatypes defined in ISO 19107 and GM_Point is an ISO datatype for a point. You can find some of these type names in the freely available ISO_TC _211_Standards_Guide.pdf document, specifically, in Figure 4 _Geometry basic classes with specialization relations, as well as in the GML

3.1.1 specification, illustrated in Figure 13. You can also see the comprehensive list of supported geospatial ISO data types and their corresponding GML types in the StandardMapEntriesUML.xml file. The second attribute will be named "id" and it will be an Integer type. Finally, the third attribute will be named "name" and it will be a String type (theses types are on the Type drop down list). In the class diagram you will see:



At this point we are almost done; now, in order for the ShapeChange tool to work you just need to specify which package in the project is the application schema package (this step has to be done even if you have only one package). To do that, you have to mark this package as an "applicationSchema" stereotype.

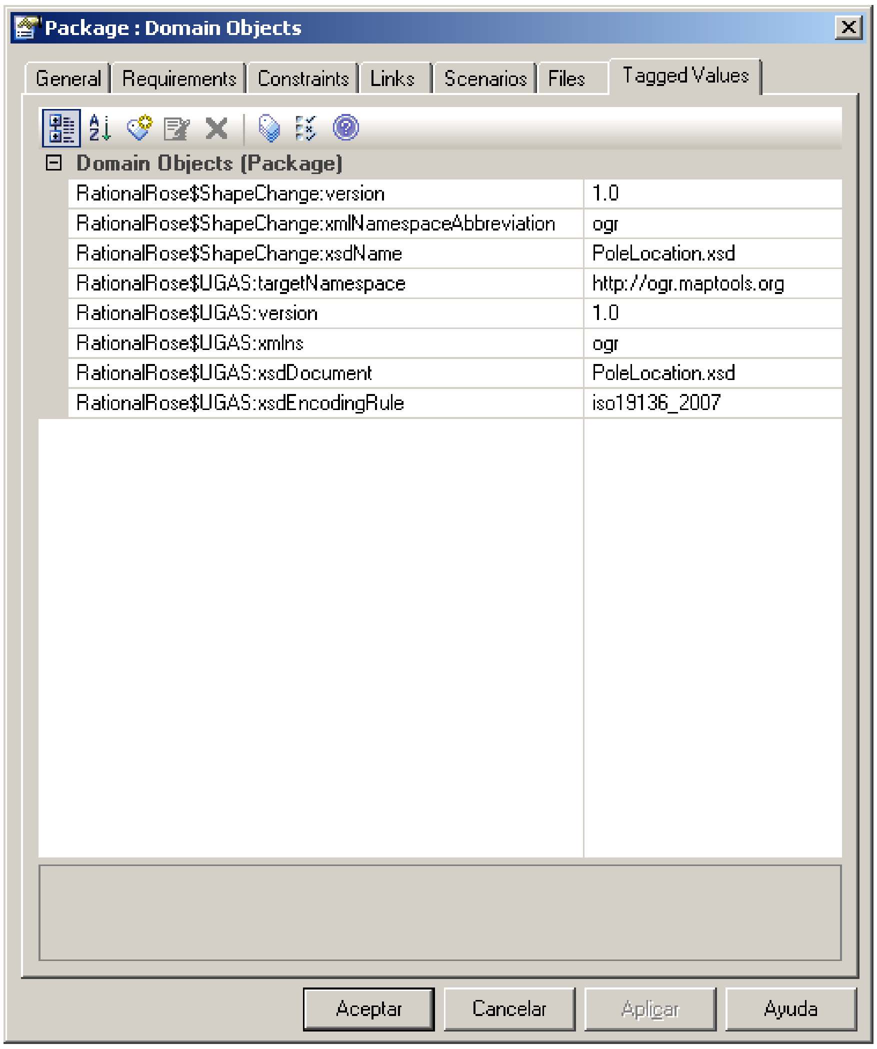


Note that now a useful trick needs to be done for the ShapeChange to communicate with the XML abbreviated namespace, namespace URI and XSD filename. This is done by adding to the applicationSchema package the following tagged values:

RationalRose$ShapeChange:version	1.0
RationalRose$ShapeChange:xsdName	PoleLocation.xsd
RationalRose$ShapeChange:xmlNamespaceAbbreviation	ogr
RationalRose$UGAS:version	1.0
RationalRose$UGAS:xmlns	ogr
RationalRose$UGAS:xsdDocument	PoleLocation.xsd
RationalRose$UGAS:targetNamespace	http://ogr.maptools.org
RationalRose$UGAS:xsdEncodingRule	iso19136_2007

This is an important step (if you miss it, the ShapeChange will not produce the result) , which is not explained in the installing documentation that comes with the program. In a project with more than one package, it will result in more than one file. The RationalRose$ShapeChange tagged values have to be documented in the package that will become the main applicationSchema, and the RationalRose$UGAS values have to be included in all packages.

In this example, we are using the same values as the one in pole location original example.

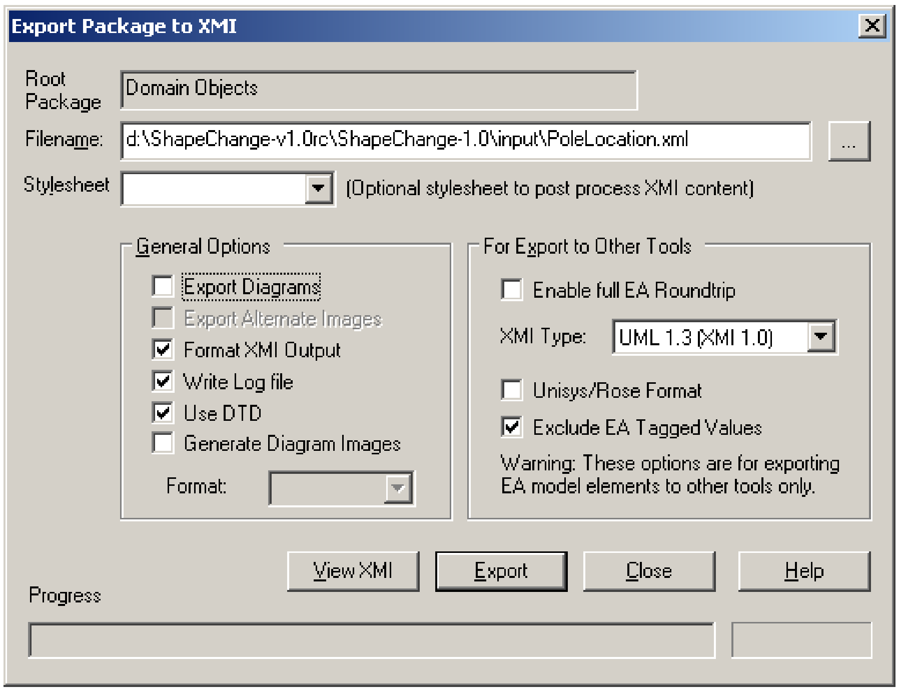


Domain Objects (Package)

RationalRose$ShapeChange:version	1.0
RationalRose$ShapeChange:xmlNamespaceAbbreviation	ogr
RationalRose$ShapeChange:xsdName	PoleLocation.xsd
RationalRose$UGAS:targetNamespace	http://ogr.maptools.org
RationalRose$UGAS:version	1.0
RationalRose$UGAS:xmlns	ogr
RationalRose$UGAS:xsdDocument	PoleLocation.xsd
RationalRose$UGAS:xsdEncodingRule	iso19136_2007

2.2.2.3.3 Automatically application schema generated by ShapeChange.

Let’s now explain how to export the UML model fromUML 1.3 (XMI 1.0) to an xml file (Project|ImportExport|Export Package to XM)



Export Package to XMI

Root Package: Domain Objects

Filename: d:\ShapeChange-v1.0rc\ShapeChange-1.0\input\PoleLocation.xml

Stylesheet: (Optional stylesheet to post process XMI content)

General Options
- [ ] Export Diagrams
- [ ] Export Alternate Images
- [✓] Format XMI Output
- [✓] Write Log file
- [✓] Use DTD
- [ ] Generate Diagram Images

Format:

For Export to Other Tools
- [ ] Enable full EA Roundtrip

XMI Type: UML 1.3 (XMI 1.0)

- [ ] Unisys/Rose Format
- [✓] Exclude EA Tagged Values

Warning: These options are for exporting EA model elements to other tools only.

We will export the "PoleLocationXMI.xml" xmi file into the "input" directory of the ShapeChange setup.

Now we are going to execute the ShapeChange from the command line and from the "input" directory. Simply by typing:

```
"C:\Archivos de programa\Java\jdk1.6.0_20\bin\java"
-Djava.endorsed.dirs="C:\Archivos de programa\Java\jdk1.6.0_20\lib\endorsed"
-Xms128m -Xmx1024m
-jar "../bin/ShapeChange.jar"
-o "../result" -s -P -v 3.1 -D NONE -r DEBUG
-c "../config/ShapeChangeConfigurationGML311.xml"
-R "iso19136_2007_ShapeChange_1.0_Extensions" PoleLocationXMI.xml
```

The result will be stored in the "result" directory.

Using "-r DEBUG" will result in reporting the maximum level of the process information:

```
D Added tagged value 'version' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: 1.0.
D Added tagged value 'version' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: 1.0.
D Added tagged value 'xsdDocument' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: PoleLocation.xsd.
D Added tagged value 'xmlns' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: ogr.
```

```
D Added tagged value 'version' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: 1.0.
D Added tagged value 'xmlns' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: ogr.
D Added tagged value 'xsdDocument' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: PoleLocation.xsd.
D Added tagged value 'targetNamespace' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value:
http://ogr.maptools.org.
D Added tagged value 'xsdEncodingRule' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678' with value: iso19136_2007.
D Added tagged value 'version' for element with ID
'EAID_15E40ACC_24C0_4c83_9380_81C311E8B73A' with value: 1.0.
D Added stereotype 'Application Schema' for element with ID
'EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678'.
D Application schema found, package name: Domain Objects
D Added stereotype 'featureType' for element with ID
'EAID_15E40ACC_24C0_4c83_9380_81C311E8B73A'.
D The package with ID MX_EAID_C0C177F7_A154_44b4_BD1A_F3A9999E3678
and name 'EA
Model' was created. Namespace: <null>.
D The package with ID EAPK_C0C177F7_A154_44b4_BD1A_F3A9999E3678 and
name 'Domain
 Objects' was created. Namespace: http://ogr.maptools.org.
D Processing Classes...
D The class with ID EAID_11111111_5487_4080_A7F4_41526CB0AA00 and name
'EARootCl
ass' was created.
D The property with ID eaxmiid2 and name 'geometricProperty' was created.
D The property with ID eaxmiid4 and name 'id' was created.
D The property with ID eaxmiid6 and name 'name' was created.
D The class with ID EAID_15E40ACC_24C0_4c83_9380_81C311E8B73A and name
'PoleLoca
tion' was created.
D The class with ID eaxmiid5 and name 'Integer' was created.
D The class with ID eaxmiid7 and name 'String' was created.
D The class with ID eaxmiid3 and name 'GM_Point' was created.
D Fixing Unknowns...
D Checking overloading. Class = Integer; current class = Integer.
D Checking overloading. Class = PoleLocation; current class = PoleLocation.
D Checking overloading. Class = String; current class = String.
D Checking overloading. Class = EARootClass; current class = EARootClass.
D Checking overloading. Class = GM_Point; current class = GM_Point.
D Generating XML Schema for application schema Domain Objects.
D Creating XSD document 'PoleLocation.xsd' for package Domain Objects.
D Processing class PoleLocation.
D Processing class 'PoleLocation', rule 'iso19136_2007'.
D Import to namespace 'http://www.opengis.net/gml' added.
D Processing local properties of class PoleLocation.
D Found: gml:PointPropertyType - direct
```

D Found: integer - direct
D Found: string – direct

The resulting xml file is the GML application schema automatically generated from the UML application schema:

```
<?xml version="1.0" encoding="windows-1252"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogr="http://ogr.maptools.org"
targetNamespace="http://ogr.maptools.org" elementFormDefault="qualified"
version="1.0">
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"/>
    <!--XML Schema document created by ShapeChange-->
    <element name="PoleLocation" type="ogr:PoleLocationType"
substitutionGroup="gml:_Feature"/>
    <complexType name="PoleLocationType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="geometryProperty"
type="gml:PointPropertyType"/>
                    <element name="id" type="integer"/>
                    <element name="name" type="string"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="PoleLocationPropertyType">
        <sequence minOccurs="0">
            <element ref="ogr:PoleLocation"/>
        </sequence>
        <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </complexType>
</schema>
```

2.2.2.3.4 Generating a feature collection from the automatically generated application schema.

A pole location GML file is a feature collection of PoleLocation elements. ShapeChange does not generate any Type for feature collections, but we can reuse an element that already exists in wfs schemas by specifying the location of both our application schema (PoleLocation.xsd) and the wfs schemas.

A GML file that includes a single GML point with the position of the geographic north pole (0.0 0.0) looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:FeatureCollection xmlns="http://ogr.maptools.org"
xmlns:wfs="http://www.opengis.net/wfs" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://ogr.maptools.org PoleLocation.xsd
http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
<gml:featureMember>
    <PoleLocation gml:id="F0">
        <geometryProperty>
            <gml:Point>
                <gml:pos>0.0 0.0</gml:pos>
            </gml:Point>
        </geometryProperty>
        <id>0</id>
        <name>Geographic North Pole</name>
    </PoleLocation>
</gml:featureMember>
</wfs:FeatureCollection>
```

2.2.2.4 GML profiles

(This text is based on a blog from Ron Lake :http://www.galdosinc.com/archives/192 and this list: http://www.ogcnetwork.net/gmlprofiles)

GML 3 includes a wide range of elements and data types in 0, 1, 2 and 3 dimension for geometric, topologic, temporary objects, coverages, CSRs, dictionaries, default styles, etc. GML xsd files are arranged in a modular fashion and you only need to import geometryBasic0D1D.xsd for different applications. Many people though, find GML 3.1 possibilities overwhelming and prefer a more limited subset of them. Generic GML profiles lower the entry bar facilitating the understanding of a GML and making it easier to apply and to be read by GML parsers (while improving interoperability). This ensures that the application schemas developed with these profiles, avoid loading components that will never be used. These profiles live in the GML namespaces (http://www.opengeospatial.org/standards/gml) and define restricted subsets of GML. In other words, any application schema that can be generated using a profile must also be a valid application schema with respect to the full GML specification. Examples of this kind of profiles are:

- A very simple GML Point Profile aimed at specification developers that have point geometric data but do not want to use the GML grammar.
- A simple GML for Simple Features profile aimed at supporting vector feature requests and transactions (e.g. to/from a WFS).
- A GML profile for GMJP2 (GML in JPEG 2000)
- A GML profile for RSS (discussed below)

Another kind of GML profile provides the basis for domain or community specific GML application schemas, which in turn support data interoperability within a community of interest. This profile incorporates new XML vocabularies defined using GML and which lives in an application-defined target namespace.

GML application schemas can be built on specific GML profiles or the full GML schema set can be used.

2.2.2.5 Using Simple feature profile.

(This text is based on the blog from Ron Lake :http://www.galdosinc.com/archives/192 and the simple features profile specification document OGC 10-100)

The GML Profile for Simple Features provides enough support for a wide range of vector feature objects. It includes:

[1] A reduced geometry model allowing 0d, 1d and 2d linear geometric objects (all based on linear interpolation) and the corresponding aggregated geometries (gml:MultiPoint, gml:MultiCurve, etc).

[2] A simplified feature model which can only be one level deep (in the general GML model, arbitrary nesting of features and feature properties is not permitted).

[3] All non-geometric properties must be XML Schema simple types – i.e. cannot contain nested elements.

[4] Remote property value references are used (xlink:href), just like in the main GML specification.

Since the profile aims to provide a simple entry point, it does not provide support for coverage, topology, observations, value objects (for real time sensor data), nor support for dynamic features.

Simple features profile defines three compliance levels called SF-0, SF-1, and SF-2.
Compliance level SF-0 limits the property types of both spatial and non-spatial properties. Non-spatial properties are limited to being one of the following common xml types: integer, measurement, date, boolean, binary, URI, character or real; and cardinality of non-spatial properties is limited to zero or one (not allowing more that one). Spatial properties are limited to being one of the following GML types: point, curve with linear and/or circular arc interpolation, planar surface, or aggregates thereof. In GML instance documents compliant with level SF-0, values for a particular property may be encoded inline or by-reference but not both. By-reference property values are only supported through the use of the type gml:ReferenceType (the more generalized GML property-type pattern allowing mixed inline and by-reference encoded property values within the same instance document is disallowed in this level).
Compliance level SF-1 supports everything in compliance level SF-0 but removes the restriction on value cardinality (allowing more that one occurrence of the same property per object) and allows user defined non-spatial property types to be used. As with compliance level SF-0, in GML instance documents compliant with level SF-1, values for a particular property may be encoded inline or by-reference but not both. By-reference property values are only supported through the use of the type gml:ReferenceType.
Finally, compliance level SF-2 essentially corresponds to the OGC Simple Features specification. There are no restrictions placed on non-spatial properties. Spatial properties, however, are limited to the set of geometric types supported by compliance levels SF-0 and SF-1. In instance documents, the restriction on remote or referenced geometric property values, found in compliance levels SF-0 and SF-1, is removed

Nonetheless, it supports a good variety of real world problems and it is quite useful. It is the right profile to convert a Shapefile (SHP) to GML data because Shapefiles have similar limitations.

2.2.2.6 GML media type

When submitting a request to a service or when receiving a response from a service using a GML file, a GML media type format has to be specified. Previously, in OGC several notations where used for this particular media type (e.g. WFS 1.0 uses "GML2", WFS 1.1 uses ""text/xml; subtype=gml/3.1.1" and GML simple features profiles suggests text/xml; subtype=gml/3.1.1/sfgml).

All this notations have been set aside in favor of a standard media type "application/gml+xml" (OGC 09-144r1. http://portal.opengeospatial.org/files/?artifact_id=37743). This MIME type has been submitted to IETF for approval: http://tools.ietf.org/html/draft-portele-ogc-gml-mime-01, which it only has 2 optional parameters: "charset" and "version". For "version" only the major and the first minor version number are provided.

To refer to a particular GML version 3.2 you can use: "application/gml+xml; version=3.2". References to profile are no longer allowed.

2.2.2.7 The axis order issue.

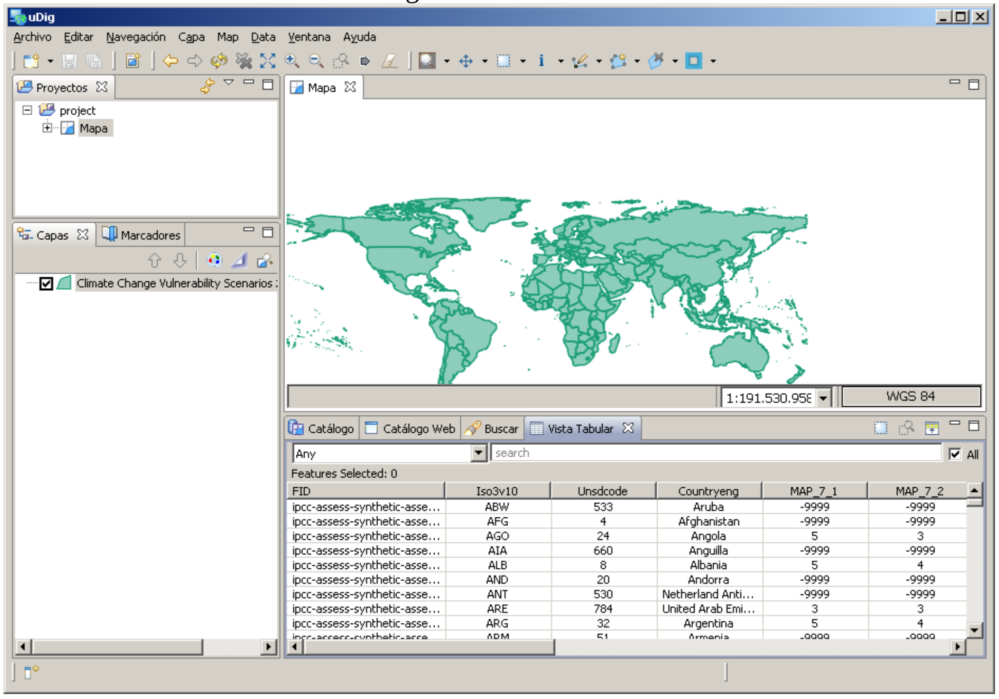
In the previous discussion we have ignored the coordinate reference system upon which the position of geospatial objects rely. In previous examples, we used a polar stereographic projection because data came from an arctic zone dataset. One of the most common geospatial reference systems is latitude/longitude in WGS84 (EPSG code database identifier: 4326). The reference system can be specified by adding an attribute to the gml:Point.
<gml:Point srsName="urn:EPSG:geographicCRS:4326">
(This notation for CRS names comes from the GML 3.1.1 standard examples and it is deprecated; currently OGC is recommending URIs instead of URNs)

There is a known issue with the axis order interpretation in the EPSG:4326. OGC edited the document 08-038r5 (http://www.ogcnetwork.net/node/491) to settle down this issue and to determine the axis order specified in the EPSG database in order to be adopted by all OGC standards. In practice, this means that EPSG:4326 shall be used in latitude/longitude ordering.

Unfortunately, many implementations of GML, as well as many layers continue following the reverse longitude-latitude ordering, this either could be explained because implementations where done before 08-038r5 or because they continue ignoring the OGC recommended standards. Some GML readers have the capability to specify a parameter to invert the coordinates ordering while reading it; this is just a transitory solution until implementations progressively adopt the axis order from the OGC policy.

For instance, the service CIESIN/SEDAC WFS Service (Center for International Earth Science Information Network (CIESIN), Columbia University; and NASA's Socioeconomic Data and Applications Center (SEDAC)) with CSR record https://geossregistries.info/geosspub/login.jsp?redirect=component_details.jsp?compId=urn:uuid:a6554210-6c88-4b81-9625-e0bed5e34b93 has a feature type for IPCC assessment that can be obtained as a GML file with this URL: http://sedac.ciesin.columbia.edu/geoserver/wfs?service=WFS&request=GetFeature&version=1.1.0&typeName=ipcc-assess:ipcc-assess-synthetic-assessment-vulnerability-

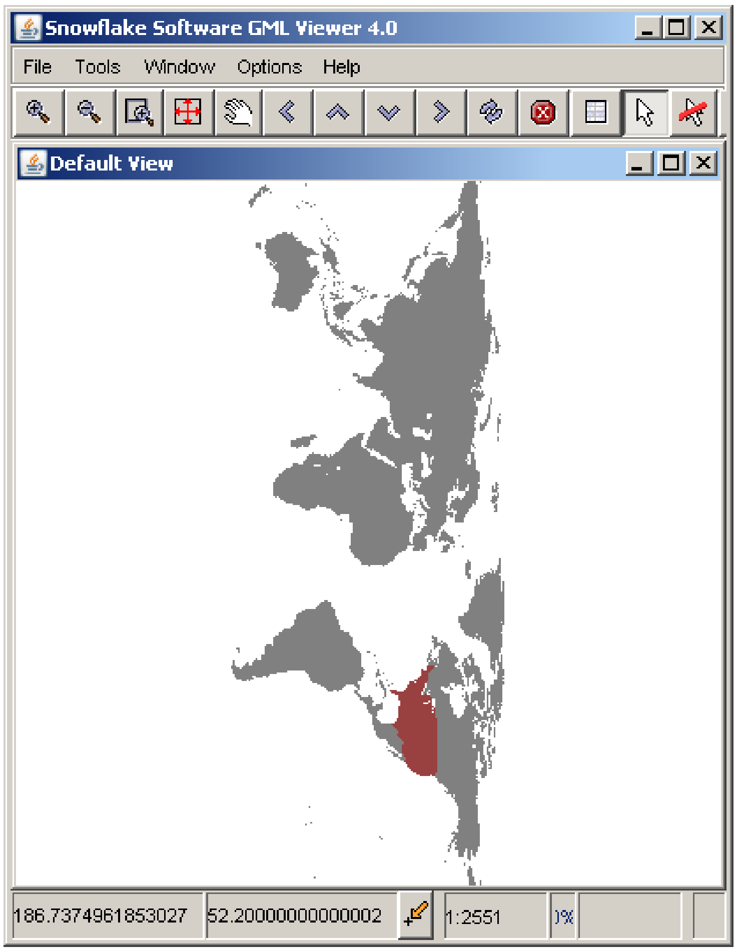
climate-2005-2050-2100. This dataset honors the axis order OGC policy and represent coordinates ordered latitude and longitude.



Features Selected: 0

FID	Iso3v10	Unsdcode	Countryeng	MAP_7_1	MAP_7_2
ipcc-assess-synthetic-asse…	ABW	533	Aruba	-9999	-9999
ipcc-assess-synthetic-asse…	AFG	4	Afghanistan	-9999	-9999
ipcc-assess-synthetic-asse…	AGO	24	Angola	5	3
ipcc-assess-synthetic-asse…	AIA	660	Anguilla	-9999	-9999
ipcc-assess-synthetic-asse…	ALB	8	Albania	5	4
ipcc-assess-synthetic-asse…	AND	20	Andorra	-9999	-9999
ipcc-assess-synthetic-asse…	ANT	530	Netherland Anti…	-9999	-9999
ipcc-assess-synthetic-asse…	ARE	784	United Arab Emi…	3	3
ipcc-assess-synthetic-asse…	ARG	32	Argentina	5	4
ipcc-assess-synthetic-asse	ARM	51	Armenia	-9999	-9999

This is what happens if the client does not take into account the right axis ordering and supposes first longitude and then latitude:


Snowflake Software GML Viewer 4.0
File Tools Window Options Help
Default View
186.7374961853027
52.20000000000002
1:2551

2.2.3 Simple WFS Service Interface. Serving features

WFS 2.0 defines 4 incremental levels of implementation of WFS: Simple WFS, Basic WFS, Transactional WFS and Locking WFS. This allows implementers to limit themselves to a minimum set of operations.

For the Simple WFS, the server shall implement the following operations: GetCapabilities (all versions), DescribeFeatureType (all versions), ListStoredQueries (WFS 2.0 only), DescribeStoredQueries (WFS 2.0 only), GetFeature (all versions) operation with only the StoredQuery (WFS 2.0 only).
One stored query, which fetches a feature using its id, shall be available although the server may also offer additional stored queries. This stored query shall have the identifier: urn:ogc:def:query:OGC-WFS::GetFeatureById (WFS 2.0 only). In WFS version 1.1, a similar functionality was available using the GetGMLObject operation.

To illustrate how WFS works, we are going to use the Atlas of the Cryosphere as an example: Northern Hemisphere. The Atlas of the Cryosphere uses mapserver software (mapserver.org) over Apache/2.2.3 (Linux/SUSE).

2.2.3.1 Describing Your WFS Server: The GetCapabilities operation

2.2.3.1.1 GetCapabilities request

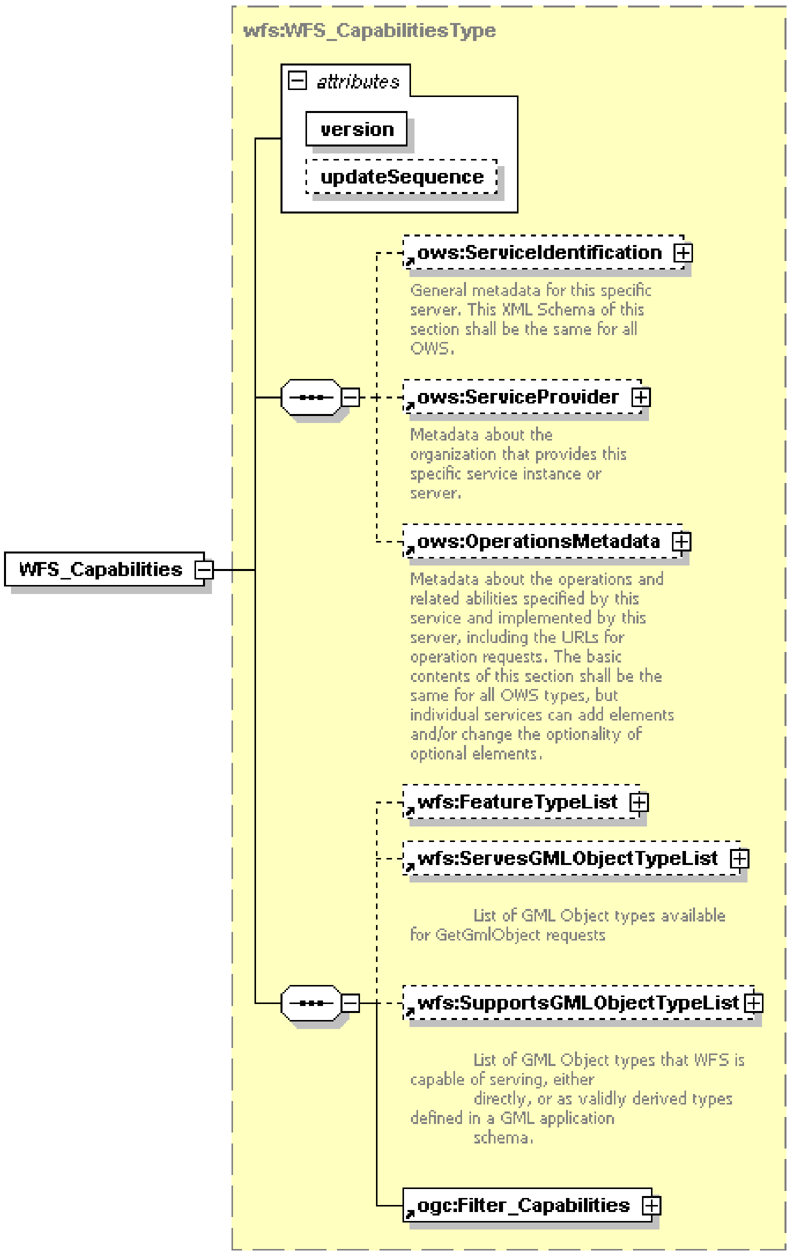
All OWS Common services are self-describing. In the case of WFS this means that you are able to request enough information to the service to know the services details, and you can formulate further requests until you gent the desired data. When you already have the elements you need to get the data, it is then possible to directly request the desired data without a previous information request about the service itself. None the less, it is good that the services are self-describing because it allows the discovery of other services. To explore a WFS service, the only thing you need is the service URL location. There are two possibilities:

- You have a URL that is a GetCapabilites complete request. In this case, you do not need to know anything else. Just using this URL in a web browser will get you the service description. This is what the CSR provides. For example, for the Atlas of the Cryosphere: Northern Hemisphere WFS CSR record (https://geossregistries.info/geosspub/service_details_ns.jsp?serviceId=urn:uuid: 37689b8b-1ffc-40e4-b7d1-e407cb23c7df) you have an entry called "Interface URL" that contains: http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=GetCapabilities
- You only have the URL of the service application: http://nsidc.org/cgi-bin/atlas_north. In this case, you will need to add two mandatory parameters that are the same for all WFS services version 1.1: http://nsidc.org/cgi-bin/atlas_north?**service=WFS&request=GetCapabilities**

Many WFS implementations support more that one standard version (e.g. you can add "&version=1.0.0" to previous requests to get a WFS 1.0 description). In fact, many current services support WFS 1.0 and 1.1. If you do not request any version in particular, you will then get the latest version available.

2.2.3.1.2 Capabilities document

The response of a correct GetCapabilities request is a capabilities document (also known as "OGC service metadata document"). This is an XML document that has different sections.


wfs:WFS_CapabilitiesType
attributes
version
updateSequence
ows:ServiceIdentification
General metadata for this specific server. This XML Schema of this section shall be the same for all OWS.
ows:ServiceProvider
Metadata about the organization that provides this specific service instance or server.
ows:OperationsMetadata
Metadata about the operations and related abilities specified by this service and implemented by this server, including the URLs for operation requests. The basic contents of this section shall be the same for all OWS types, but individual services can add elements and/or change the optionality of optional elements.
WFS_Capabilities
wfs:FeatureTypeList
wfs:ServesGMLObjectTypeList
List of GML Object types available for GetGmlObject requests
wfs:SupportsGMLObjectTypeList
List of GML Object types that WFS is capable of serving, either directly, or as validly derived types defined in a GML application schema.
ogc:Filter_Capabilities

In this example we are going to examine the Atlas of the Cryosphere: Northern Hemisphere GetCapabilities response document that only uses some of the previous sections:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:WFS_Capabilities xmlns:gml="http://www.opengis.net/gml" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ows="http://www.opengis.net/ows" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ogc="http://www.opengis.net/ogc"
xmlns="http://www.opengis.net/wfs" version="1.1.0" xsi:schemaLocation="http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
    <ows:ServiceIdentification>[…]</ows:ServiceIdentification>
    <ows:ServiceProvider>[…]</ows:ServiceProvider>
    <ows:OperationsMetadata>[…]</ows:OperationsMetadata>
    <FeatureTypeList>[…]</FeatureTypeList>
    <ogc:Filter_Capabilities>[…]</ogc:Filter_Capabilities>
</wfs:WFS_Capabilities>
```

The first two sections do not require much explanation because XML element names are quite clear: they provide general information about the service we are going to use such as a title, a description, etc., and information about the people responsible to set it up, as we can see in this XML fragment:

```
<ows:ServiceIdentification>
    <ows:Title>Atlas of the Cryosphere: Northern Hemisphere</ows:Title>
    <ows:Abstract>The National Snow and Ice Data Center (NSIDC) Atlas of the
Cryosphere is a map server that provides data and information pertinent to the frozen
regions of Earth, including monthly climatologies of sea ice extent and concentration,
snow cover extent, and snow water equivalent, in addition to glacier outlines,
permafrost extent and classification, ice sheet elevation and accumulation, and more.
[…]
    </ows:Abstract>
    <ows:Keywords>
        <ows:Keyword>Arctic</ows:Keyword>
        <ows:Keyword>Cryosphere</ows:Keyword>
         […]
    </ows:Keywords>
    <ows:ServiceType codeSpace="OGC">OGC WFS</ows:ServiceType>
    <ows:ServiceTypeVersion>1.1.0</ows:ServiceTypeVersion>
    <ows:Fees>none</ows:Fees>
    <ows:AccessConstraints>none</ows:AccessConstraints>
</ows:ServiceIdentification>
<ows:ServiceProvider>
    <ows:ProviderName>National Snow and Ice Data Center</ows:ProviderName>
    <ows:ProviderSite xlink:type="simple" xlink:href="http://nsidc.org"/>
    <ows:ServiceContact>
        <ows:IndividualName>NSIDC User Services</ows:IndividualName>
         […]
        <ows:PositionName>User Services</ows:PositionName>
        <ows:ContactInfo>
             […]
            <ows:Address>
                 […]
                <ows:ElectronicMailAddress>nsidc@nsidc.org
                        </ows:ElectronicMailAddress>
            </ows:Address>
             […]
        </ows:ContactInfo>
        <ows:Role>resourceProvider</ows:Role>
    </ows:ServiceContact>
</ows:ServiceProvider>
```

OperationsMetadata section is more interesting to us because it reviles operations that the services support, while allowing us to introduce two further operations:

DescribeFeatureType and GetFeature. It also describes the supported values for some parameters like outputFormat.

```
<ows:OperationsMetadata>
    <ows:Operation name="GetCapabilities">[…]</ows:Operation>
    <ows:Operation name="DescribeFeatureType">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get xlink:type="simple"
                xlink:href="http://nsidc.org/cgi-bin/atlas_north?"/>
                <ows:Post xlink:type="simple"
                xlink:href="http://nsidc.org/cgi-bin/atlas_north?"/>
            </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="outputFormat">
            <ows:Value>XMLSCHEMA</ows:Value>
            <ows:Value>text/xml; subtype=gml/2.1.2</ows:Value>
            <ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
        </ows:Parameter>
    </ows:Operation>
    <ows:Operation name="GetFeature">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get xlink:type="simple"
                xlink:href="http://nsidc.org/cgi-bin/atlas_north?"/>
                <ows:Post xlink:type="simple"
                xlink:href="http://nsidc.org/cgi-bin/atlas_north?"/>
            </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="resultType">
            <ows:Value>results</ows:Value>
        </ows:Parameter>
        <ows:Parameter name="outputFormat">
            <ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
        </ows:Parameter>
    </ows:Operation>
</ows:OperationsMetadata>
```

WMS services structure the service content into layers that have names. A WFS server does not do that, so, a WFS server can be seen as an enormous mixed collection of features ready to be downloaded. Fortunately, we can filter the information by feature types and, in practice, people are using feature type filtering as a way to separate content in different units (like we do when separating information into different shapefiles. Obviously, it is important to know the name of the contents in order to separate the feature type content. The next section, FeatureTypeList, tells us the type names as well as many details about each feature type. This service has up to 19 feature types, but the following fragment only shows two of them:

```
    <FeatureTypeList>
[…]
        <FeatureType>
```

```
            <Name>greenland_elevation_contours</Name>
[…]
        </FeatureType>
[…]
        <FeatureType>
            <Name>north_poles_wfs</Name>
            <Title>North Poles</Title>
            <Abstract>Labels the location of various types of North Poles: geographic,
geomagnetic, magnetic, pole of cold, and pole of inaccessibility. Citations:
[…]</Abstract>
            <ows:Keywords>
                <ows:Keyword>Arctic</ows:Keyword>
[…]
            </ows:Keywords>
            <DefaultSRS>EPSG:32661</DefaultSRS>
            <OutputFormats>
                <Format>text/xml; subtype=gml/3.1.1</Format>
            </OutputFormats>
            <ows:WGS84BoundingBox dimensions="2">
                <ows:LowerCorner>-179.398145800191
                                67.4138497780757</ows:LowerCorner>
                <ows:UpperCorner>179.629931564272
                                86.7641921273786</ows:UpperCorner>
            </ows:WGS84BoundingBox>
        </FeatureType>
[…]
    </FeatureTypeList>
```

Finally, this service supports filtering features by several criteria detailed in the last section of the XML file:

```
    <ogc:Filter_Capabilities>
        <ogc:Spatial_Capabilities>
            <ogc:GeometryOperands>
                <ogc:GeometryOperand>gml:Point</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:LineString</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:Polygon</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:Envelope</ogc:GeometryOperand>
            </ogc:GeometryOperands>
            <ogc:SpatialOperators>
                <ogc:SpatialOperator name="Equals"/>
                <ogc:SpatialOperator name="Disjoint"/>
                <ogc:SpatialOperator name="Touches"/>
                <ogc:SpatialOperator name="Within"/>
                <ogc:SpatialOperator name="Overlaps"/>
                <ogc:SpatialOperator name="Crosses"/>
                <ogc:SpatialOperator name="Intersects"/>
                <ogc:SpatialOperator name="Contains"/>
                <ogc:SpatialOperator name="DWithin"/>
                <ogc:SpatialOperator name="Beyond"/>
                <ogc:SpatialOperator name="BBOX"/>
            </ogc:SpatialOperators>
        </ogc:Spatial_Capabilities>
        <ogc:Scalar_Capabilities>
            <ogc:LogicalOperators/>
            <ogc:ComparisonOperators>
                <ogc:ComparisonOperator>LessThan</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>GreaterThan</ogc:ComparisonOperator>

    <ogc:ComparisonOperator>LessThanEqualTo</ogc:ComparisonOperator>

    <ogc:ComparisonOperator>GreaterThanEqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>EqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>NotEqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>Like</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>Between</ogc:ComparisonOperator>
            </ogc:ComparisonOperators>
        </ogc:Scalar_Capabilities>
        <ogc:Id_Capabilities>
            <ogc:FID/>
        </ogc:Id_Capabilities>
    </ogc:Filter_Capabilities>
```

2.2.3.2 Describing Your FeatureTypes: The DescribeFeatureType operation

So far we have been discussing data models in GML and the way to describe them in XML application schemas. DescribeFeatureType is used to obtain application schemas.

2.2.3.2.1 DescribeFeatureType request

To build a DescribeFeatureType request we use the server URL http://nsidc.org/cgi-bin/atlas_north and, for the WFS version 1.1, and adding "?service=WFS&request=DescribeFeatureType&version=1.1.0&typename=". Then, we have two situations:

- We leave the request like this. Then, we are going to get all feature types in the response.
- We add a list of feature type names separated by comas. FeatureType names can be found on the capabilities document for the content of the elements "/wfs:WFS_Capabilities/FeatureTypeList/FeatureType[]/Name".

In our example, we are going to use: north_poles_wfs and greenland_elevation_contours. A valid request to this server is:
http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=DescribeFeatureType&version=1.1.0&typename=north_poles_wfs,greenland_elevation_contours

2.2.3.2.2 DescribeFeatureType response: GML application schema

The response to a well formed DescribeFeatureType request is a GML application schema for the requested types. In our example we are getting:

```
<schema
    targetNamespace="http://mapserver.gis.umn.edu/mapserver"
```

```
     xmlns:ms="http://mapserver.gis.umn.edu/mapserver"
     xmlns:ogc="http://www.opengis.net/ogc"
     xmlns="http://www.w3.org/2001/XMLSchema"
     xmlns:gml="http://www.opengis.net/gml"
     elementFormDefault="qualified" version="0.1" >

    <import namespace="http://www.opengis.net/gml"
            schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd" />

    <element name="greenland_elevation_contours"
                    type="ms:greenland_elevation_contoursType"
                    substitutionGroup="gml:_Feature" />

    <complexType name="greenland_elevation_contoursType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="msGeometry" type="gml:GeometryPropertyType"
minOccurs="0" maxOccurs="1"/>
                    <element name="Label" type="string"/>
                    <element name="Elevation" type="string"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

    <element name="north_poles_wfs"
                    type="ms:north_poles_wfsType"
                    substitutionGroup="gml:_Feature" />

    <complexType name="north_poles_wfsType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="msGeometry" type="gml:GeometryPropertyType"
minOccurs="0" maxOccurs="1"/>
                    <element name="Id" type="string"/>
                    <element name="NAME" type="string"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

</schema>
```

In previous sections we have discussed about data models and the translation into GML applications schemas. This application schema for the north_poles example has already been explained. A particular problem that occurs when implementing it, is that the gml:GeometryPropertyType hides the actual geometric objects used to describe both feature types. We already know that north_poles will be expressed as GML points and we suppose that greenland_elevation_contoursType will be expressed as GML linestrings, but we can not confirm it until we actually request the features. This is why we would recommend that a more concrete subclass of gml:AbstractFeatureType was used instead. It is also surprising that "Elevation" has a "string" type value since we expect numbers in it.

2.2.3.3 Serving a Feature collection: The GetFeature operation

This operation allows us to retrieve a subset of the features that the WFS stores.

2.2.3.3.1 Implementing GetFeature: Required Parameters

To build a DescribeFeatureType request we use the server URL http://nsidc.org/cgi-bin/atlas_north and, for the WFS version 1.1, we add "http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=GetFeature&version=1.1.0&typename=". Then we add a coma separated list of feature type names. FeatureType names can be found on the capabilities document for the content of the elements "/wfs:WFS_Capabilities/FeatureTypeList/FeatureType[]/Name".
In our example, we use: north_poles_wfs or greenland_elevation_contours.
Request of features for both feature types will be something like this:
http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=GetFeature&version=1.1.0&typename=north_poles_wfs
and
http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=GetFeature&version=1.1.0&typename=greenland_elevation_contours

2.2.3.3.2 GetFeature Response. GML data

The response to a well formed GetFeature request is a GML application schema for the requested types. In our first example we get:

```
<wfs:FeatureCollection xmlns:ms="http://mapserver.gis.umn.edu/mapserver"
xmlns:gml="http://www.opengis.net/gml" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://mapserver.gis.umn.edu/mapserver http://nsidc.org/cgi-
bin/atlas_north?SERVICE=WFS&amp;VERSION=1.1.0&amp;REQUEST=DescribeFe
atureType&amp;TYPENAME=north_poles_wfs http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
    <gml:boundedBy>
        <gml:Envelope srsName="EPSG:32661">
            <gml:lowerCorner>903627.830798 1639531.171915</gml:lowerCorner>
            <gml:upperCorner>3845580.759766 3744059.030924</gml:upperCorner>
        </gml:Envelope>
    </gml:boundedBy>
    <gml:featureMember>
        <ms:north_poles_wfs gml:id="north_poles_wfs.0">
            <gml:boundedBy>
                <gml:Envelope srsName="EPSG:32661">
                    <gml:lowerCorner>
                    2000000.000000 2000000.000000</gml:lowerCorner>
                    <gml:upperCorner>
                    2000000.000000 2000000.000000</gml:upperCorner>
                </gml:Envelope>
            </gml:boundedBy>
            <ms:msGeometry>
                <gml:Point srsName="EPSG:32661">
                    <gml:pos>2000000.000000 2000000.000000</gml:pos>
```

Formatat: espanyol (Espanya - alfab. internacional)

```
                </gml:Point>
            </ms:msGeometry>
            <ms:Id>0</ms:Id>
            <ms:NAME>Geographic North Pole</ms:NAME>
        </ms:north_poles_wfs>
    </gml:featureMember>
    <gml:featureMember>
        <ms:north_poles_wfs gml:id="north_poles_wfs.1">
[…]
            <ms:msGeometry>
                <gml:Point srsName="EPSG:32661">
                    <gml:pos>1256015.530206 2337486.323100</gml:pos>
```

Formatat: espanyol (Espanya - alfab. internacional)

```
                </gml:Point>
            </ms:msGeometry>
            <ms:Id>0</ms:Id>
            <ms:NAME>Magnetic North Pole (2005)</ms:NAME>
        </ms:north_poles_wfs>
    </gml:featureMember>
    <gml:featureMember>
[…]
    </gml:featureMember>
</wfs:FeatureCollection>
```

It is interesting to see that the xsi:schemaLocation of the GML application schema is a dynamically generated document that is in fact a DescribeFeatureType request, which will return a GML application schema for these features.

Sometimes things do not work fully as expected. The complete response from the WFS service does not validate in XMLSpy. This is due to some mismatching problems, firstly, the DescribeFeatureType has an optional parameter OutputFormat=text/xml; subtype=gml/3.1.1 (with a space in it), this makes the schema location unreadable. We can fix this by removing this optional parameter, leaving the default value as indicated. Secondly, all gml_id attributes were set to "0", XMLSpy detects this as a violation of the uniqueness of the identifier. The xml fragment illustrated here fixes both issues.

In our second example we get:

```
<wfs:FeatureCollection xmlns:ms="http://mapserver.gis.umn.edu/mapserver"
xmlns:gml="http://www.opengis.net/gml" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://mapserver.gis.umn.edu/mapserver http://nsidc.org/cgi-
bin/atlas_north?SERVICE=WFS&amp;VERSION=1.1.0&amp;REQUEST=DescribeFe
atureType&amp;TYPENAME=greenland_elevation_contours
http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
[…]
    <gml:featureMember>
        <ms:greenland_elevation_contours>
            <gml:boundedBy>
                <gml:Envelope srsName="EPSG:32661">
                    <gml:lowerCorner>
                -327189.501776 -456661.863393</gml:lowerCorner>
                    <gml:upperCorner>
                -327164.993574 -456636.649888</gml:upperCorner>
                </gml:Envelope>
            </gml:boundedBy>
            <ms:msGeometry>
                <gml:LineString srsName="EPSG:32661">
                    <gml:posList srsDimension="2">-327164.993574 -456638.840870 -
327189.501776 -456636.649888 -327183.636846 -456661.863393 -327168.660334 -
456653.526955 -327164.993574 -456638.840870 </gml:posList>
                </gml:LineString>
            </ms:msGeometry>
            <ms:Label>1000 m</ms:Label>
            <ms:Elevation>1000.000000</ms:Elevation>
        </ms:greenland_elevation_contours>
    </gml:featureMember>
    <gml:featureMember>
        <ms:greenland_elevation_contours>
[…]
    </gml:featureMember>
[…]
</wfs:FeatureCollection>
```

As expected, the second example has a geometric element of the gml:LineStringType (a sequence of coordinates). Also, the "Elevation" is a decimal number so it would be better to use xs:decimal instead of xs:string in the definition of the greenland_elevation_contours type.

2.2.3.4. Exceptions

Sometimes the client does not provide a well formed supported request or for some reason, the service is not able to perform the requested task. Under these circumstances, the service will respond with an XML file that explains the origin of the problem.

For example, if we send a GetCoverage request to a WFS service:
http://nsidc.org/cgi-bin/atlas_north?service=WFS&request=GetCoverage&version=1.1.0
it will generate an exception like this:

```
<ows:ExceptionReport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ows="http://www.opengis.net/ows/1.1" version="1.1.0" xml:lang="en-US"
```

```
xsi:schemaLocation="http://www.opengis.net/ows/1.1
http://schemas.opengis.net/ows/1.1.0/owsExceptionReport.xsd">
    <ows:Exception exceptionCode="request" locator="InvalidParameterValue">
        <ows:ExceptionText>msWFSDispatch(): WFS server error. Invalid WFS
request: GetCoverage</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

The sentence: "Invalid WFS request: GetCoverage" reveals the reason of the problem.

- If the origin of the problem is in our side, we could try to identify it and fix it.
- If the origin of the problem is not in our side, there isn't much to be done except to contact the service provider using the contact information available in the ServiceProvider section of the capabilities document.

This is an example of a situation where we cannot fix the problem:

```
<ows:ExceptionReport>
    <ows:Exception exceptionCode="NoApplicableCode">
        <ows:ExceptionText>java.io.FileNotFoundException: Shapefile not
found:file:/usr/local/geoserver/data_dir/data/sibessc/external/iiasa_landresourc/soils/soil
nat_dd.shp</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

The exception report seems to suggest a missing file, but since we do not have access to the service file system, we cannot fix the problem ourselves. This problem was found in the Siberian Earth System Science Cluster WFS service (CSR urn:uuid:1fc2a913-7739-48e2-8222-d1fd2e2f79fd) trying to test a GetGmlObject operation.

2.2.4 Basic WFS Service Interface. Creating new queries

A basic WFS server shall implement the Simple WFS operations, and additionally it shall implement the GetFeature operation with the Query action and the GetPropertyValue operation. All these additions to the Simple WFS have been introduced in WFS 2.0, so previous versions of the standard do not have this level.

2.2.4.1 Forgetting about geometries: The GetPropertyValue operation

Sometimes there is the need for getting specific information about features without getting the geometrical description (since that for linestrings or polygons could take long). WFS 2.0 introduces a new operation that is similar to GetFeature but permits the selection of the properties recovered.

2.2.5 Transactional WFS Service Interface. Updating features

A transactional WFS server (many times referred to as WFS-T) shall implement the Basic WFS operations and it shall also implement the Transaction operation. The transaction operation is a 4 in 1 operation that changes the direction of the information flow. With this operation the user is able to send information to the server and update the data stored in the server. The data becomes immediately available to others. It is composed of 4 operations: "insert", "update” and "delete" and other possible vendor specific operations using the "native" operation.

2.2.5.1 Implementing an Insert Transaction: Required Parameters

It will be more complicated to illustrate this feature, mainly because the data that has to be sent to the server can not be easily carried through the URL in the form of key and value pairs (what is known as HTTP-GET method with KVP parameters), but it has to be transported in an XML file.

In this example, we are going to use the Haiti SDI VGI WFS-T example (https://geossregistries.info/geosspub/service_details_ns.jsp?serviceId=urn:uuid:e234ac3c-9e55-41de-9908-dfed4d3fe9ef). It is a Web Feature Server implemented using the CubeWerx Suite 5.5.2.

We know this service supports transactions because the OperationsMetadata section in the Capabilities document declares to support this operation (http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DATASTORE=vgi&service=WFS&request=GetCapabilities):

```
    <ows:OperationsMetadata>
        <ows:Operation name="GetCapabilities">[…]</ows:Operation>
        <ows:Operation name="DescribeFeatureType">[…]</ows:Operation>
        <ows:Operation name="GetFeature">[…]</ows:Operation>
        <ows:Operation name="Transaction">[…]</ows:Operation>
            <ows:DCP>
                <ows:HTTP>
                    <ows:Post
xlink:href="http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&amp;DATASTORE=vgi"/>
                </ows:HTTP>
            </ows:DCP>
[…]
            <ows:Parameter name="idgen">
                <ows:Value>GenerateNew</ows:Value>
            </ows:Parameter>
[…]
        </ows:Operation>
[…]
    <ows:OperationsMetadata>
```

From the set of FeatureTypes available on this service we have chosen one that is called cw:POINTS.

```
        <FeatureType>
            <Name>cw:POINTS</Name>
            <Title>Points of Interest</Title>
            <DefaultSRS>EPSG:4326</DefaultSRS>
            <ows:WGS84BoundingBox>
                <ows:LowerCorner>-74.4074639 17.8961279</ows:LowerCorner>
                <ows:UpperCorner>-68.3466872 20.0086847</ows:UpperCorner>
            </ows:WGS84BoundingBox>
        </FeatureType>
```

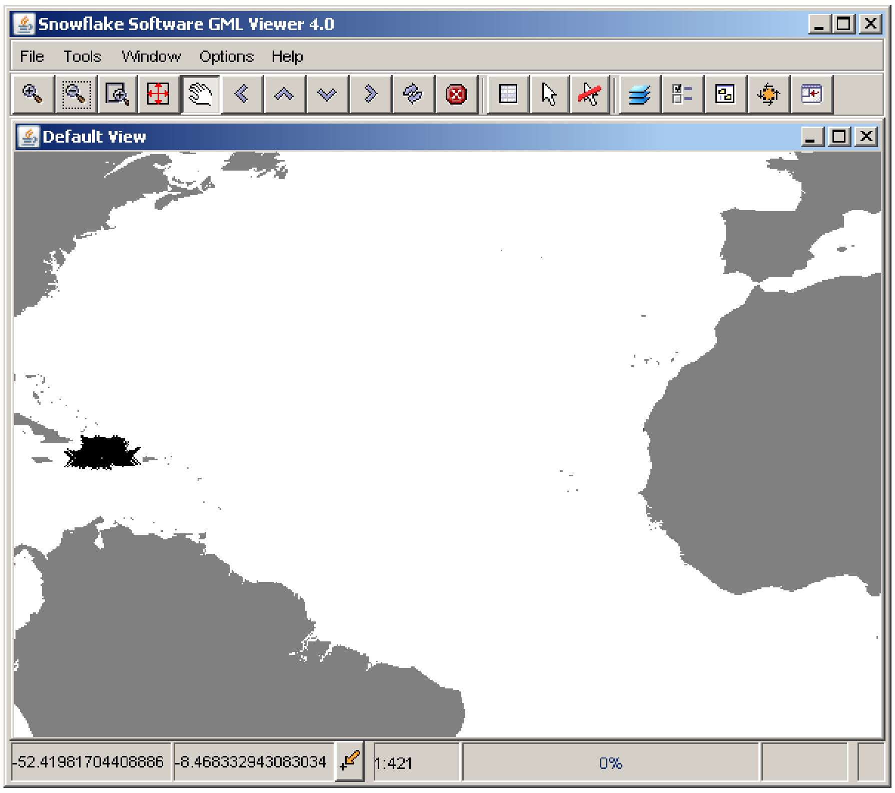
We also know the GML application schema that describes them using http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DATASTORE=vgi&service=WFS&version=1.1.0&request=DescribeFeatureType&TypeName=cw:POINTS:

```
<xs:schema targetNamespace="http://www.cubewerx.com/cw"
xmlns:cw="http://www.cubewerx.com/cw"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml"
xmlns:cwmeta="http://www.cubewerx.com/cwmeta" elementFormDefault="qualified"
version="1.0">
    <xs:import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"/>
    <xs:import namespace="http://www.cubewerx.com/cwmeta"
schemaLocation="http://www.cubewerx.com/schemas/cwmeta/1.0.0/cwmeta.xsd"/>
    <xs:element name="POINTS" type="cw:POINTSType"
substitutionGroup="gml:_Feature"/>
    <xs:complexType name="POINTSType">
        <xs:complexContent>
            <xs:extension base="gml:AbstractFeatureType">
                <xs:sequence>
                    <xs:element name="POINTS.GEOMETRY"
type="gml:PointPropertyType"/>
                    <xs:element name="POINTS.OSM_ID" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction base="xs:integer">
                                <xs:totalDigits value="10"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="POINTS.TIMESTAMP_" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:maxLength value="20"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="POINTS.NAME" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:maxLength value="48"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="POINTS.TYPE_" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:maxLength value="16"/>
                            </xs:restriction>
                        </xs:simpleType>
```

Formatat: alemany (Alemanya)

```
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
</xs:schema>
```

We can also get the current content of the Features of this feature type by doing: http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DATASTORE=vgi&service=WFS&version=1.1.0&request=GetFeature&TypeName=cw:POINTS.


Snowflake Software GML Viewer 4.0
File Tools Window Options Help
Default View
-52.41981704408886
-8.468332943083034
1:421
0%

A particular point looks like:

```
    <gml:featureMember>
        <POINTS gml:id="CWFID.POINTS.0.0.61CB7559D106A1BE1F20020000">
            <POINTS.GEOMETRY>
                <gml:Point srsName="EPSG:4326">
                    <gml:pos>-74.1659157 18.2752643</gml:pos>
                </gml:Point>
            </POINTS.GEOMETRY>
            <POINTS.OSM_ID>616162720</POINTS.OSM_ID>
            <POINTS.TIMESTAMP_>2010-01-
18T11:43:29Z</POINTS.TIMESTAMP_>
            <POINTS.NAME>2010-01-18T11:43:29Z</POINTS.NAME>
            <POINTS.TYPE_>place_of_worship</POINTS.TYPE_>
        </POINTS>
```

Formatat: espanyol (Espanya - alfab. internacional)

```
    </gml:featureMember>
```

Note that this particular server is not honoring the OGC axis ordering policy, so we have to order the coordinates in longitude-latitude ordering.

Now we have enough information to prepare a new point to be inserted in the server. We are going to insert "Barcelona" in the data because it will be clearly visible as a point far away from Haiti. We have to prepare an XML file that looks like the feature collection we have just downloaded but with different headers:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:Transaction version="1.1.0" service="WFS"
xmlns="http://www.cubewerx.com/cw" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cubewerx.com/cw
http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&amp;
DATASTORE=vgi&amp;service=WFS&amp;version=1.1.0&amp;request=DescribeFe
atureType&amp;outputFormat=GML3&amp;typeName=POINTS
http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd
http://www.opengis.net/gml http://schemas.opengis.net/gml/3.1.1/base/feature.xsd">
        <wfs:Insert idgen="GenerateNew">
                <POINTS gml:id="Catalonia">
                        <POINTS.GEOMETRY>
                                <gml:Point srsName="EPSG:4326">
                                        <gml:pos>2.17 41.39</gml:pos>
                                </gml:Point>
                        </POINTS.GEOMETRY>
                        <POINTS.OSM_ID>1111</POINTS.OSM_ID>
                        <POINTS.TIMESTAMP_>2011-08-
14T11:00:00Z</POINTS.TIMESTAMP_>
                        <POINTS.NAME>Barcelona</POINTS.NAME>
                        <POINTS.TYPE_>Non Haiti place</POINTS.TYPE_>
                </POINTS>
        </wfs:Insert>
</wfs:Transaction>
```

Formatat: espanyol (Espanya - alfab. internacional)

Let's save this file as insert.xml. To transfer an XML to a server without a specifically developed WFS-T client application, can be done by using the gnu wget application (http://gnuwin32.sourceforge.net/packages/wget.htm). The wget application has an optional parameter "--post-file=file_name" that allows us to send some file to the service as a POST method. We have also to specify --header="Content-Type:text/xml to report that we are transferring an xml file. This is the complete syntax:

```
wget -O insert_response.xml --post-
file="insert_request.xml" --header="Content-Type:text/xml
http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?C
ONFIG=haiti_vgi&DATASTORE=vgi
```

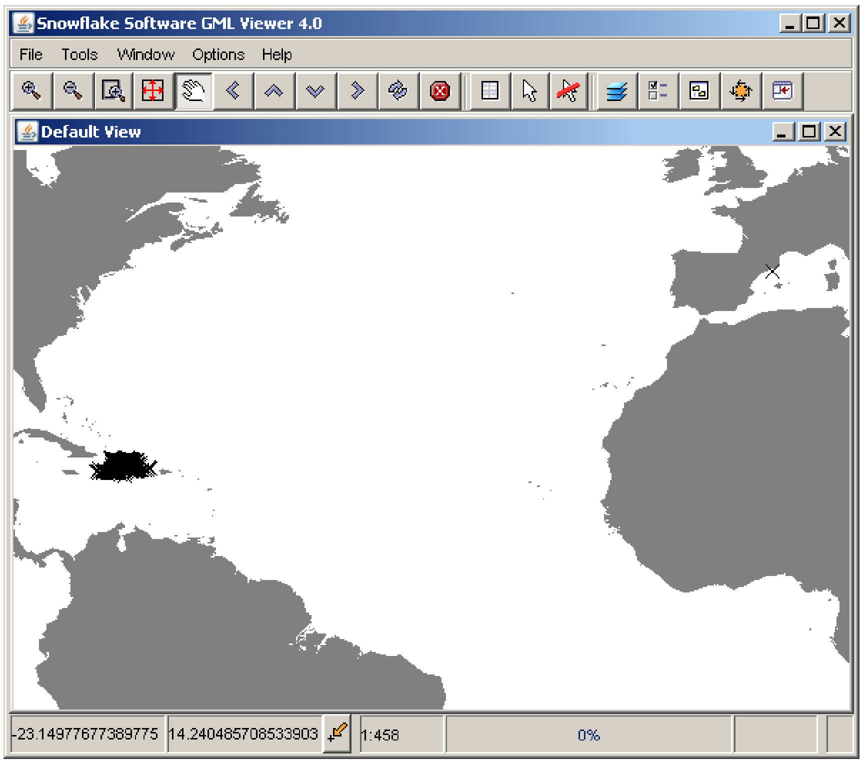
2.2.5.2. Insert transaction response.

To a correct transaction request the service will answer with a transaction report. The previous example report is:

```
<wfs:TransactionResponse xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd http://www.opengis.net/ogc
http://schemas.opengis.net/filter/1.1.0/filter.xsd" version="1.1.0">
	<wfs:TransactionSummary>
		<wfs:totalInserted>1</wfs:totalInserted>
		<wfs:totalUpdated>0</wfs:totalUpdated>
		<wfs:totalDeleted>0</wfs:totalDeleted>
	</wfs:TransactionSummary>
	<wfs:InsertResults>
		<wfs:Feature handle="Action #1">
			<ogc:FeatureId
fid="CWFID.POINTS.0.1454.C546D1DB448763701F20020000"/>
		</wfs:Feature>
	</wfs:InsertResults>
</wfs:TransactionResponse>
```

As you can see, we have inserted a new object that has received a new gml:id because we requested that by indicating: idgen="GenerateNew".

We can get the features of the cw:POINTS feature type again: http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DATASTORE=vgi&service=WFS&version=1.1.0&request=GetFeature&TypeName=cw:POINTS and we get:


Snowflake Software GML Viewer 4.0
File Tools Window Options Help
Default View
-23.14977677389775 14.240485708533903
1:458
0%

Where you can see our new feature as the last one:

```
    <gml:featureMember>
        <POINTS
gml:id="CWFID.POINTS.0.1454.C546D1DB448763701F20020000">
            <POINTS.GEOMETRY>
                <gml:Point srsName="EPSG:4326">
                    <gml:pos>2.17 41.39</gml:pos>
                </gml:Point>
            </POINTS.GEOMETRY>
            <POINTS.OSM_ID>1111</POINTS.OSM_ID>
            <POINTS.TIMESTAMP_>2011-08-
14T11:00:00Z</POINTS.TIMESTAMP_>
            <POINTS.NAME>Barcelona</POINTS.NAME>
            <POINTS.TYPE_>Non Haiti place</POINTS.TYPE_>
        </POINTS>
    </gml:featureMember>
```

Formatat: espanyol (Espanya - alfab. internacional)

2.2.5.3 Implementing an Update Transaction: Required Parameters

We are going to update the OSM_ID that we have introduced in our previous example. We have to prepare a transaction file with a point for updating it in the server. The file looks similar to the previous one but with some modifications:

```
<wfs:Transaction version="1.1.0" service="WFS"
xmlns="http://www.cubewerx.com/cw" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cubewerx.com/cw
http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&amp;
DATASTORE=vgi&amp;service=WFS&amp;version=1.1.0&amp;request=DescribeFe
atureType&amp;outputFormat=GML3&amp;typeName=POINTS
http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd
http://www.opengis.net/gml http://schemas.opengis.net/gml/3.1.1/base/feature.xsd">
        <wfs:Update typeName="cw:POINTS">
                <wfs:Property>
                        <wfs:Name>cw:POINTS.OSM_ID</wfs:Name>
                        <wfs:Value>9999</wfs:Value>
                </wfs:Property>
                <ogc:Filter>
                        <ogc:FeatureId
fid="CWFID.POINTS.0.1454.C546D1DB448763701F20020000"/>
                </ogc:Filter>
        </wfs:Update>
</wfs:Transaction>
```

Let's save this file as update.xml and transfer the file to the server again:

```
wget -O update_response.x
ml --post-file="update_request.xml" --header="Content-
Type:text/xml" "http://portal.cubew
erx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DAT
ASTORE=vgi"
```

2.2.5.4. Update transaction response.

To a correct transaction request the service will answer with a transaction report. In our previous example the report is:

```
<wfs:TransactionResponse xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd http://www.opengis.net/ogc
http://schemas.opengis.net/filter/1.1.0/filter.xsd" version="1.1.0">
    <wfs:TransactionSummary>
        <wfs:totalInserted>0</wfs:totalInserted>
        <wfs:totalUpdated>1</wfs:totalUpdated>
        <wfs:totalDeleted>0</wfs:totalDeleted>
    </wfs:TransactionSummary>
    <wfs:InsertResults/>
</wfs:TransactionResponse>
```

We can get the features of the cw:POINTS feature type again and see our updated feature as the last one:

```
    <gml:featureMember>
        <POINTS
gml:id="CWFID.POINTS.0.1454.C546D1DB448763701F20020000">
            <POINTS.GEOMETRY>
```

```
                <gml:Point srsName="EPSG:4326">
                    <gml:pos>2.17 41.39</gml:pos>
                </gml:Point>
            </POINTS.GEOMETRY>
            <POINTS.OSM_ID>9999</POINTS.OSM_ID>
            <POINTS.TIMESTAMP_>2011-08-
14T11:00:00Z</POINTS.TIMESTAMP_>
            <POINTS.NAME>Barcelona</POINTS.NAME>
            <POINTS.TYPE_>Non Haiti place</POINTS.TYPE_>
        </POINTS>
    </gml:featureMember>
```

Formatat: espanyol (Espanya - alfab. internacional)

2.2.5.5 Implementing a Delete Transaction: Required Parameters

We are going to delete the feature we introduced in our previous example to leave the dataset clean again. We have to prepare a transaction file with a point for updating it in the server. Te file looks similar to the previous one but with some modifications:

```
<wfs:Transaction version="1.1.0" service="WFS"
xmlns="http://www.cubewerx.com/cw" xmlns:wfs="http://www.opengis.net/wfs"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cubewerx.com/cw
http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&amp;
DATASTORE=vgi&amp;service=WFS&amp;version=1.1.0&amp;request=DescribeFe
atureType&amp;outputFormat=GML3&amp;typeName=POINTS
http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd
http://www.opengis.net/gml http://schemas.opengis.net/gml/3.1.1/base/feature.xsd">
    <wfs:Delete typeName="cw:POINTS">
        <ogc:Filter>
            <ogc:FeatureId fid="
CWFID.POINTS.0.1454.C546D1DB448763701F20020000"/>
        </ogc:Filter>
    </wfs:Delete>
</wfs:Transaction>
```

Let's save this file as update.xml and transfer the file to the server again:

```
wget -O delete_response.x
ml --post-file="delete_request.xml" --header="Content-
Type:text/xml" "http://portal.cubew
erx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DAT
ASTORE=vgi"
```

2.2.5.6. Delete transaction response

To a correct transaction request the service will answer with a transaction report. In our previous example the report is:

```
<wfs:TransactionResponse xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd http://www.opengis.net/ogc
http://schemas.opengis.net/filter/1.1.0/filter.xsd" version="1.1.0">
	<wfs:TransactionSummary>
		<wfs:totalInserted>0</wfs:totalInserted>
		<wfs:totalUpdated>0</wfs:totalUpdated>
		<wfs:totalDeleted>1</wfs:totalDeleted>
	</wfs:TransactionSummary>
	<wfs:InsertResults/>
</wfs:TransactionResponse>
```

2.2.6 Locking WFS Service Interface. Decentralized updating features

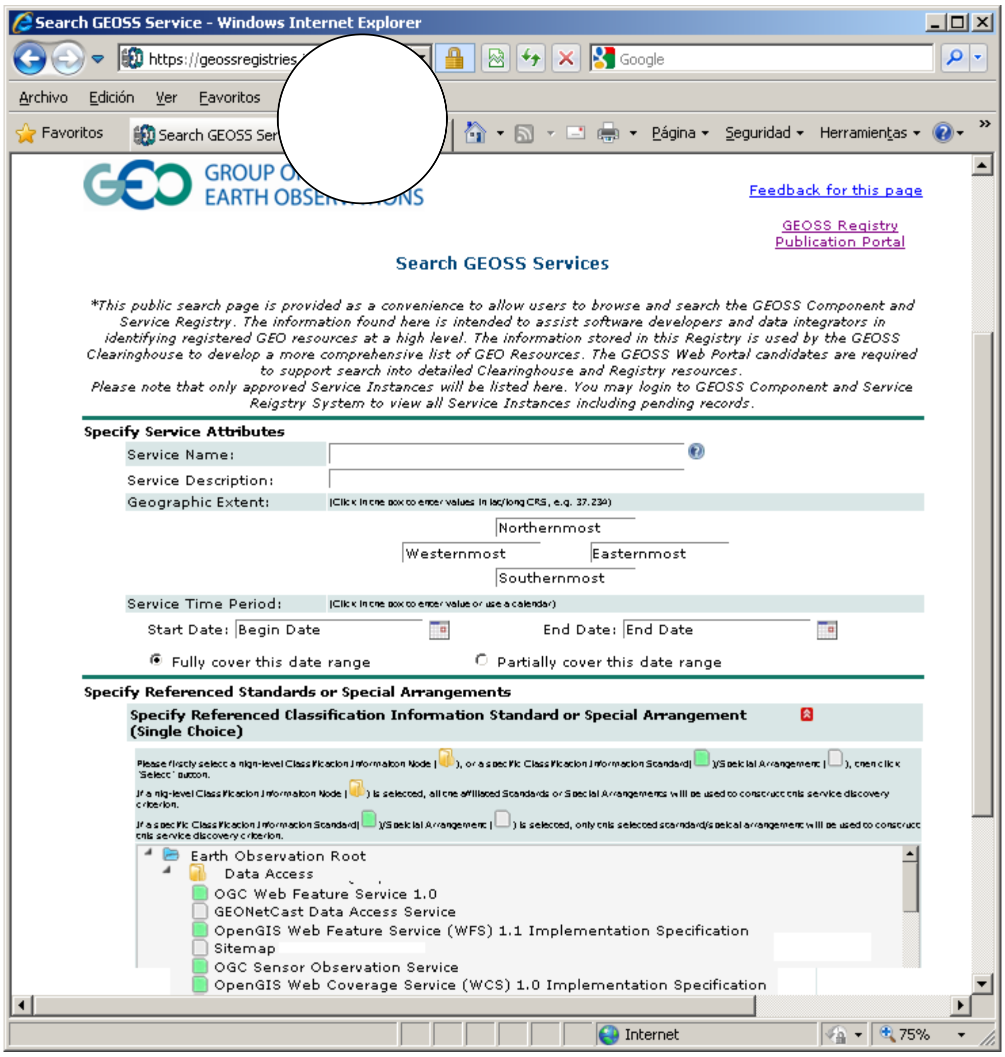
A locking WFS server shall implement the Transactional WFS operations and shall implement at least one of the GetFeatureWithLock or LockFeature operations.

In a distributed environment, a client that has downloaded a dataset form a server has no way to guarantee that while the feature was being modified on the client side, no other client was updateing that same feature in the database.
A way to avoid concurrent editions is to require that access to data has to be done in a mutually exclusive manner; which means that while one transaction accesses a data item, no other transaction can modify the same data item. This can be accomplished by using locks that control access to the data with LockFeature and GetFeatureWithLock and using the received LockId in the Transaction operation.

2.3 Reference to associated videos, books, and other resources, when available

2.3.1 How to find WFS services in the CSR

The page https://geossregistries.info/geosspub/geoss_search_ns.htm of the CSR allows you to find services that use WFS. You have to choose to look for **services** and then choose the appropriate "Referenced Classification Information Standard or Special Arrangement (Single Choice)". In this taxonomy you have to look for "Data access" and look for Web Feature Service standard. Once selected, you can press select and then search. At least there are 2 standards registered (1.0 and 1.1) but you can only select one of them for each search.


Search GEOSS Service - Windows Internet Explorer
https://geossregistries
Google
Archivo Edición Ver Favoritos
Favoritos
Search GEOSS Ser
Página Seguridad Herramientas
GROUP ON EARTH OBSERVATIONS
Internet
75%

Feedback for this page

GEOSS Registry Publication Portal

Search GEOSS Services

*This public search page is provided as a convenience to allow users to browse and search the GEOSS Component and Service Registry. The information found here is intended to assist software developers and data integrators in identifying registered GEO resources at a high level. The information stored in this Registry is used by the GEOSS Clearinghouse to develop a more comprehensive list of GEO Resources. The GEOSS Web Portal candidates are required to support search into detailed Clearinghouse and Registry resources.
Please note that only approved Service Instances will be listed here. You may login to GEOSS Component and Service Reigstry System to view all Service Instances including pending records.

Specify Service Attributes

Service Name:

Service Description:

Geographic Extent: (Click in the box to enter values in lat/long CRS, e.g. 37.234)

Northernmost
Westernmost Easternmost
Southernmost

Service Time Period: (Click in the box to enter value or use a calendar)

Start Date: Begin Date End Date: End Date

[✓ Fully cover this date range] Partially cover this date range

Specify Referenced Standards or Special Arrangements

Specify Referenced Classification Information Standard or Special Arrangement (Single Choice)

Please firstly select a high-level Classification Information Node ( ), or a specific Classification Information Standard( )/Special Arrangement ( ), then click 'Select' button.

If a hig-level Classification Information Node ( ) is selected, all the affiliated Standards or Special Arrangements will be used to construct this service discovery criterion.

If a specific Classification Information Standard( )/Special Arrangement ( ) is selected, only this selected standard/special arrangement will be used to construct this service discovery criterion.

- Earth Observation Root
  - Data Access
    - 🟩 OGC Web Feature Service 1.0
    - ☐ GEONetCast Data Access Service
    - 🟩 OpenGIS Web Feature Service (WFS) 1.1 Implementation Specification
    - ☐ Sitemap
    - 🟩 OGC Sensor Observation Service
    - 🟩 OpenGIS Web Coverage Service (WCS) 1.0 Implementation Specification

Unfortunately, many people do not register the standards associated to a particular service and thus, you will not find them by using the procedure just explained. Instead, by simply searching for the word "WFS" in the name of the service, extra WFS services can be discover that were not carefully registered.

3 Use Cases

3.1 Use Case: Offering a DXF file with mapserver WFS

3.1.1 Introduction of use case

MapServer is an Open Source geographic data rendering engine written in C. Beyond browsing GIS data, MapServer allows you create "geographic image maps", that is, maps that can direct users to content.
MapServer is a popular Open Source project whose purpose is to display dynamic spatial maps over the Internet.
Some of its major features include:

- support for display and querying of hundreds of raster, vector, and database formats
- ability to run on various operating systems (Windows, Linux, Mac OS X, etc.)
- support for popular scripting languages and development environments (PHP, Python, Perl, Ruby, Java, .NET)
- on-the-fly projections
- high quality rendering
- fully customizable application output
- many ready-to-use Open Source application environments

In its most basic form, MapServer is a CGI program that sits inactive on your Web server. When a request is sent to MapServer, it uses information passed in the request URL and the Mapfile to create an image of the requested map. The request may also return images for legends, scale bars, reference maps, and values passed as CGI variables.

MapServer was originally developed by the University of Minnesota (UMN) ForNet project in cooperation with NASA, and the Minnesota Department of Natural Resources (MNDNR). Later it was hosted by the TerraSIP project, a NASA sponsored project between the UMN and a consortium of land management interests.

MapServer is now a project of OSGeo, and is maintained by a growing number of developers (nearing 20) from around the world. It is supported by a diverse group of organizations that fund enhancements and maintenance, and administered within OSGeo by the MapServer Project Steering Committee made up of developers and other contributors.

More information can be found at http://mapserver.org/MapServer.pdf

Firstly, we are going to describe the installation of mapserver by using two different installers. And later, we are going to illustrate how to offer a DXF file.

3.1.2 Description of use case

A DXF, or Drawing Exchange Format, is an open sourced CAD data file format. It was developed by Autodesk in 1982 to enable data interoperability of 2D and 3D drawings between AutoCAD and other programs. Ever since, it has been updated several times, and the latest version (Release 14) supports both ASCII and binary forms. Files created with the earlier versions can also be opened with the later releases.

In this particular case we are going to illustrate in detail on how to publish vector data in DXF format on Mapserver. In particular, our example is going to combine (2D) vector data (points and polygons) from position.

3.1.3 Pointer from parts of use case to tutorial sections

3.1.4 Example used to showcase use case

All data used was originally from WFS servers registered and compliant with the OGC standards. This data is from the positions of birds, reptiles and amphibians of the Siberian region. (Section 4.3.1.2)

3.2 User case: Offering a data with geoserver WFS

3.2.1 Introduction of use case

In this tutorial we are going to describe the steps to set up the Geoserver in your computer. We are also going to illustrate how to offer data with Geoserver WFS, in particular, we are going to explain in detail how to offer shapefile data and PostGIS tables.

3.2.2 Description of use case

The characteristics of the data we are going to offer are:
- A Shapefile, which is a collection of files (with the extensions: .shp, .dbf, .shx, .prj, and sometimes others). All of these files have to be in the same directory in order for GeoServer to accurately read them.

- A PostGIS table, it is an open source spatial database based on PostgreSQL, and it is currently one of the most popular open source spatial databases today.

In both examples we are going to create a new “Workspace” and a new “Store”. Then we are going to select the Vector data we want to upload, and finally all information about the new layer will have to be edited.

3.2.3 Pointer from parts of use case to tutorial sections

3.2.4 Examples used to showcase use case

For the Shapefile, the use case is going to be illustrated by a coverage related to the Arctic layer. (section 4.3.2.2)
For the PostGIS database, the example used is a map of NewYork buildings (Section 4.3.2.3)

3.3 User case: Offering a shapefile with deegree

3.3.1 Introduction of use case

To offer a shapefile with deegree, we are going to guide you on the steps to install deegree. These are products easily-installable packages / configurations of deegree components (e.g. web services) bundled for certain use-cases.

After more than 5 years of ongoing development, the deegree project is the most extensive implementation of OGC/ISO standards in the field of Free Software. deegree evolved from the project JaGo and has been published under the terms of the Lesser GNU Public License (L-GPL) of the Free Software Foundation. Widespread improvements of the architecture, enhancements of the object model and the support of Java 5 guided to deegree to the second major release.

In deegree2, there are OGC WebServices for Web Map Service (WMS) 1.1.1, Web Feature Service (WFS) 1.1.0, Web Coverage Service (WCS) 1.0.0 and Catalogue Service Web-Profile (CSW) 2.0.0 available. WMS and WCS are the official reference implementations of the Open Geospatial Consortium; WFS and CSW are fully transactional. CSW supports ISO19115/ISO19119 Application Profile and DE-Profile 1.0.1.

Furthermore, there are implementations of the OGC pre-standards of Sensor Observation Service (SOS), Web Terrain Service / Web Perspective and View Service (WTS/WPVS) as well as Web Processing Service (WPS). The security package contains a user and rights management system, owsProxy as a security facade of OGC Web Services and Web Authentication (WAS) and Web Security Service (WSS) definied in Spatial Data Infrastructure of NRW.

deegree iGeoPortal standard and portlet edition are the browser-based clients for the mentioned OGC WebServices. deeJUMP provides a SDI-enabled Desktop-GIS.

Besides the improvements of the architecture and the development process (e.g. automatic builds) the following are also important changes for users:

- Simplified installation and configuration
- Tool-based configuration (for WFS and WCS)
- Support of GML 3.1 with a complex Feature Model and 3D-geometries
- Support of PostGIS 1.0 and Oracle spatial/locator (9i/10g)
- Advanced capabilities for object-relational mappings in the WFS
- Multiple data sources for WMS layers
- Dynamic rendering rules within SLD
- High-quality and large-size print outputs through Web Map Print Service (WMPS)

3.3.2 Description of use case

This example is going to describe how to publish a shapefile with deegree, by describing the installation of the program and the packages needed.

3.3.3 Pointer from parts of use case to tutorial sections

3.3.4 Example used to showcase use case

This particular use case is going to be illustrated using the example of the borders of the world. This is a shapefile document that has been modified to be correctly be served by a degree server.

4 Provisioning/Using the service or application

4.1Requirements

4.1.1 Requirements for offering a DXF with mapserver WFS

Hardware requirements: MapServer runs on Linux, Windows, Mac OS X, Solaris, and more. To compile or install some of the required programs, you may need administrative rights to the machine. People commonly ask questions about minimum hardware specifications for MapServer applications, but the answers are really specific to the individual application. For development and learning purposes, a very minimal machine will work fine. For deployment, you will want to investigate Optimization of everything from your data to server configuration.

Software requirements: You need a working and properly configured Web (HTTP) server, such as Apache or Microsoft Internet Information
Server, on the machine on which you are installing MapServer. OSGeo4W contains Apache already, but you can reconfigure things to use IIS if you need to. Alternatively, MS4W can be used to install MapServer on Windows. If you are on aWindows machine, and you don’t have a web server installed, you may want to check out MS4W, which will install a pre-configured web server, MapServer, and more. The FGS Linux Installer provides similar functionality for several Linux distributions. This introduction will assume you are using pre-compiled OSGeo4W Windows binaries to follow along. Obtaining MapServer or Linux or Mac OS X should be straightforward. Visit Download for installing pre-compiled MapServer builds on Mac OS X and Linux. You will also need aWeb browser, and a text editor (vi, emacs, notepad, homesite) to modify your HTML and mapfiles.

In order for Mapserver to offer a DXF, the GRASS application has to be installed. When using the Osgeo4W installer GRASS is one of the applications that is optional to download. We strongly recommend using these application with mapserver when working with DXF images.

4.1.2 Requirements for offering a data with geoserver WFS

For Geoserver to run, it needs Java JDK to be installed.

The requirements to offer a Shapefile with Geoserver are that this file .shp, needs to be accompanied with 3 more files, in order for the Geoserver to be alble to read it. The files needed are .dvf, .shp, .shx and .xml. All of them need to be stored under the same folder for the server to find it.

Note: While GeoServer has robust support for the shapefile format, it is not the recommended format of choice in a production environment. Databases such as PostGIS are more suitable in production and offer better performance and scalability. See the section on *Running in a Production Environment* for more information. (Geoserver web page http://docs.geoserver.org/stable/en/user/data/shapefile.html).

Shapefiles are a very common format for geospatial data. But if you are running GeoServer in a production environment, it is better to use a spatial database such as PostGIS. This is essential if doing transactions (WFS-T). Most spatial databases provide shapefile conversion tools. Although there are many options for spatial databases (see the section on *Working with Data*), PostGIS is recommended. Oracle, DB2, and ArcSDE are also supported.

4.1.3 Requirements for offering a shapefile with degree

The only mandatory requirement for deegree is the correct Java installation.

4.2 Summary of Steps

4.2.1 Summary of Steps for offering a DXF file with mapserver WFS

When offering a DXF file with mapserver, it is recommended to download the packaged specified, such as GRASS, to ease the process. Next step is to define the Location in GRASS, which are the working folder and the project space.

4.2.2 Summary of Steps for offering a data with geoserver WFS

In order to offer data with Geoserver, there are common steps that need to be followed, and these are to start by adding a new "Workingspace", adding a new "Store" where all the layers from these new project will be stored, and by "adding a new layer". At this point, there is the need to define all the characteristics specific to the particular layer to be offered (format, coordinates, src,…).

4.2.3 Summary of Steps for offering a shapefile with deegree

Adding new shapefile data can be done by creating a new Feature store as "shape" type, and then, on the window that opens, manually edit the positions of the vector shapefiel. Here, the content of the <File> also have to be changed by the relative position of the shpafile, but without the extension .shp. Finally, change the <FeatureTypeName> for the new name that you want to be published in the WFS. Now you just need to reload the server to view the new layer.

4.3 Detail Steps for Use Cases

4.3.1 Offering a DXF file with mapserver WFS

4.3.1.1. Step 1: MapServer Installation

MapServer is an Open Source project which can be installed from several Windows Installers. In this tutorial we are going to describe two different installers: OSGeo4W and MS4W. In both cases a step by step explanation is done, while specifying requirements of the system and common errors during the installation.

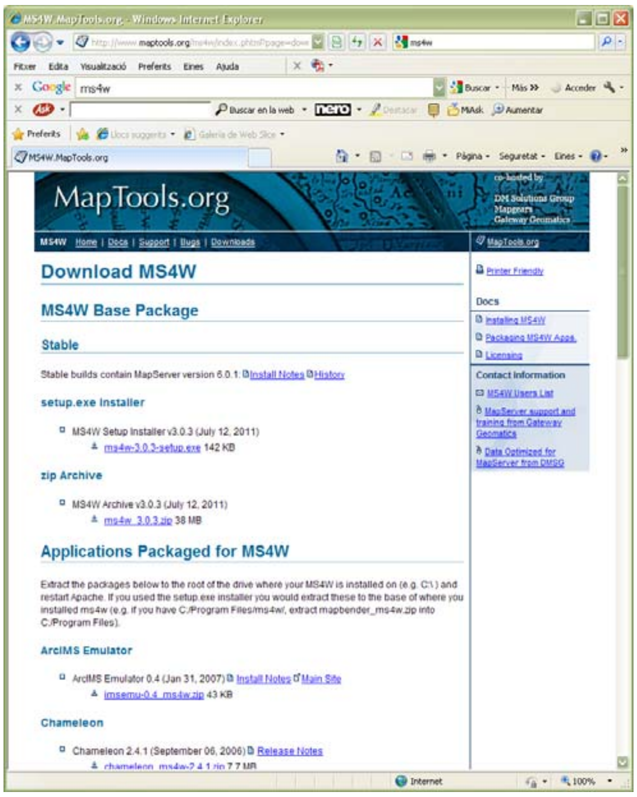
Other documentation of interest:
• MapServer http://www.mapserver.org/
• Introduction to MapServer: http://www.mapserver.org/introduction.html
• Mapfile Reference: http://www.mapserver.org/mapfile/
• Data Input in MapServer: http://www.mapserver.org/input/
• OGC Support in MapServer: http://www.mapserver.org/ogc/

A. Installation of mapserver with MS4W

Download MS4W

1. Download the version available at

http://www.maptools.org/ms4w/index.phtml?page=downloads.html
of the Set up exe installer: ms4w-3.0.3-setup.exe


MS4W.MapTools.org - Windows Internet Explorer
Fitxer Edita Visualització Preferits Eines Ajuda
MapTools.org
co-hosted by
DM Solutions Group
Mapgears
Gateway Geomatics
MS4W Home | Docs | Support | Bugs | Downloads
MapTools.org
Printer Friendly
Docs
Installing MS4W
Packaging MS4W Apps.
Licensing
Contact Information
MS4W Users List
MapServer support and training from Gateway Geomatics
Data Optimized for MapServer from DMSG
Internet
100%

Download MS4W

MS4W Base Package

Stable

Stable builds contain MapServer version 6.0.1: Install Notes History

setup.exe Installer

- MS4W Setup Installer v3.0.3 (July 12, 2011)
  - ms4w-3.0.3-setup.exe 142 KB

zip Archive

- MS4W Archive v3.0.3 (July 12, 2011)
  - ms4w_3.0.3.zip 38 MB

Applications Packaged for MS4W

Extract the packages below to the root of the drive where your MS4W is installed on (e.g. C:\ ) and restart Apache. If you used the setup.exe installer you would extract these to the base of where you installed ms4w (e.g. if you have C:/Program Files/ms4w/, extract mapbender_ms4w.zip into C:/Program Files).

ArcIMS Emulator

- ArcIMS Emulator 0.4 (Jan 31, 2007) Install Notes Main Site
  - imsemu-0.4_ms4w.zip 43 KB

Chameleon

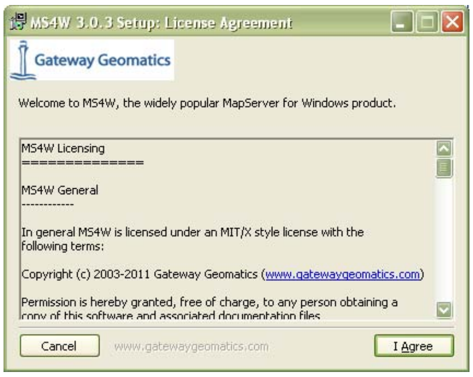
- Chameleon 2.4.1 (September 06, 2006) Release Notes
  - chameleon_ms4w-2.4.1.zip 7.7 MB

Installation of the MS4W

2. To install the MS4W .zip file, use a compression program (e.g. WinZip) to extract the package at the root of a drive, e.g., drive C:. If successful, you should have a new directory named 'ms4w' at the root of the drive you chose (e.g. C:/ms4w).
   Start your MS4W Apache Web Server by running /ms4w/apache-install.bat (at the command line or by double-clicking it). This file installs Apache as a Windows service (called "Apache Web Server") so that it starts whenever your machine is restarted. When executed, a DOS window should pop up with the following message:

```
Installing the Apache MS4W Web Server service
The Apache MS4W Web Server service is successfully installed.
Testing httpd.conf....
Errors reported here must be corrected before the service
can be started.
The Apache MS4W Web Server service is starting.
The Apache MS4W Web Server service was started successfully.
```

This means that Apache is running and installed as a service.
The window “Setup: License Agreement” will appear. Read the license agreement and click I Agree.

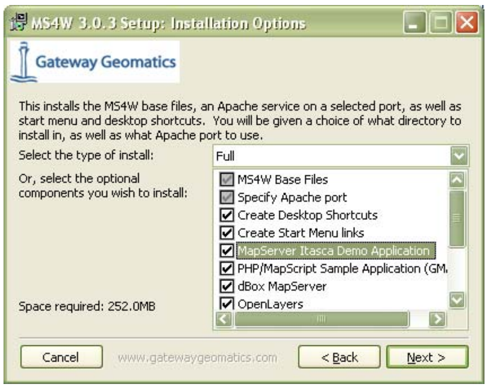

MS4W 3.0.3 Setup: License Agreement

Gateway Geomatics

Welcome to MS4W, the widely popular MapServer for Windows product.

```
MS4W Licensing
==============

MS4W General
-----------

In general MS4W is licensed under an MIT/X style license with the
following terms:

Copyright (c) 2003-2011 Gateway Geomatics (www.gatewaygeomatics.com)

Permission is hereby granted, free of charge, to any person obtaining a
copy of this software and associated documentation files
```

3. An “Installation Option” windows will open. Select your installation options.


MS4W 3.0.3 Setup: Installation Options

Gateway Geomatics

This installs the MS4W base files, an Apache service on a selected port, as well as start menu and desktop shortcuts. You will be given a choice of what directory to install in, as well as what Apache port to use.

Select the type of install: Full

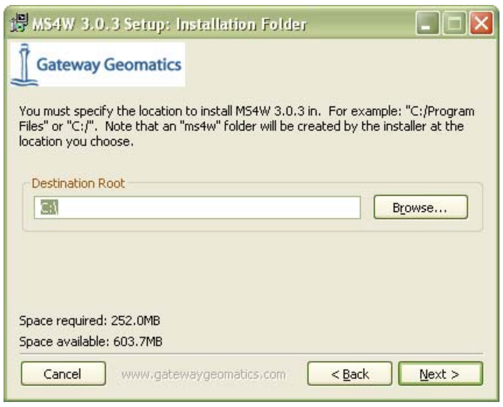
Or, select the optional components you wish to install:

- ✓ MS4W Base Files
- ✓ Specify Apache port
- ✓ Create Desktop Shortcuts
- ✓ Create Start Menu links
- ✓ MapServer Itasca Demo Application
- ✓ PHP/MapScript Sample Application (GM
- ✓ dBox MapServer
- ✓ OpenLayers

Space required: 252.0MB

After selecting your installation options click “Next”.

4. When the “Installation Folder” opens, you must specify the location to install the MS4W. Usually this is set to the root directory (C:). Then click “Next”.


MS4W 3.0.3 Setup: Installation Folder

Gateway Geomatics

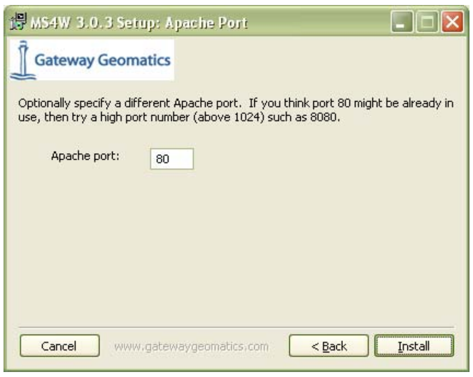
You must specify the location to install MS4W 3.0.3 in. For example: "C:/Program Files" or "C:/". Note that an "ms4w" folder will be created by the installer at the location you choose.

Destination Root: C:\

Space required: 252.0MB

Space available: 603.7MB

5. Set the Apache port. The port you set doesn’t have to be already in use. If you think that the port 80 might be already in use then try a higher port number (above 1024) such as 8081. Click “Install”.


MS4W 3.0.3 Setup: Apache Port

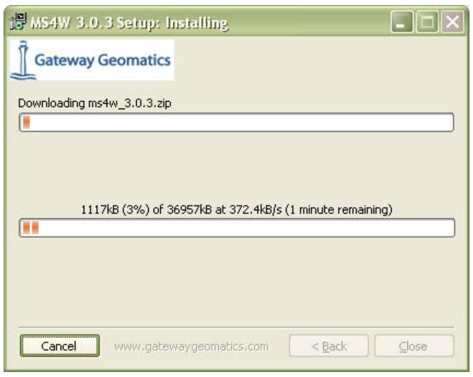
Gateway Geomatics

Optionally specify a different Apache port. If you think port 80 might be already in use, then try a high port number (above 1024) such as 8080.

Apache port: 80

Cancel www.gatewaygeomatics.com < Back Install

A downloading window will show the progress of the installation.

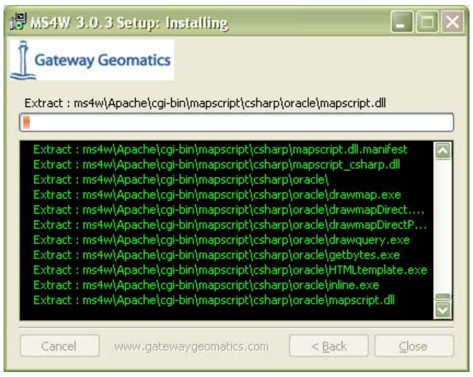

MS4W 3.0.3 Setup: Installing

Gateway Geomatics

Downloading ms4w_3.0.3.zip

1117kB (3%) of 36957kB at 372.4kB/s (1 minute remaining)

Cancel www.gatewaygeomatics.com < Back Close


MS4W 3.0.3 Setup: Installing

Gateway Geomatics

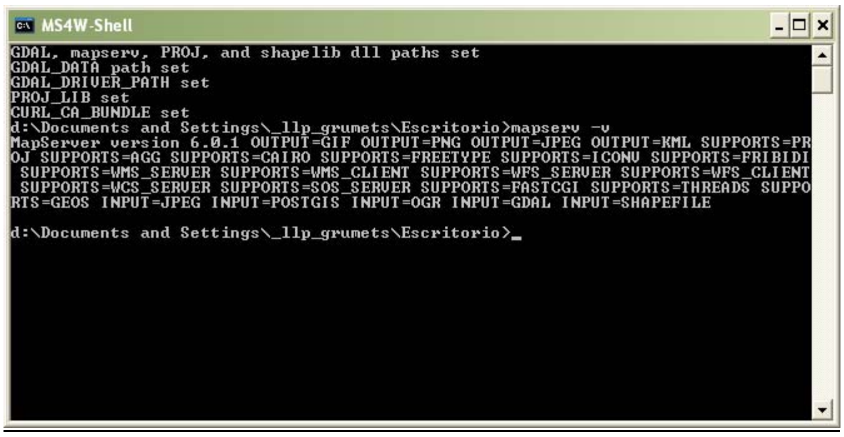
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\mapscript.dll

```
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\mapscript.dll.manifest
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\mapscript_csharp.dll
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\drawmap.exe
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\drawmapDirect....
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\drawmapDirectP...
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\drawquery.exe
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\getbytes.exe
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\HTMLtemplate.exe
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\inline.exe
Extract : ms4w\Apache\cgi-bin\mapscript\csharp\oracle\mapscript.dll
```

Cancel www.gatewaygeomatics.com < Back Close

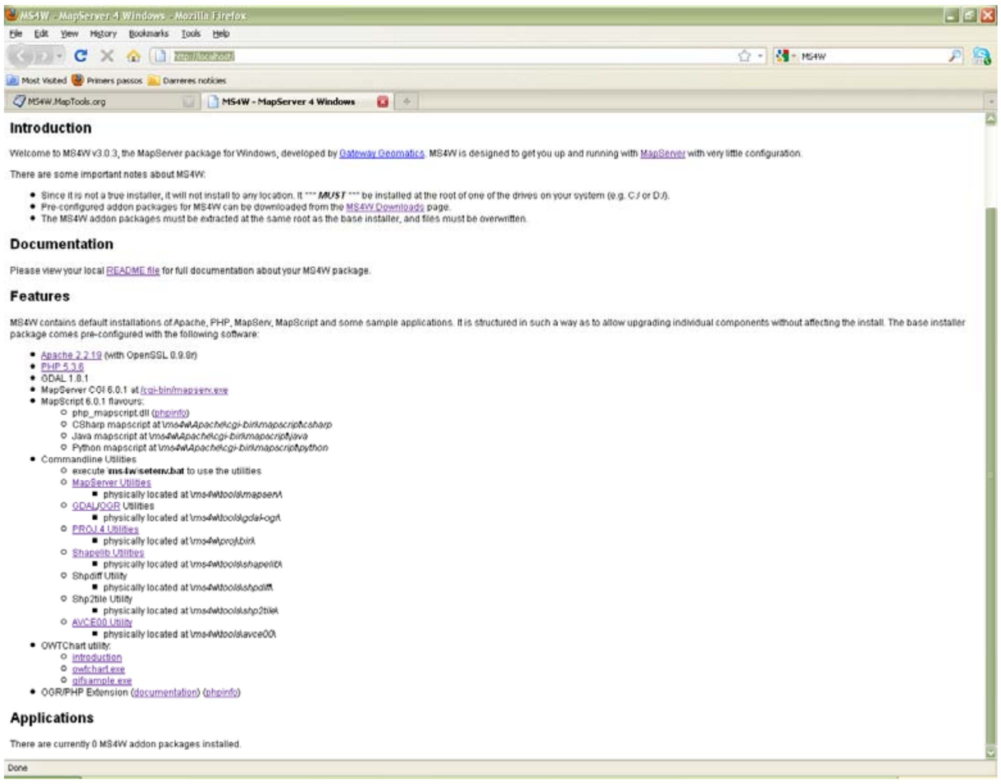
Verification

6. There are two options to verify that the mapserver is correctly installed, one way would be that on the MS4W-Shell DOS window type mapserv –v. The DOS window should show the following message:



```
GDAL, mapserv, PROJ, and shapelib dll paths set
GDAL_DATA path set
GDAL_DRIVER_PATH set
PROJ_LIB set
CURL_CA_BUNDLE set
d:\Documents and Settings\_llp_grumets\Escritorio>mapserv -v
MapServer version 6.0.1 OUTPUT=GIF OUTPUT=PNG OUTPUT=JPEG OUTPUT=KML SUPPORTS=PR
OJ SUPPORTS=AGG SUPPORTS=CAIRO SUPPORTS=FREETYPE SUPPORTS=ICONV SUPPORTS=FRIBIDI
 SUPPORTS=WMS_SERVER SUPPORTS=WMS_CLIENT SUPPORTS=WFS_SERVER SUPPORTS=WFS_CLIENT
 SUPPORTS=WCS_SERVER SUPPORTS=SOS_SERVER SUPPORTS=FASTCGI SUPPORTS=THREADS SUPPO
RTS=GEOS INPUT=JPEG INPUT=POSTGIS INPUT=OGR INPUT=GDAL INPUT=SHAPEFILE

d:\Documents and Settings\_llp_grumets\Escritorio>_
```

Another way to verify if mapserver is installed is by browsing to the localhost, and then the following page should be seen:


MS4W - MapServer 4 Windows - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/
MS4W
Most Visited
Primers passos
Darreres notícies
MS4W.MapTools.org
MS4W - MapServer 4 Windows
Done

Introduction

Welcome to MS4W v3.0.3, the MapServer package for Windows, developed by Gateway Geomatics. MS4W is designed to get you up and running with MapServer with very little configuration.

There are some important notes about MS4W:

- Since it is not a true installer, it will not install to any location. It *** ***MUST*** *** be installed at the root of one of the drives on your system (e.g. C:/ or D:/).
- Pre-configured addon packages for MS4W can be downloaded from the MS4W Downloads page.
- The MS4W addon packages must be extracted at the same root as the base installer, and files must be overwritten.

Documentation

Please view your local README file for full documentation about your MS4W package.

Features

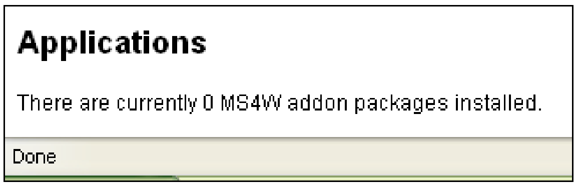
MS4W contains default installations of Apache, PHP, MapServ, MapScript and some sample applications. It is structured in such a way as to allow upgrading individual components without affecting the install. The base installer package comes pre-configured with the following software:

- Apache 2.2.19 (with OpenSSL 0.9.8r)
- PHP 5.3.6
- GDAL 1.8.1
- MapServer CGI 6.0.1 at /cgi-bin/mapserv.exe
- MapScript 6.0.1 flavours:
  - php_mapscript.dll (phpinfo)
  - CSharp mapscript at *\ms4w\Apache\cgi-bin\mapscript\csharp*
  - Java mapscript at *\ms4w\Apache\cgi-bin\mapscript\java*
  - Python mapscript at *\ms4w\Apache\cgi-bin\mapscript\python*
- Commandline Utilities
  - execute '**\ms4w\setenv.bat**' to use the utilities
  - MapServer Utilities
    - physically located at *\ms4w\tools\mapserv\*
  - GDAL/OGR Utilities
    - physically located at *\ms4w\tools\gdal-ogr\*
  - PROJ.4 Utilities
    - physically located at *\ms4w\proj\bin\*
  - Shapelib Utilities
    - physically located at *\ms4w\tools\shapelib\*
  - Shpdiff Utility
    - physically located at *\ms4w\tools\shpdiff\*
  - Shp2tile Utility
    - physically located at *\ms4w\tools\shp2tile\*
  - AVCE00 Utility
    - physically located at *\ms4w\tools\avce00\*
- OWTChart utility:
  - introduction
  - owtchart.exe
  - gifsample.exe
- OGR/PHP Extension (documentation) (phpinfo)

Applications

There are currently 0 MS4W addon packages installed.

If you want to install more programs, they have to be downloaded into the root of the ms4w directory.



Applications

There are currently 0 MS4W addon packages installed.

Not that in both cases there are no addon packages installed into your localhost. If you want to install new applications these should be downloaded from Mapserver web page http://www.mapserver.org/, and save the new application to the root directory were the mapserver is installed, as accept to replace the files you have for the new ones.

B. Installation with OSGeo4W

OSGeo4W is an installer for the Windows operating system that contains several packages related to the Open Source Geospatial Foundation.

Website: http://www.osgeo.org

Introduction

OSGeo4W is a Windows installer that downloads and/or updates MapServer, add-on applications, and also other Open Source geospatial software. The following steps illustrate how to use OSGeo4W:

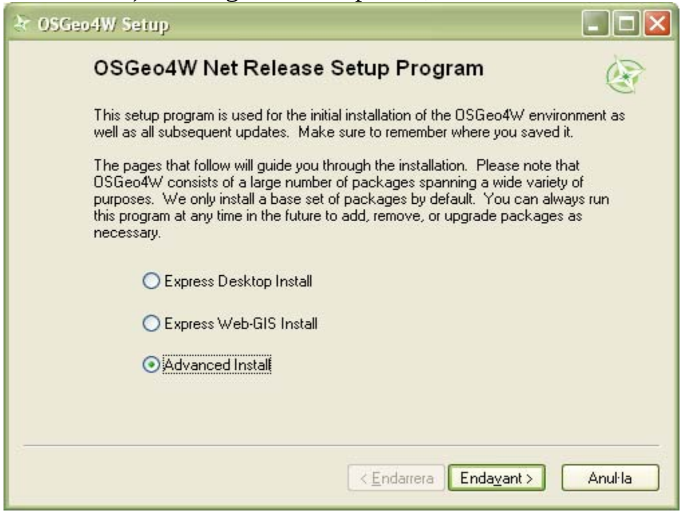
Download OSGeo4W

1. Download OSGeo4W

Download OSGeo4W Installer from:
http://download.osgeo.org/osgeo4w/osgeo4w-setup.exe

2. Start the OSGeo4W Installer

2.1 Execute (double-click) the 'osgeo4w-setup.exe', and a new window should open



OSGeo4W Setup

OSGeo4W Net Release Setup Program

This setup program is used for the initial installation of the OSGeo4W environment as well as all subsequent updates. Make sure to remember where you saved it.

The pages that follow will guide you through the installation. Please note that OSGeo4W consists of a large number of packages spanning a wide variety of purposes. We only install a base set of packages by default. You can always run this program at any time in the future to add, remove, or upgrade packages as necessary.

Express Desktop Install [✓ Advanced Install]

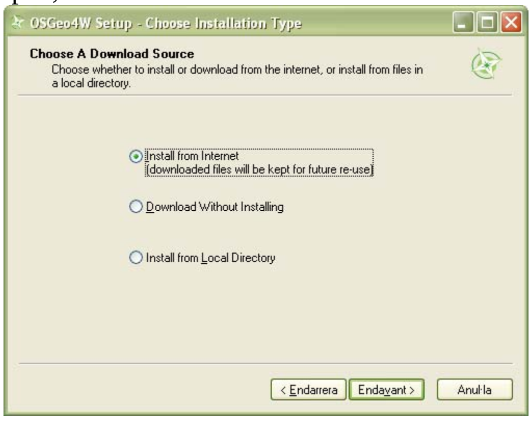
Express Web-GIS Install

Choose an option on this new window.

- The “Express Desktop Install” option contains options for higher-level packages such as MapServer, GRASS and u-Dig.
- The “Express Web-GIS Install” option, can be a basic option
- The “Advanced Install” gives you full access to choosing command line tools and applications for MapServer that are not included in the Express installation.

Once you have one of the options installed, all other options or packages can be updated or downloaded afterwards. It will be explained later on this tutorial.

2.2 On this new window choose the “Advanced Install” option.
A new window will open, choose the Download Source and click “Next”.



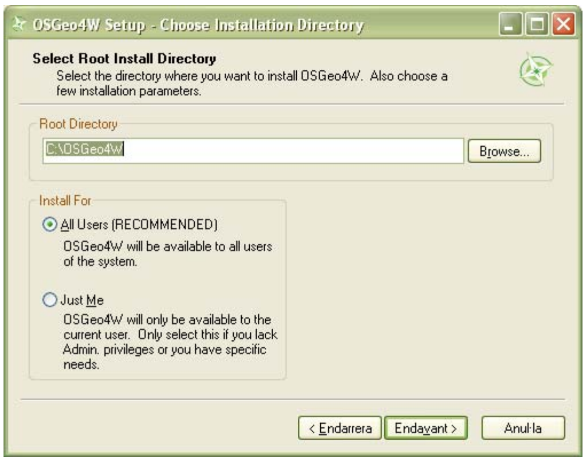
OSGeo4W Setup - Choose Installation Type

Choose A Download Source
Choose whether to install or download from the internet, or install from files in a local directory.

(●) Install from Internet
(downloaded files will be kept for future re-use)
( ) Download Without Installing
( ) Install from Local Directory

< Endarrera | Endavant > | Anul·la

Select the root directory where the program will be installed.



OSGeo4W Setup - Choose Installation Directory

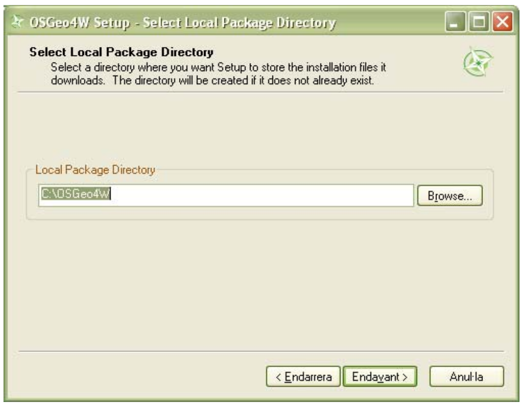
Select Root Install Directory
Select the directory where you want to install OSGeo4W. Also choose a few installation parameters.

Root Directory
C:\OSGeo4W
Browse...

Install For
(●) All Users (RECOMMENDED)
OSGeo4W will be available to all users of the system.
( ) Just Me
OSGeo4W will only be available to the current user. Only select this if you lack Admin. privileges or you have specific needs.

< Endarrera | Endavant > | Anul·la

Next window that appears has it set by default by the Firefox. We change it to the desired one.

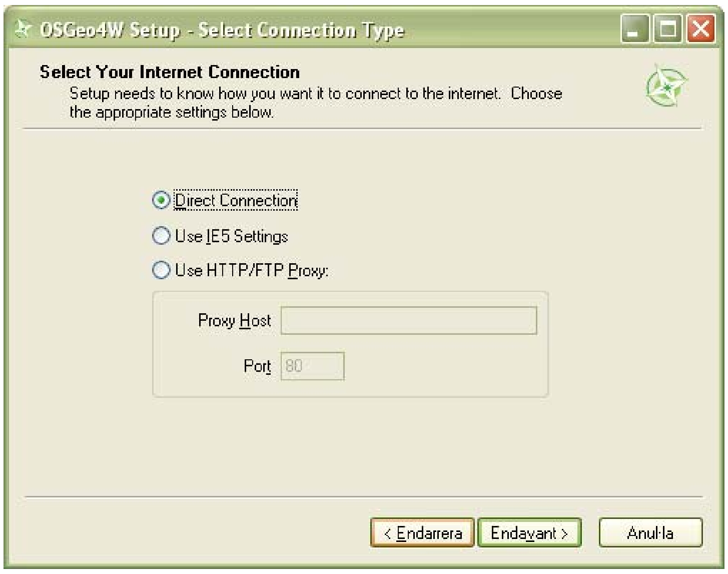

OSGeo4W Setup - Select Local Package Directory

Select Local Package Directory

Select a directory where you want Setup to store the installation files it downloads. The directory will be created if it does not already exist.

Local Package Directory: C:\OSGeo4W

Select your internet connection


OSGeo4W Setup - Select Connection Type

Select Your Internet Connection

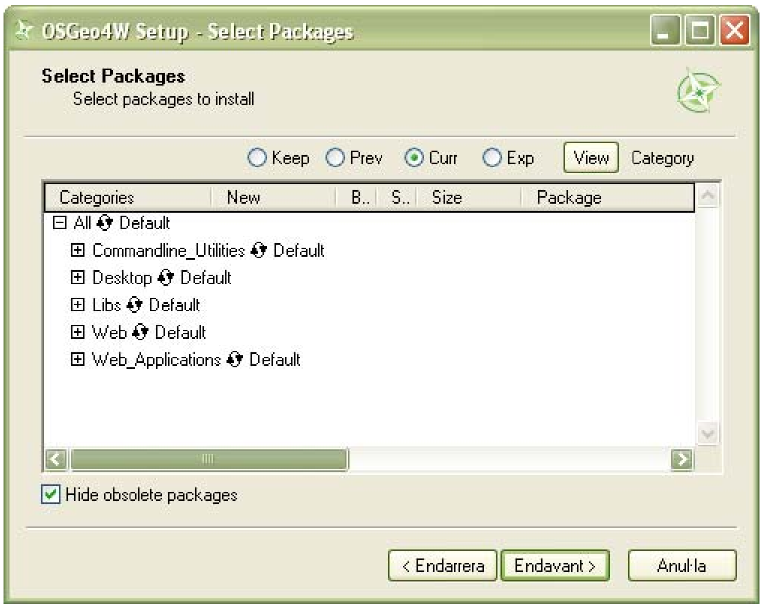
Setup needs to know how you want it to connect to the internet. Choose the appropriate settings below.

[✓ Direct Connection] Use IE5 Settings Use HTTP/FTP Proxy:

Proxy Host:

Port: 80

2.3 Select the packages to install


OSGeo4W Setup - Select Packages

Select Packages
Select packages to install

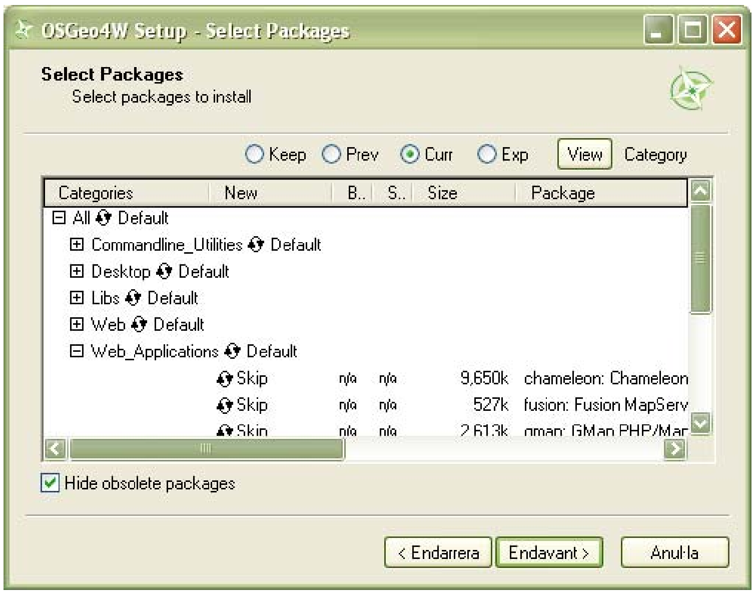
Keep Prev [✓ Curr] Exp View Category

Categories	New	B..	S..	Size	Package
⊟ All ♻ Default					
⊞ Commandline_Utilities ♻ Default					
⊞ Desktop ♻ Default					
⊞ Libs ♻ Default					
⊞ Web ♻ Default					
⊞ Web_Applications ♻ Default					

[✓] Hide obsolete packages

< Endarrera | Endavant > | Anul·la

On this page, you can click into each of the Default texts. For example, when clicking on the “Web Default” text, a drop-down menu will open indicating which Web’s sub-packages are going to be installed by default. You can also click on the “skip” text besides the sub-package name (such as MapServer) to install a package that by default would not be installed.


OSGeo4W Setup - Select Packages

Select Packages
Select packages to install

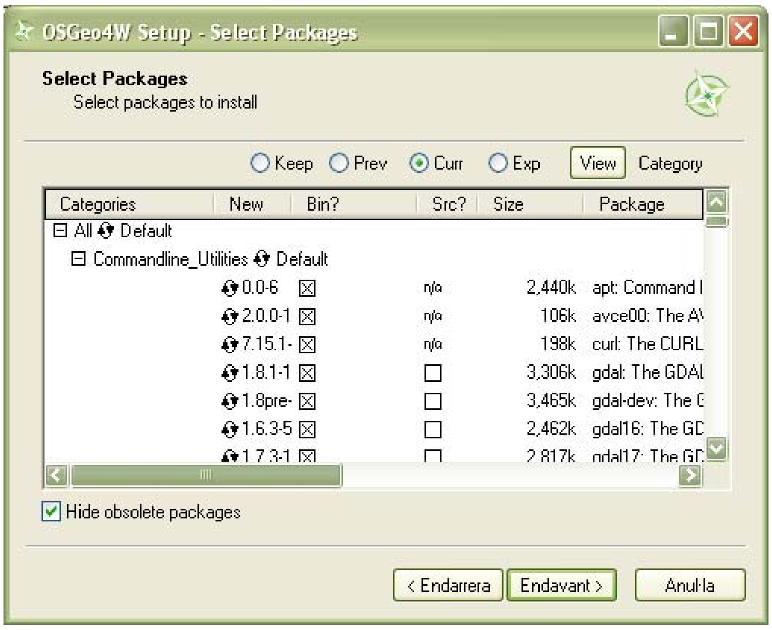
Keep Prev [✓ Curr] Exp View Category

Categories	New	B..	S..	Size	Package
⊟ All ♻ Default					
⊞ Commandline_Utilities ♻ Default					
⊞ Desktop ♻ Default					
⊞ Libs ♻ Default					
⊞ Web ♻ Default					
⊟ Web_Applications ♻ Default					
	♻ Skip	n/a	n/a	9,650k	chameleon: Chameleon
	♻ Skip	n/a	n/a	527k	fusion: Fusion MapServ
	♻ Skip	n/a	n/a	2,613k	gmap: GMap PHP/Map

[✓] Hide obsolete packages

< Endarrera | Endavant > | Anul·la

Note: If the Selected package option is left with the Skip text on, this will mean that the installation of that package will not be done. To select a package, you have to click on the Skip text to obtain:


OSGeo4W Setup - Select Packages

Select Packages
Select packages to install

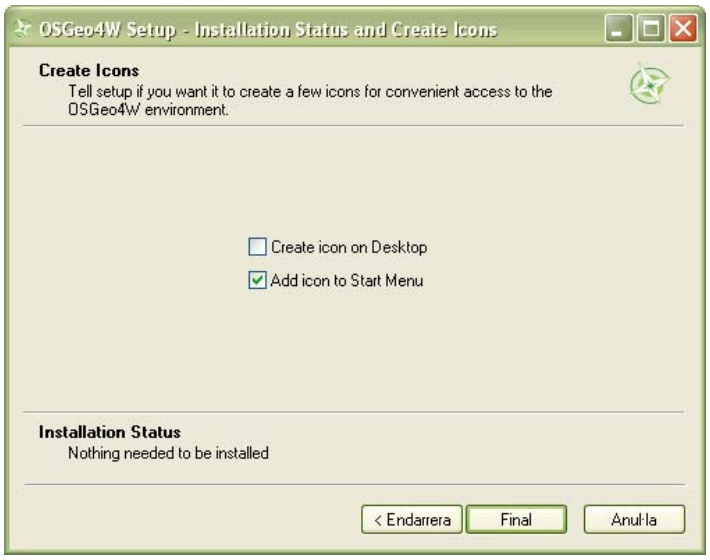
○ Keep ○ Prev [✓ ◉ Curr] ○ Exp View Category

Categories	New	Bin?	Src?	Size	Package
⊟ All Default					
⊟ Commandline_Utilities Default					
	0.0-6	☒	n/a	2,440k	apt: Command l
	2.0.0-1	☒	n/a	106k	avce00: The A\
	7.15.1-	☒	n/a	198k	curl: The CURL
	1.8.1-1	☒	☐	3,306k	gdal: The GDAL
	1.8pre-	☒	☐	3,465k	gdal-dev: The G
	1.6.3-5	☒	☐	2,462k	gdal16: The GD
	1.7.3-1	☒	☐	2,817k	gdal17: The GD

☑ Hide obsolete packages

< Endarrera | Endavant > | Anul·la

You can see that the size of the package vary depending on the option you set up.
After setting up which packages you want to install, a new window opens. On the "Create Icons" window you have to select one or both options offered.


OSGeo4W Setup - Installation Status and Create Icons

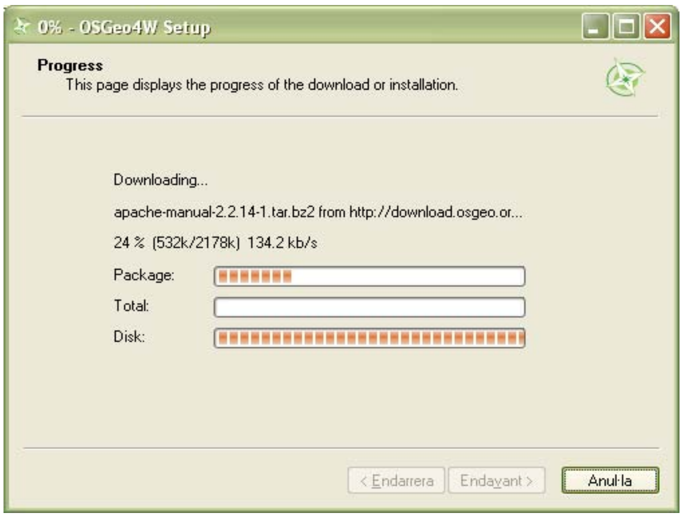
Create Icons
Tell setup if you want it to create a few icons for convenient access to the OSGeo4W environment.

☐ Create icon on Desktop
☑ Add icon to Start Menu

Installation Status
Nothing needed to be installed

< Endarrera | Final | Anul·la

2.4 Download Packages/Installation



0% - OSGeo4W Setup
Progress
This page displays the progress of the download or installation.
Downloading...
apache-manual-2.2.14-1.tar.bz2 from http://download.osgeo.or...
24 % (532k/2178k) 134.2 kb/s
Package:
Total:
Disk:
< Endarrera
Endavant >
Anul·la

The installer should now automatically fetch and install MapServer and all of its libraries from the OSGeo download server.

2.5 Finish Installation

Once downloading and installation from the OSGeo download server is complete, you can now optionally choose to creat shortcuts for OSGeo4W on your desktop, and also in your Start Menu (both are recommended). Click the "Finish" button to finalize the installation.

2.6 To verify that MapServer installation is working

You can verify that MapServer is installed through several ways.

To run the program firstly you need to open the OSGeo4W Shell that you will find on:

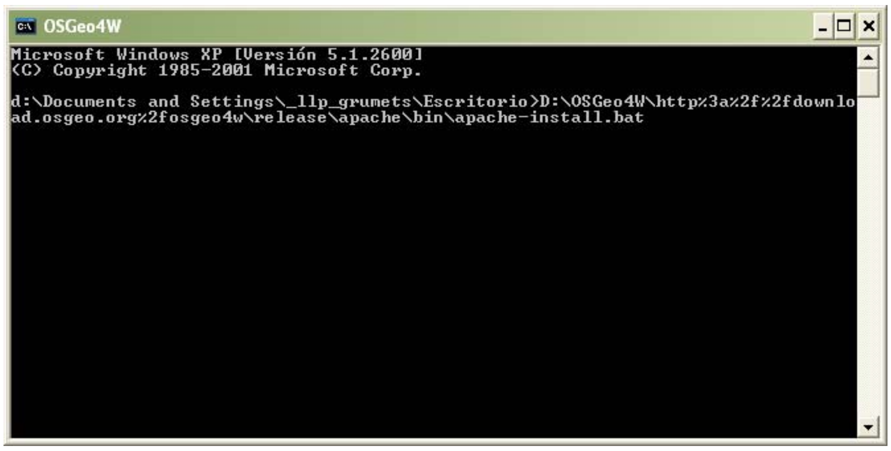
Start/OSGeo4W/OSGeo4W Shell

Or

It can be also available as a shortcut in your computer

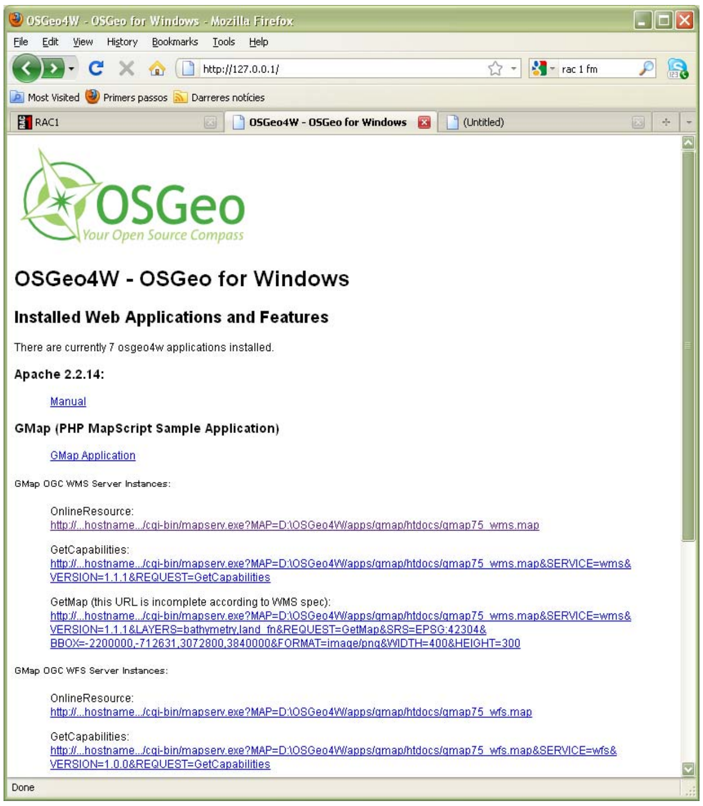
Note: The fist time you try to run the OSGeo4W Shell, you may need to browse to your Shell in order to open it. Comprovar la manera que diu el tutorial descarregat….

Secondly, you need to run the apache-install.bat script to install the Appache Service. This will be ran through the OSGeo4W shell (DOS window)



```
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

d:\Documents and Settings\_llp_grumets\Escritorio>D:\OSGeo4W\http%3a%2f%2fdownlo
ad.osgeo.org%2fosgeo4w\release\apache\bin\apache-install.bat
```

Access to the http://127.0.0.1/


OSGeo4W - OSGeo for Windows - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1/
rac 1 fm
Most Visited Primers passos Darreres notícies
RAC1
OSGeo4W - OSGeo for Windows
(Untitled)
OSGeo
Your Open Source Compass

OSGeo4W - OSGeo for Windows

Installed Web Applications and Features

There are currently 7 osgeo4w applications installed.

Apache 2.2.14:

Manual

GMap (PHP MapScript Sample Application)

GMap Application

GMap OGC WMS Server Instances:

OnlineResource:
http://...hostname.../cgi-bin/mapserv.exe?MAP=D:\OSGeo4W/apps/gmap/htdocs/gmap75_wms.map

GetCapabilities:
http://...hostname.../cgi-bin/mapserv.exe?MAP=D:\OSGeo4W/apps/gmap/htdocs/gmap75_wms.map&SERVICE=wms&
VERSION=1.1.1&REQUEST=GetCapabilities

GetMap (this URL is incomplete according to WMS spec):
http://...hostname.../cgi-bin/mapserv.exe?MAP=D:\OSGeo4W/apps/gmap/htdocs/gmap75_wms.map&SERVICE=wms&
VERSION=1.1.1&LAYERS=bathymetry,land_fn&REQUEST=GetMap&SRS=EPSG:42304&
BBOX=-2200000,-712631,3072800,3840000&FORMAT=image/png&WIDTH=400&HEIGHT=300

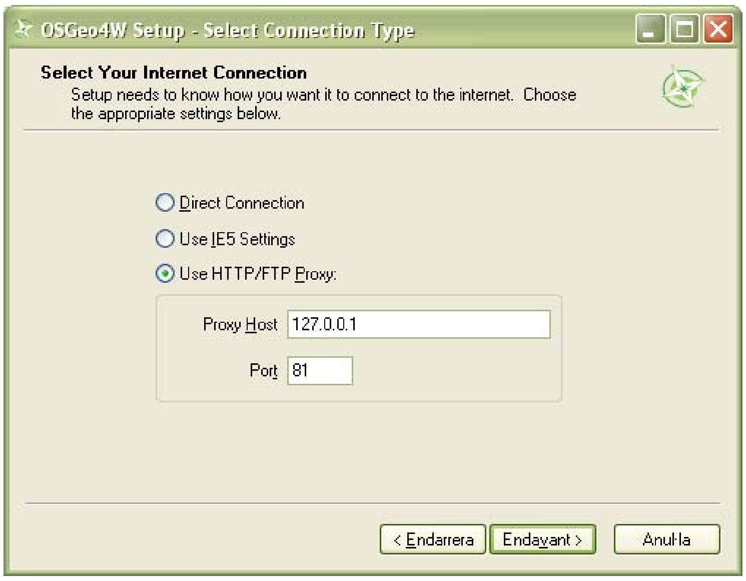
GMap OGC WFS Server Instances:

OnlineResource:
http://...hostname.../cgi-bin/mapserv.exe?MAP=D:\OSGeo4W/apps/gmap/htdocs/gmap75_wfs.map

GetCapabilities:
http://...hostname.../cgi-bin/mapserv.exe?MAP=D:\OSGeo4W/apps/gmap/htdocs/gmap75_wfs.map&SERVICE=wfs&
VERSION=1.0.0&REQUEST=GetCapabilities

Done

Note: Be aware that if you have changed the default port number during the installation process you will then have to access to the http://127.0.0.1:81



Select Your Internet Connection

Setup needs to know how you want it to connect to the internet. Choose the appropriate settings below.

( ) Direct Connection

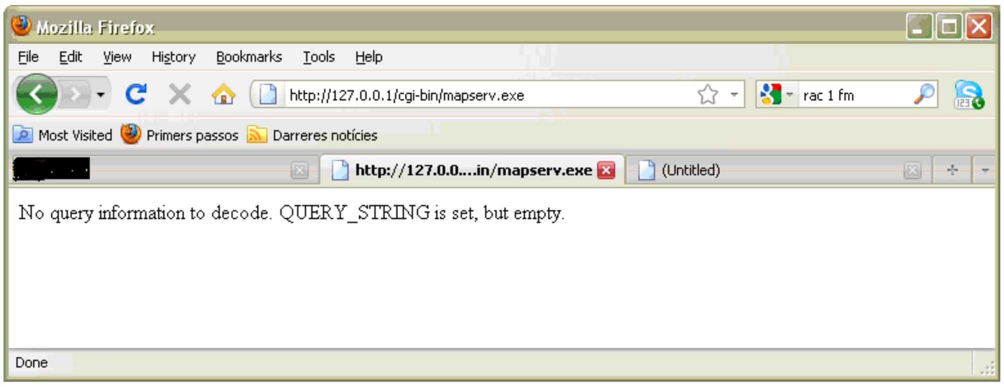
( ) Use IE5 Settings

[✓ Use HTTP/FTP Proxy:]

Proxy Host: 127.0.0.1

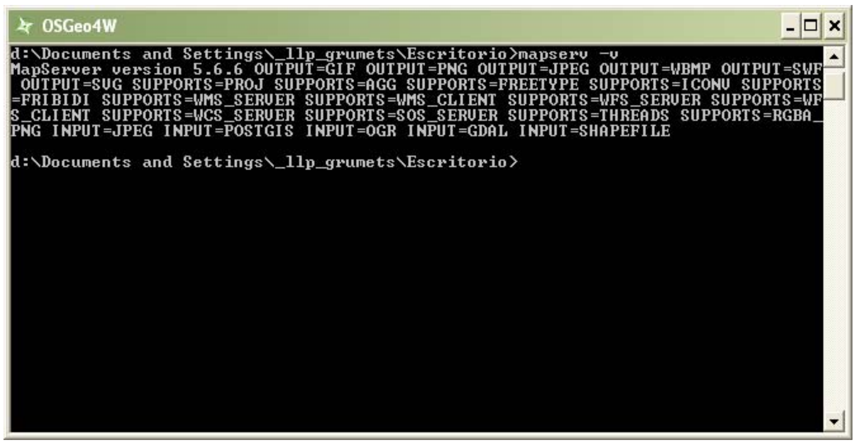
Port: 81

To verify that MapServer is working, go to http://127.0.0.1/cgi-bin/mapserv.exe . This page should appear.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1/cgi-bin/mapserv.exe
rac 1 fm
Most Visited
Primers passos
Darreres notícies
[redacted]
http://127.0.0....in/mapserv.exe
(Untitled)
Done

No query information to decode. QUERY_STRING is set, but empty.

Or by opening the OSGeo4W shell and type mapserv –v



```
d:\Documents and Settings\_llp_grumets\Escritorio>mapserv -v
MapServer version 5.6.6 OUTPUT=GIF OUTPUT=PNG OUTPUT=JPEG OUTPUT=WBMP OUTPUT=SWF
 OUTPUT=SVG SUPPORTS=PROJ SUPPORTS=AGG SUPPORTS=FREETYPE SUPPORTS=ICONV SUPPORTS
=FRIBIDI SUPPORTS=WMS_SERVER SUPPORTS=WMS_CLIENT SUPPORTS=WFS_SERVER SUPPORTS=WF
S_CLIENT SUPPORTS=WCS_SERVER SUPPORTS=SOS_SERVER SUPPORTS=THREADS SUPPORTS=RGBA_
PNG INPUT=JPEG INPUT=POSTGIS INPUT=OGR INPUT=GDAL INPUT=SHAPEFILE

d:\Documents and Settings\_llp_grumets\Escritorio>
```

4.3.1.2. Step 2: Setup a DXF file in MapService

In order to set up a DXF file with mapserver, you first need make sure that the GRASS package has been installed when installing the mapserver. This can be easily done by accessing to the http://127.0.0.1/ (or your mapserver home page), if GRASS is installed, you should see it on the list of packages. Otherwise, you should go to the step 2. Installatiojn, and select the package on the step 2.3.

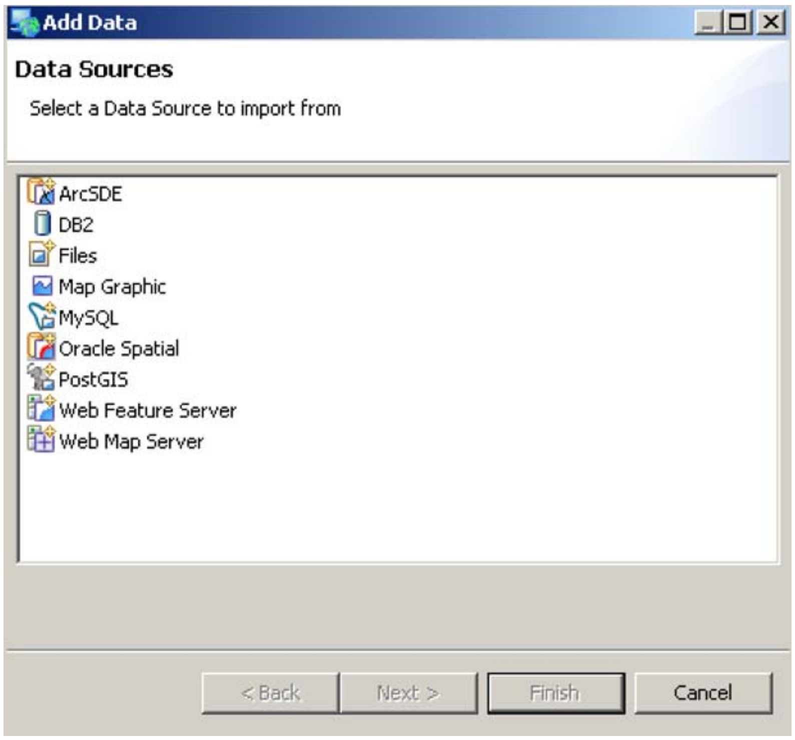
Ones installed open the OSGeo4W Shell
Run the apache-install.bat script in the OSGeo4W Shell

On your "All Programs", under the OSGeo4W it should also have a quick link to GRASS.
To have more information on GRASS, please go to http://mirrors.ibiblio.org/grass/intro/general.php and http://mirrors.ibiblio.org/grass/intro/firsttime.php.

4.3.1.3. Step 3: Test your WFS Service in clients

Accessing the service

1. Start uDig, under the File menu select New > New Map to create a new map.
2. Make sure your local GeoServer is started and ready to go.
3. Select Layer > Add
4. This will bring up a Wizard allowing you to choose which Data source you want to use.. Select **Web Feature Server**.



5. This open a window where you have to insert the URL of a web feature server you want to access. In our case type: http://localhost:8080/geoserver/wfs? and click **Next**.

6. You have now access to all WFS layers available on our Mapserver, press the **Finish** button. The layer will be drawn on your map.

4.3.1.4 Step 6: Optimizing the service performance and maintenance

Information regarding optimization can be found in
http://mapserver.org/optimization/vector.html

4.3.1.5 Step 7: Service errors and troubleshooting

Common errors can be found in http://mapserver.org/errors.html

4.3.2 Offering a data with geoserver WFS

We are going to illustrate in detail and step by step two cases: How to publish a single GeoTIFF image on your GeoServer and How to publish a GeoTIFF mosaic scene on your GeoServer.

GeoServer can be downloaded from http://geoserver.org/display/GEOS/Download. IN this example we are going to use the version 2.1.1.

Before starting, if you do not have the GeoServer set up as a service you need to start it manually. To do so, go to the Start>Programs> GeoServer2.1.1>Start GeoServer. A DOS window will open (and should be kept open) indicating that GeoServer has started and it is running.

Go to the GeoServer Administration link in the Start Menu (GeoServer Web Admin Page) or point your web browser to http://localhost:8080/geoserver. This page is the Web administration interface, which is used to configure all aspects of the GeoServer, and add data to tweaking services settings.

This documentation is directly coming from the GeoServer website
(http://docs.geoserver.org)

4.3.2.1 Case 1: How to setup GeoServer

Step 1: GeoServer Installation

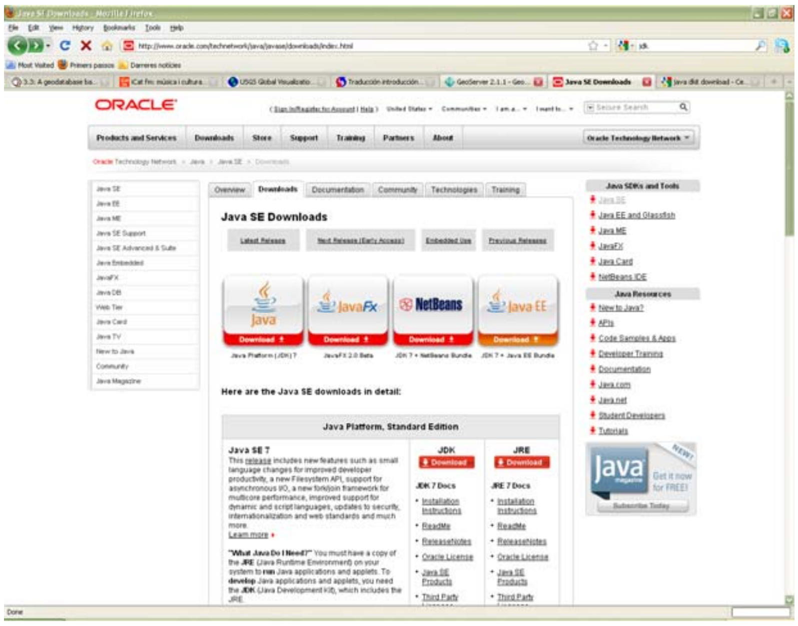
Related documentation (for installation on Linux, Mac, or using WAR file):
http://docs.geoserver.org/stable/en/user/installation/index.html

System requirements
GeoServer can run either on MS Windows, Linux or Mac OS X.
Some general system requirements for the software to run without problems are listed:
Processor: 2 GHz or higher Memory (RAM): 1 GB or higher

Disk Space: 200 MB minimum. However, it is suggested to have a minimum of 1 GB of free disk space. Additional space is required depending on the amount of spatial data that you expect to upload.
Other Software requirements: A Java Runtime Environment (JRE 1.5.0). For server installations, Apache Tomcat can be used instead of Jetty and McKoiDB respectively.

1. Installation

In order to install de GeoServer, firstly the Installation of Java Developement Kit has to be done
Latest version can be found at:
http://www.oracle.com/technetwork/java/javase/downloads/index.html



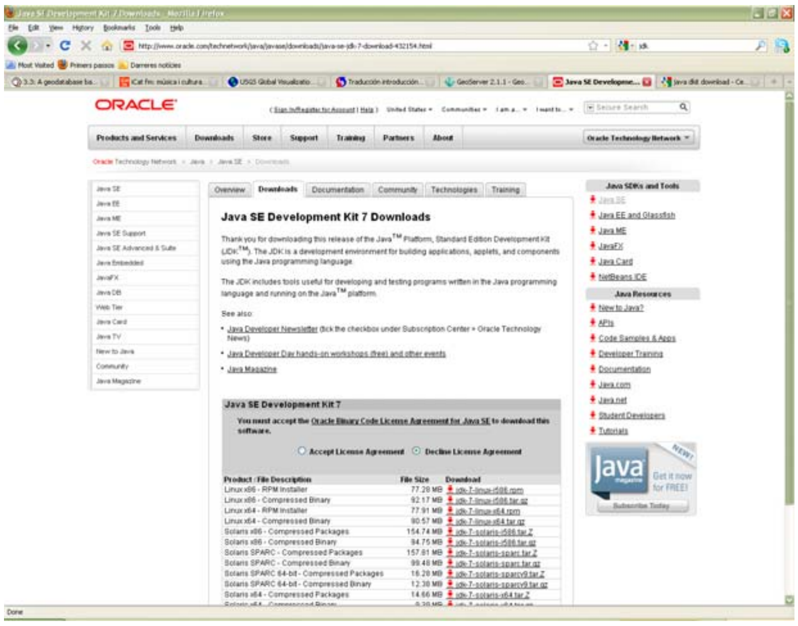
Here are the Java SE downloads in detail:

Java Platform, Standard Edition

Java SE 7
This release includes new features such as small language changes for improved developer productivity, a new Filesystem API, support for asynchronous I/O, a new fork/join framework for multicore performance, improved support for dynamic and script languages, updates to security, internationalization and web standards and much more.
Learn more

"What Java Do I Need?" You must have a copy of the **JRE** (Java Runtime Environment) on your system to **run** Java applications and applets. To **develop** Java applications and applets, you need the **JDK** (Java Development Kit), which includes the JRE.

1. Select Download Java Platform (JDK) 7



Java SE Development Kit 7 Downloads

Thank you for downloading this release of the Java^{TM} Platform, Standard Edition Development Kit (JDK^{TM}). The JDK is a development environment for building applications, applets, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java^{TM} platform.

See also:

- Java Developer Newsletter (tick the checkbox under Subscription Center > Oracle Technology News)
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

Java SE Development Kit 7

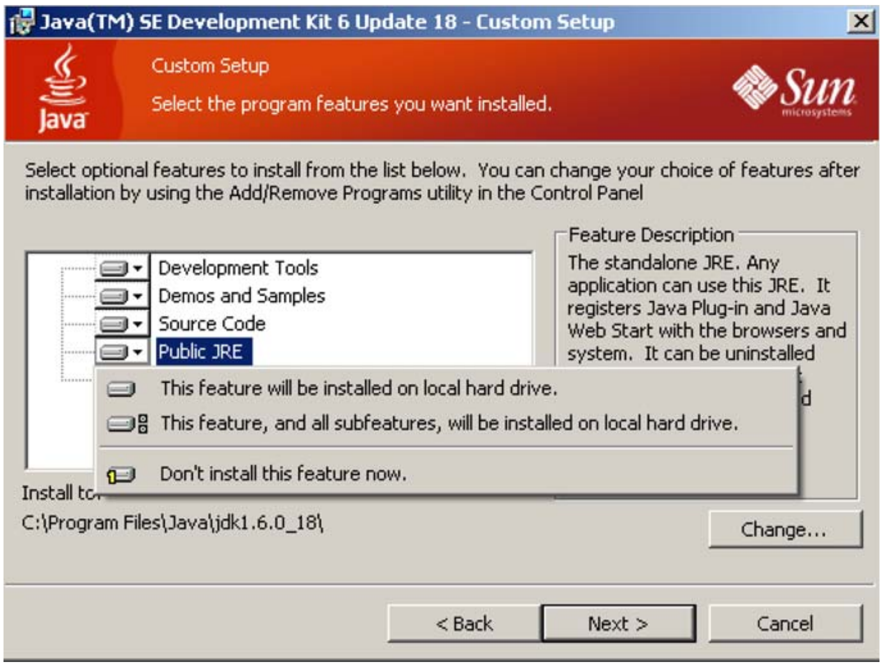
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement [✓ Decline License Agreement]

Product / File Description	File Size	Download
Linux x86 - RPM Installer	77.28 MB	jdk-7-linux-i586.rpm
Linux x86 - Compressed Binary	92.17 MB	jdk-7-linux-i586.tar.gz
Linux x64 - RPM Installer	77.91 MB	jdk-7-linux-x64.rpm
Linux x64 - Compressed Binary	90.57 MB	jdk-7-linux-x64.tar.gz
Solaris x86 - Compressed Packages	154.74 MB	jdk-7-solaris-i586.tar.Z
Solaris x86 - Compressed Binary	94.75 MB	jdk-7-solaris-i586.tar.gz
Solaris SPARC - Compressed Packages	157.81 MB	jdk-7-solaris-sparc.tar.Z
Solaris SPARC - Compressed Binary	99.48 MB	jdk-7-solaris-sparc.tar.gz
Solaris SPARC 64-bit - Compressed Packages	16.28 MB	jdk-7-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit - Compressed Binary	12.38 MB	jdk-7-solaris-sparcv9.tar.gz
Solaris x64 - Compressed Packages	14.66 MB	jdk-7-solaris-x64.tar.Z

Here you need to select your file description, that is to say which java you need according to your system (windows x86,…). When the option to download is selected, a new http page will open for you to register: https://hs-ws1.oracle.com/RegistrationWeb/registration/default/en_US/register-login.jsp?legacy=true&tgtPrd=jdk&prd=registration&uuid=urn%3Ast%3Ac15685dd-d2e0-422f-b848-7f555d27c22b&tgtVer=7

2. Run the executable file you downloaded.
3. Accept the license agreement and click "**Next"**
4. A "Custom Setup" screen will open, and then all optional features can be selected.
5. Click "**Next**".
6. Once the Disable the Public Java Runtime Environment (JRE)


Java(TM) SE Development Kit 6 Update 18 - Custom Setup

Custom Setup

Select the program features you want installed.

Select optional features to install from the list below. You can change your choice of features after installation by using the Add/Remove Programs utility in the Control Panel

Feature Description

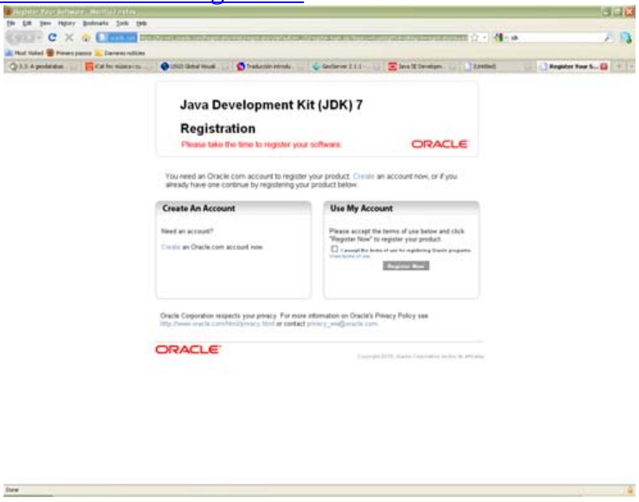
The standalone JRE. Any application can use this JRE. It registers Java Plug-in and Java Web Start with the browsers and system. It can be uninstalled

This feature will be installed on local hard drive.
This feature, and all subfeatures, will be installed on local hard drive.
Don't install this feature now.

Install to:

C:\Program Files\Java\jdk1.6.0_18\

When your installation is completed click “**Finish**”.
Registration of your program can be done at:
https://hs-ws1.oracle.com/RegistrationWeb/registration/default/en_US/register-login.jsp?legacy=true&tgtPrd=jdk&prd=registration&uuid=urn%3Ast%3A666b6296-2f85-4311-b735-7d4d80e32641&tgtVer=7


Java Development Kit (JDK) 7

Registration

Please take the time to register your software.

You need an Oracle.com account to register your product. Create an account now, or if you already have one continue by registering your product below.

Create An Account

Need an account?

Create an Oracle.com account now.

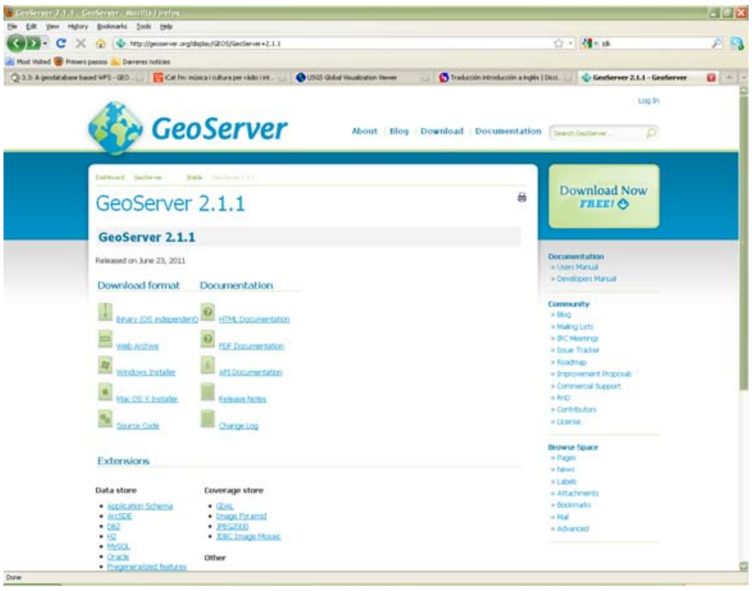
Use My Account

Please accept the terms of use below and click "Register Now" to register your product.

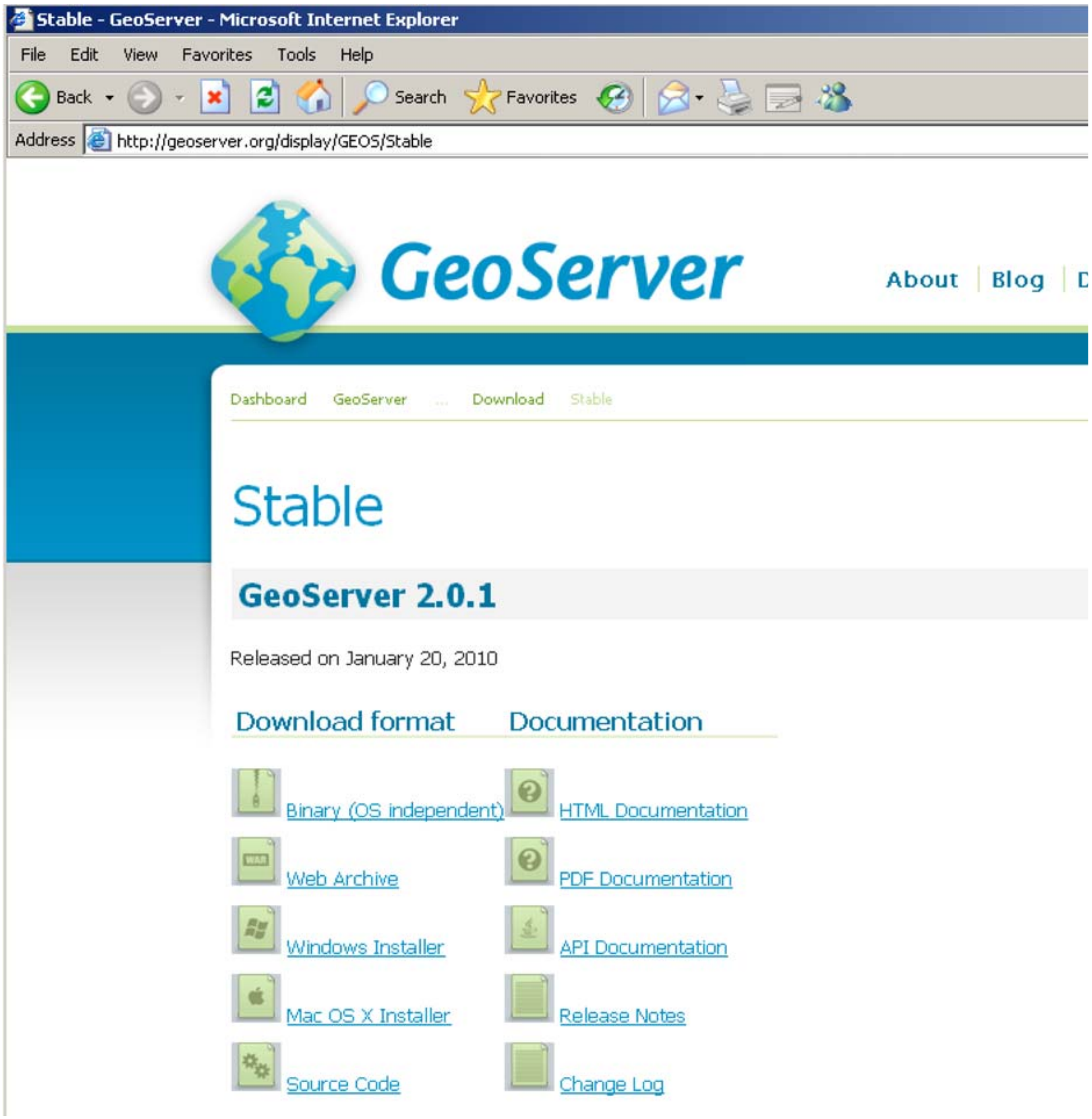
Oracle Corporation respects your privacy. For more information on Oracle's Privacy Policy see http://www.oracle.com/html/privacy.html or contact privacy_ww@oracle.com.

Download GeoServer

1. Now we are ready to download GeoServer
Go to the download page and get the latest Stable release:
http://geoserver.org/display/GEOS/GeoServer+2.1.1



And download the format that best fits you. In our case this will be Windows Installer. It can take some time for the Download to start transferring the geoserver executable


Stable - GeoServer - Microsoft Internet Explorer

Address http://geoserver.org/display/GEOS/Stable

Dashboard GeoServer ... Download Stable

Stable

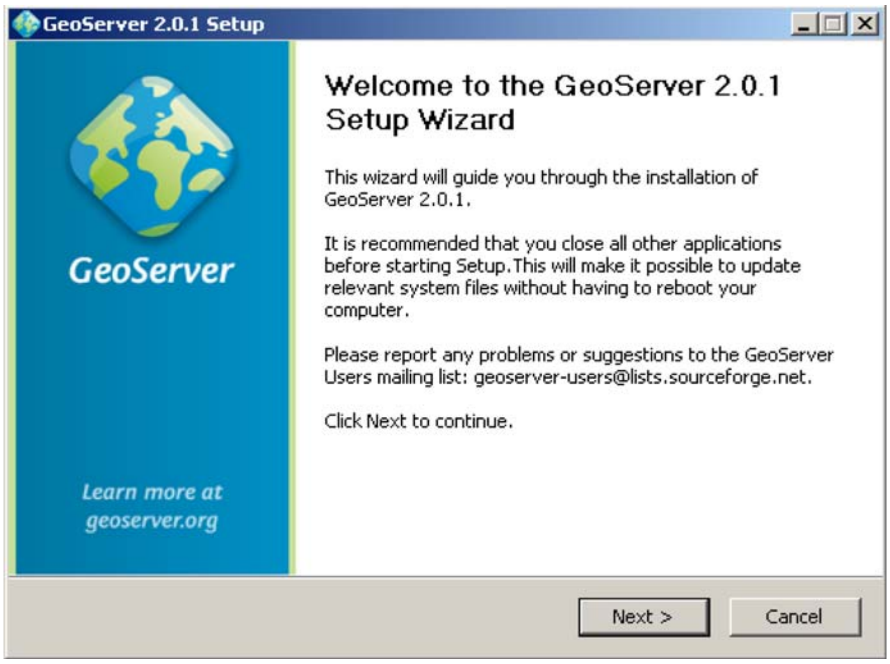
GeoServer 2.0.1

Released on January 20, 2010

Download format	Documentation
Binary (OS independent)	HTML Documentation
Web Archive	PDF Documentation
Windows Installer	API Documentation
Mac OS X Installer	Release Notes
Source Code	Change Log

In our case, download the “Windows Installer”.

2. Run the Installer
    1. To run the installer, double click the geoserver executable that has just been downloaded.


GeoServer 2.0.1 Setup

Welcome to the GeoServer 2.0.1 Setup Wizard

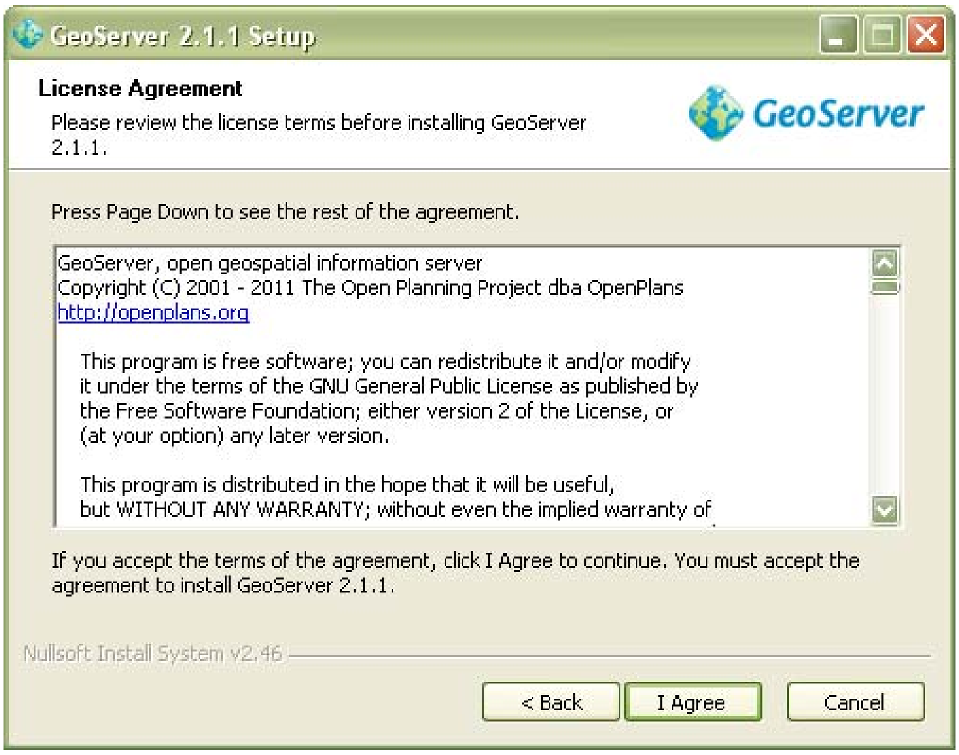
This wizard will guide you through the installation of GeoServer 2.0.1.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Please report any problems or suggestions to the GeoServer Users mailing list: geoserver-users@lists.sourceforge.net.

Click Next to continue.

Click into the "Next" button.


GeoServer 2.1.1 Setup

License Agreement

Please review the license terms before installing GeoServer 2.1.1.

Press Page Down to see the rest of the agreement.

GeoServer, open geospatial information server
Copyright (C) 2001 - 2011 The Open Planning Project dba OpenPlans
http://openplans.org

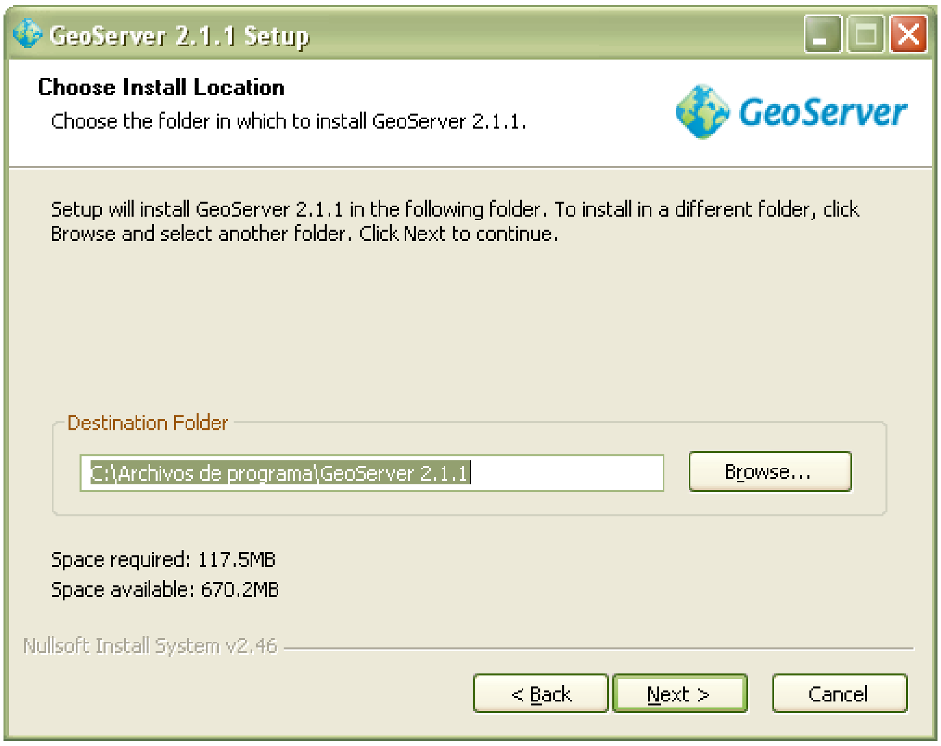
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install GeoServer 2.1.1.

Read the GPL license and click I Agree

Choose your install location and click Next, you can leave it in the default directory or change it into another one you prefer.


GeoServer 2.1.1 Setup

Choose Install Location

Choose the folder in which to install GeoServer 2.1.1.

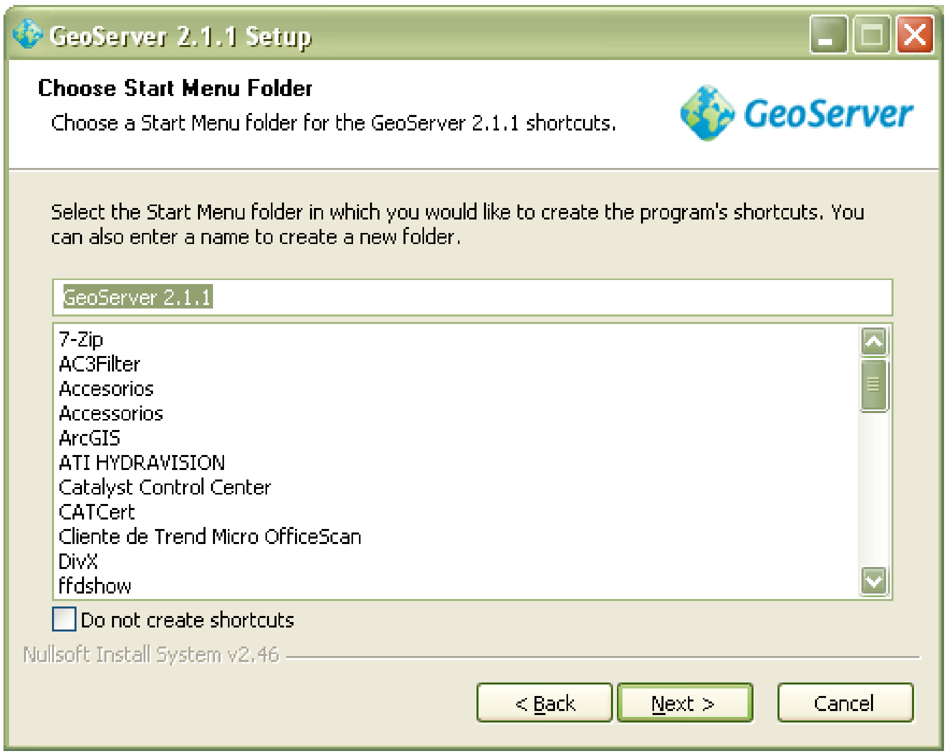
Setup will install GeoServer 2.1.1 in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\Archivos de programa\GeoServer 2.1.1

Space required: 117.5MB
Space available: 670.2MB

Nullsoft Install System v2.46

3. Choose the name of the Start Menu Folder and click "**Next**". Usually this is left to its default value.


GeoServer 2.1.1 Setup

Choose Start Menu Folder

Choose a Start Menu folder for the GeoServer 2.1.1 shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

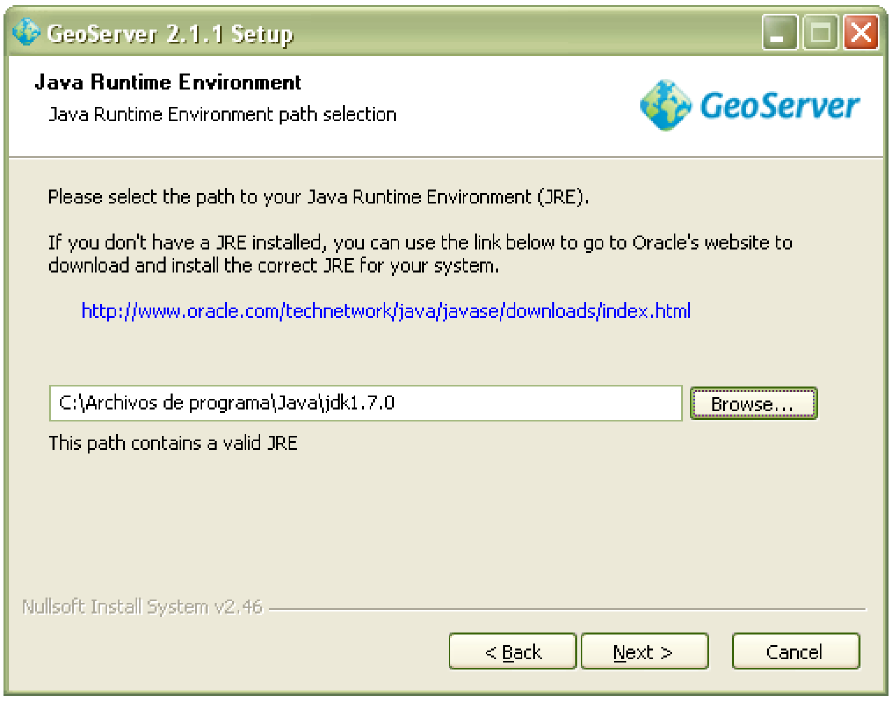
GeoServer 2.1.1

7-Zip
AC3Filter
Accesorios
Accessorios
ArcGIS
ATI HYDRAVISION
Catalyst Control Center
CATCert
Cliente de Trend Micro OfficeScan
DivX
ffdshow

☐ Do not create shortcuts

Nullsoft Install System v2.46

4. Define the path to your JDK install directory by browsing, and click "Next".


GeoServer 2.1.1 Setup

Java Runtime Environment

Java Runtime Environment path selection

Please select the path to your Java Runtime Environment (JRE).

If you don't have a JRE installed, you can use the link below to go to Oracle's website to download and install the correct JRE for your system.

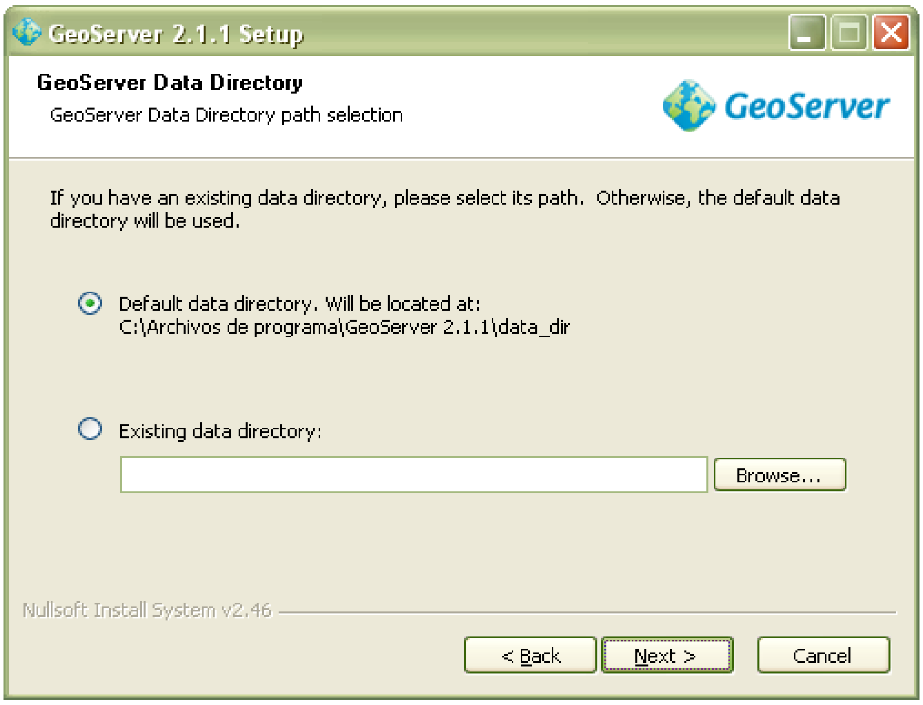
http://www.oracle.com/technetwork/java/javase/downloads/index.html

C:\Archivos de programa\Java\jdk1.7.0

This path contains a valid JRE

Nullsoft Install System v2.46

5. The next window asks where your GeoServer Data Directory is. If this is your first time installing GeoServer, just leave it as Default and click "**Next**". If you have a previous version of the GeoServer installed, and you want to use your existing data directory, enter the path to the existing Data Directory. Click "**Next**".


GeoServer 2.1.1 Setup

GeoServer Data Directory

GeoServer Data Directory path selection

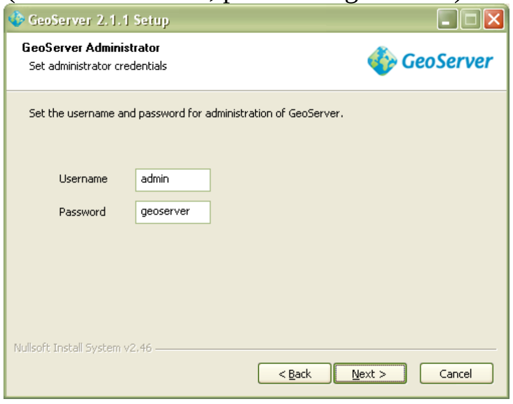
If you have an existing data directory, please select its path. Otherwise, the default data directory will be used.

● Default data directory. Will be located at:
C:\Archivos de programa\GeoServer 2.1.1\data_dir

○ Existing data directory:

Nullsoft Install System v2.46

6. Define user name and Password for an administrator account. You can leave the default values (username: admin, password: geoserver)


GeoServer 2.1.1 Setup

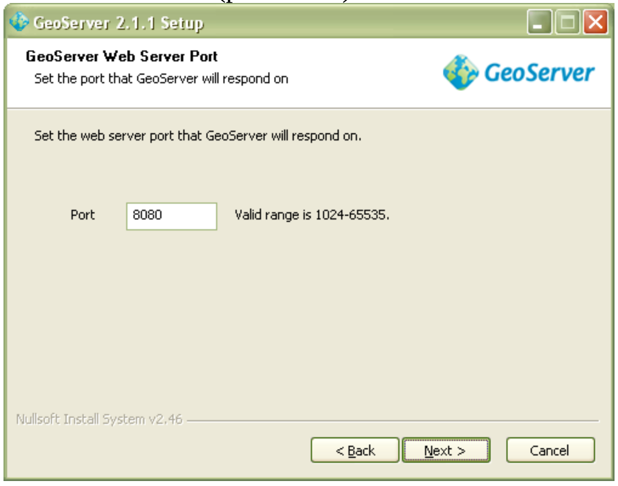
GeoServer Administrator
Set administrator credentials

Set the username and password for administration of GeoServer.

Username: admin
Password: geoserver

Nullsoft Install System v2.46

7. The next window allows you to select the port number on which GeoServer will run. You can leave it as it is (port: 8080).


GeoServer 2.1.1 Setup

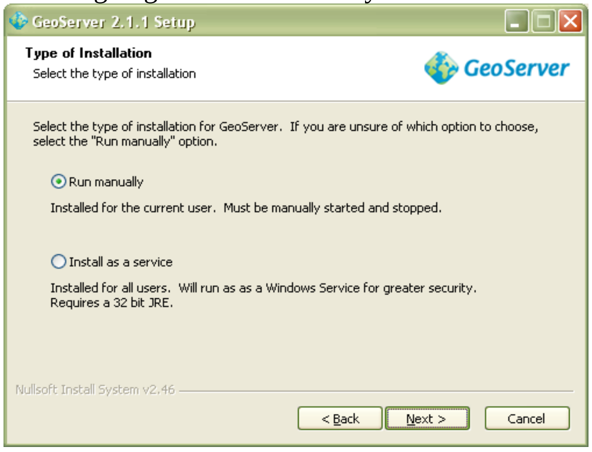
GeoServer Web Server Port
Set the port that GeoServer will respond on

Set the web server port that GeoServer will respond on.

Port: 8080 Valid range is 1024-65535.

Nullsoft Install System v2.46

8. The following window request us which kind of installation are we going to do. In our case we are going to Run it Manually


GeoServer 2.1.1 Setup

Type of Installation
Select the type of installation

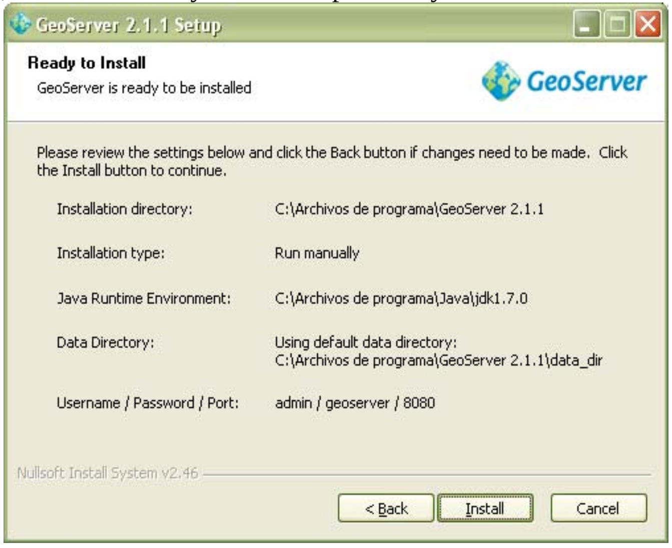
Select the type of installation for GeoServer. If you are unsure of which option to choose, select the "Run manually" option.

[✓ Run manually]
Installed for the current user. Must be manually started and stopped.

Install as a service
Installed for all users. Will run as as a Windows Service for greater security. Requires a 32 bit JRE.

Nullsoft Install System v2.46

9. Finally, a window Ready to Install opens for you to install the GeoServer



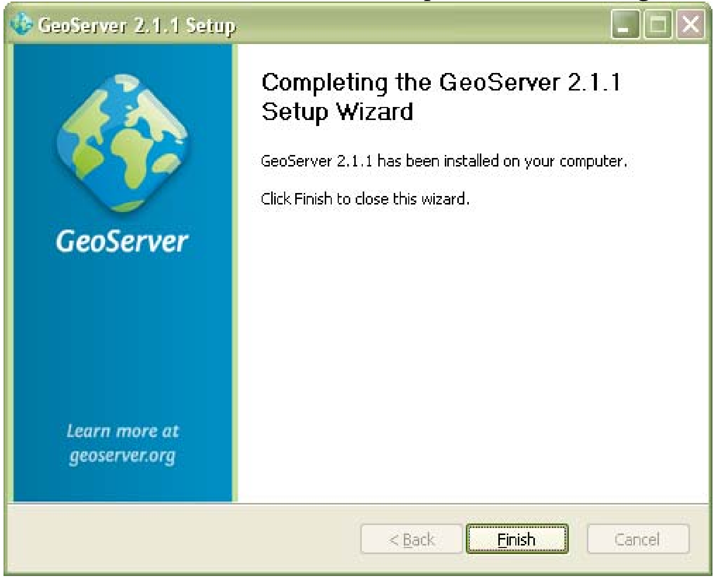
Ready to Install

GeoServer is ready to be installed

Please review the settings below and click the Back button if changes need to be made. Click the Install button to continue.

Installation directory:	C:\Archivos de programa\GeoServer 2.1.1
Installation type:	Run manually
Java Runtime Environment:	C:\Archivos de programa\Java\jdk1.7.0
Data Directory:	Using default data directory: C:\Archivos de programa\GeoServer 2.1.1\data_dir
Username / Password / Port:	admin / geoserver / 8080

10. When the installation of GeoServer is completed the following window appears:



Completing the GeoServer 2.1.1 Setup Wizard

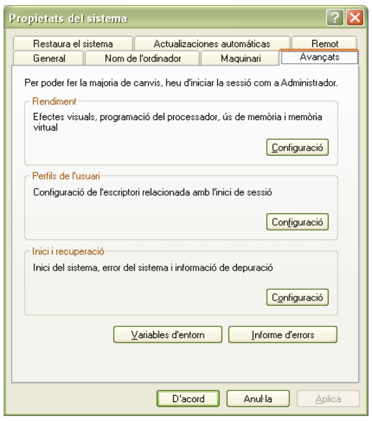
GeoServer 2.1.1 has been installed on your computer.

Click Finish to close this wizard.

At this point you must have Java DK installed and GeoServer Installed. The Geoserver should be seen as a short cut in your Start Menu.

Step 2. Starting GeoServer

1. Before starting GeoServer, the following environment variables have to be defined: JAVA_HOME and GEOSERVER_HOME. These are the paths to the JDK and GeoServer install directories.
2. You have to go to Start/Settings/Control panel/System, and click the "Advanced" tab. Then click on the " Environment vaiables" button.


Propietats del sistema

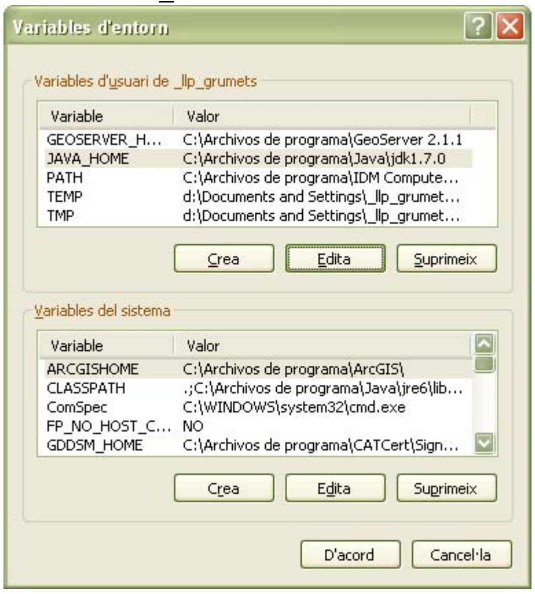
Per poder fer la majoria de canvis, heu d'iniciar la sessió com a Administrador.

Rendiment
Efectes visuals, programació del processador, ús de memòria i memòria virtual

Perfils de l'usuari
Configuració de l'escriptori relacionada amb l'inici de sessió

Inici i recuperació
Inici del sistema, error del sistema i informació de depuració

3. Under “User Variables”, click on New and define your JAVA_HOME variable and then your GEOSERVER_HOME.


Variables d'entorn

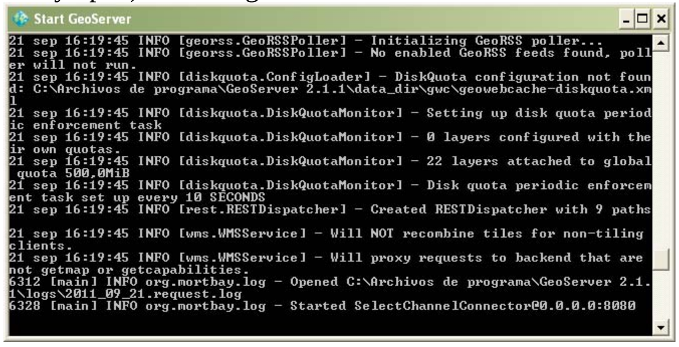
Variables d'usuari de _llp_grumets

Variable	Valor
GEOSERVER_H...	C:\Archivos de programa\GeoServer 2.1.1
JAVA_HOME	C:\Archivos de programa\Java\jdk1.7.0
PATH	C:\Archivos de programa\IDM Compute...
TEMP	d:\Documents and Settings_llp_grumet...
TMP	d:\Documents and Settings_llp_grumet...

Variables del sistema

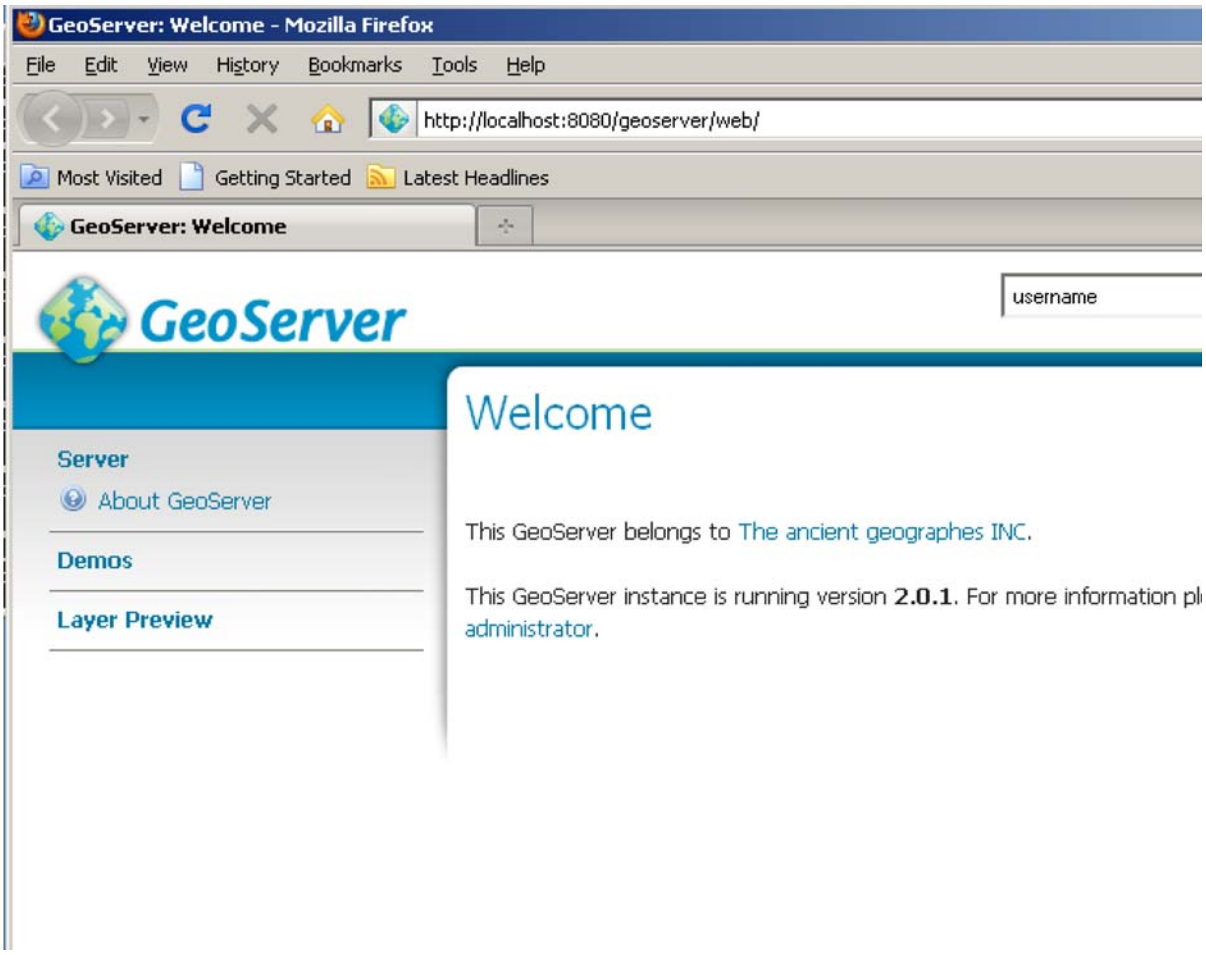
Variable	Valor
ARCGISHOME	C:\Archivos de programa\ArcGIS\
CLASSPATH	.;C:\Archivos de programa\Java\jre6\lib...
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
GDDSM_HOME	C:\Archivos de programa\CATCert\Sign...

4. Go to Start/Programs/GeoServer 2.1.1/Start GeoServer. A DOS window will open (and stay open) indicating that GeoServer has started.


Start GeoServer

```
21 sep 16:19:45 INFO [georss.GeoRSSPoller] - Initializing GeoRSS poller...
21 sep 16:19:45 INFO [georss.GeoRSSPoller] - No enabled GeoRSS feeds found, poll
er will not run.
21 sep 16:19:45 INFO [diskquota.ConfigLoader] - DiskQuota configuration not foun
d: C:\Archivos de programa\GeoServer 2.1.1\data_dir\gwc\geowebcache-diskquota.xm
l
21 sep 16:19:45 INFO [diskquota.DiskQuotaMonitor] - Setting up disk quota period
ic enforcement task
21 sep 16:19:45 INFO [diskquota.DiskQuotaMonitor] - 0 layers configured with the
ir own quotas.
21 sep 16:19:45 INFO [diskquota.DiskQuotaMonitor] - 22 layers attached to global
 quota 500,0MiB
21 sep 16:19:45 INFO [diskquota.DiskQuotaMonitor] - Disk quota periodic enforcem
ent task set up every 10 SECONDS
21 sep 16:19:45 INFO [rest.RESTDispatcher] - Created RESTDispatcher with 9 paths

21 sep 16:19:45 INFO [wms.WMSService] - Will NOT recombine tiles for non-tiling
clients.
21 sep 16:19:45 INFO [wms.WMSService] - Will proxy requests to backend that are
not getmap or getcapabilities.
6312 [main] INFO org.mortbay.log - Opened C:\Archivos de programa\GeoServer 2.1.
1\logs\2011_09_21.request.log
6328 [main] INFO org.mortbay.log - Started SelectChannelConnector@0.0.0.0:8080
```

5. After starting up the program, the administrator interface should be available through the GeoServer Administration link in the Start Menu or by pointing your web browser to http://localhost:8080/geoserver/web/



Welcome

This GeoServer belongs to The ancient geographes INC.

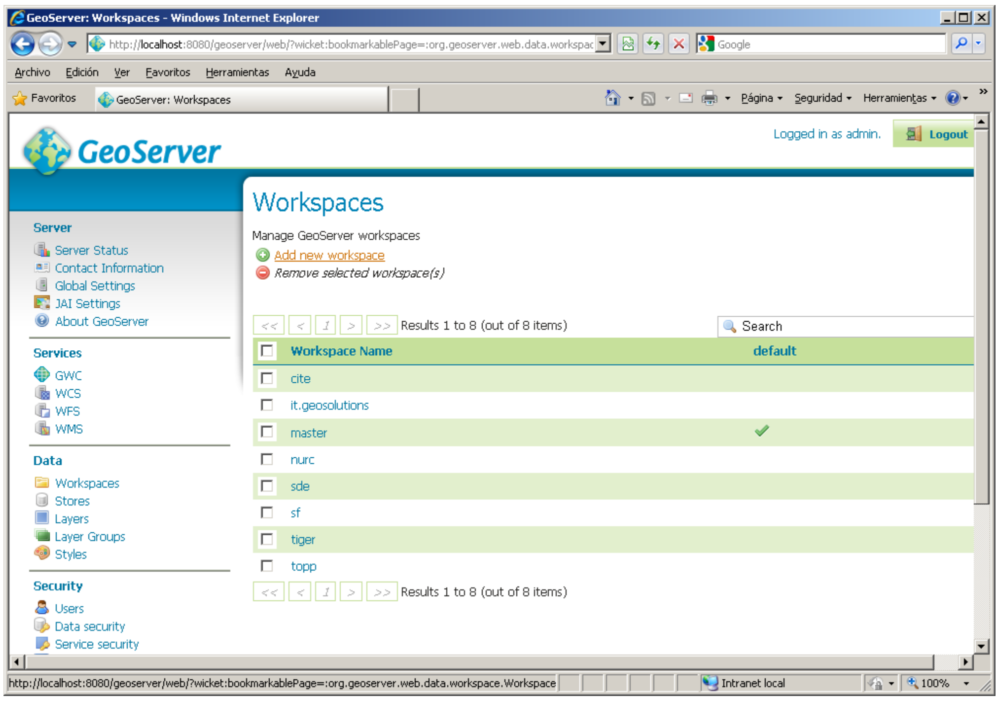
This GeoServer instance is running version **2.0.1**. For more information pl
administrator.

For insights on the Web Administration Interface, please refer to:
http://docs.geoserver.org/stable/en/user/webadmin/index.html

4.3.2.2 Case 2: Offering a shapefile with geoserver WFS

Step 1: Setup a Shapefile
We are going to publish a coverage that is related to the Arctic layers that we previously used in the definition of the data model section.

First, you have to create a new namespace:


GeoServer: Workspaces - Windows Internet Explorer
http://localhost:8080/geoserver/web/?wicket:bookmarkablePage=:org.geoserver.web.data.workspac
Archivo Edición Ver Favoritos Herramientas Ayuda
Favoritos
GeoServer: Workspaces
Página Seguridad Herramientas
GeoServer
Logged in as admin.
Logout
Server
Server Status
Contact Information
Global Settings
JAI Settings
About GeoServer
Services
GWC
WCS
WFS
WMS
Data
Workspaces
Stores
Layers
Layer Groups
Styles
Security
Users
Data security
Service security
http://localhost:8080/geoserver/web/?wicket:bookmarkablePage=:org.geoserver.web.data.workspace.Workspace
Intranet local
100%

Workspaces

Manage GeoServer workspaces

Add new workspace

Remove selected workspace(s)

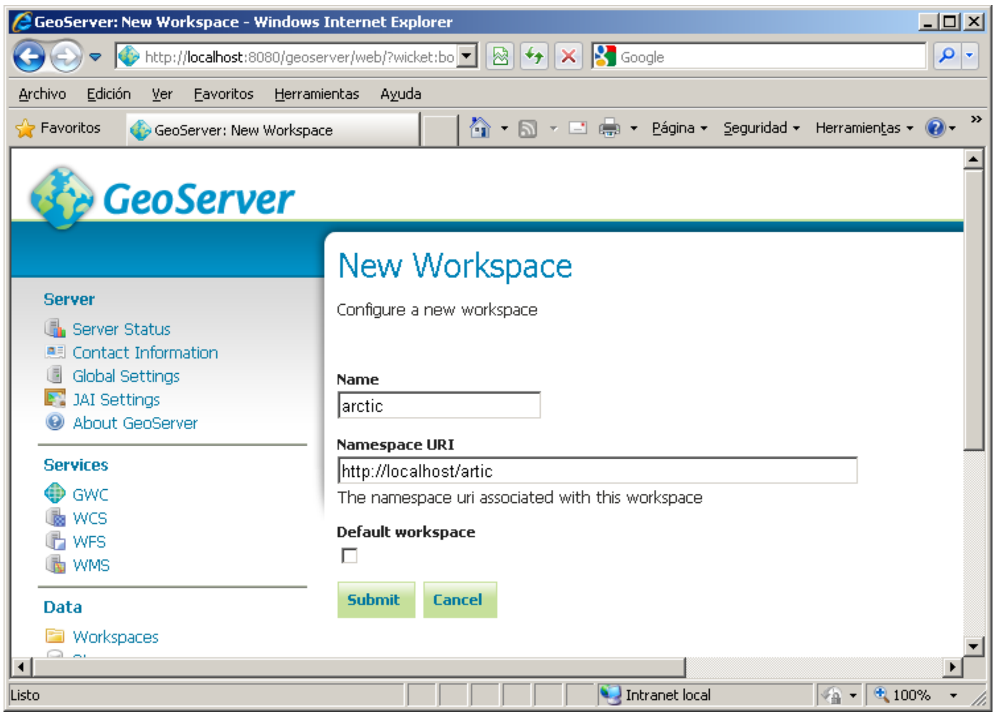
<< < 1 > >> Results 1 to 8 (out of 8 items)

Search

	Workspace Name	default
☐	cite	
☐	it.geosolutions	
☐	master	✓
☐	nurc	
☐	sde	
☐	sf	
☐	tiger	
☐	topp	

<< < 1 > >> Results 1 to 8 (out of 8 items)

We will name it "arctic"



New Workspace

Configure a new workspace

Name

arctic

Namespace URI

http://localhost/artic

The namespace uri associated with this workspace

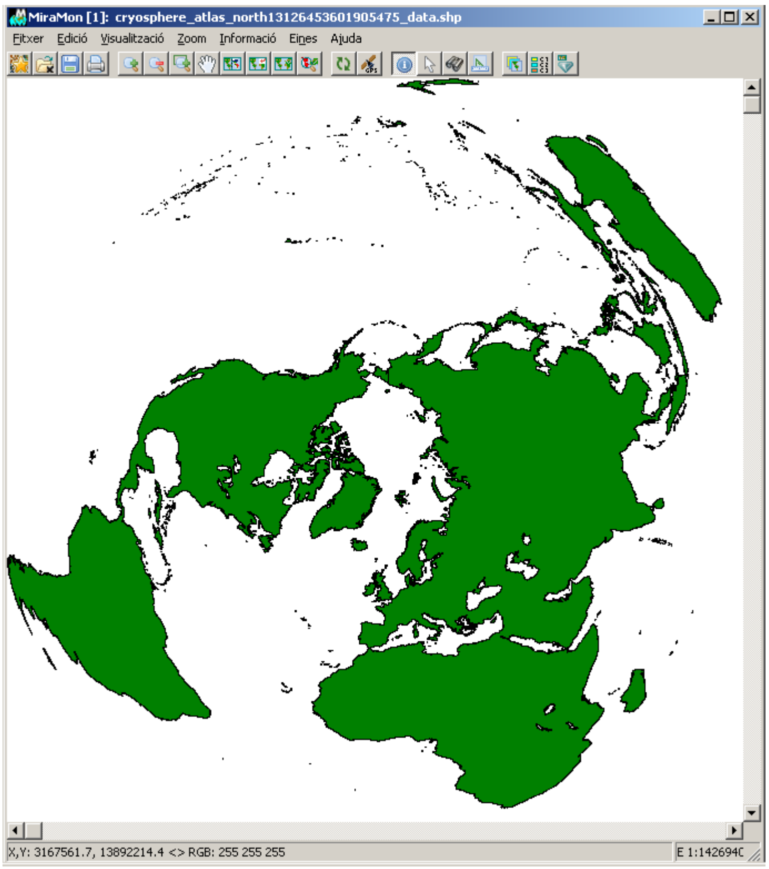
Default workspace

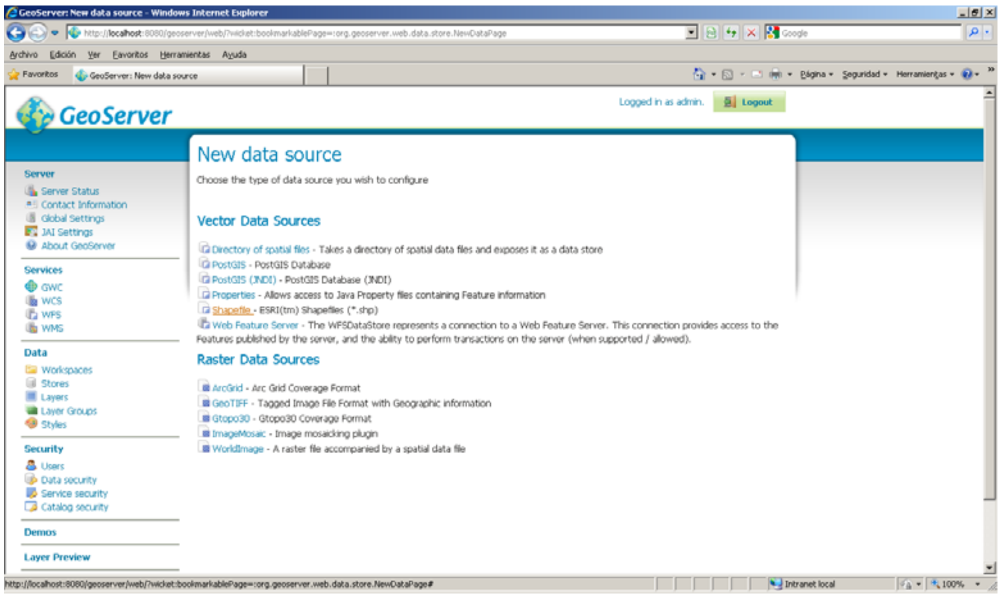
☐

Submit Cancel

Now, we are going to create a data store to include the arctic shape files that we have previously downloaded from the atlas of the cryosphere.

In our case, the idea is to publish the following data:


MiraMon [1]: cryosphere_atlas_north13126453601905475_data.shp
Fitxer Edició Visualització Zoom Informació Eines Ajuda
X,Y: 3167561.7, 13892214.4 <> RGB: 255 255 255
E 1:142694C



New data source

Choose the type of data source you wish to configure

Vector Data Sources

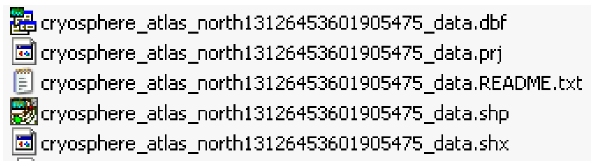
- Directory of spatial files - Takes a directory of spatial data files and exposes it as a data store
- PostGIS - PostGIS Database
- PostGIS (JNDI) - PostGIS Database (JNDI)
- Properties - Allows access to Java Property files containing Feature information
- Shapefile - ESRI(tm) Shapefiles (*.shp)
- Web Feature Server - The WFSDataStore represents a connection to a Web Feature Server. This connection provides access to the Features published by the server, and the ability to perform transactions on the server (when supported / allowed).

Raster Data Sources

- ArcGrid - Arc Grid Coverage Format
- GeoTIFF - Tagged Image File Format with Geographic information
- Gtopo30 - Gtopo30 Coverage Format
- ImageMosaic - Image mosaicking plugin
- WorldImage - A raster file accompanied by a spatial data file

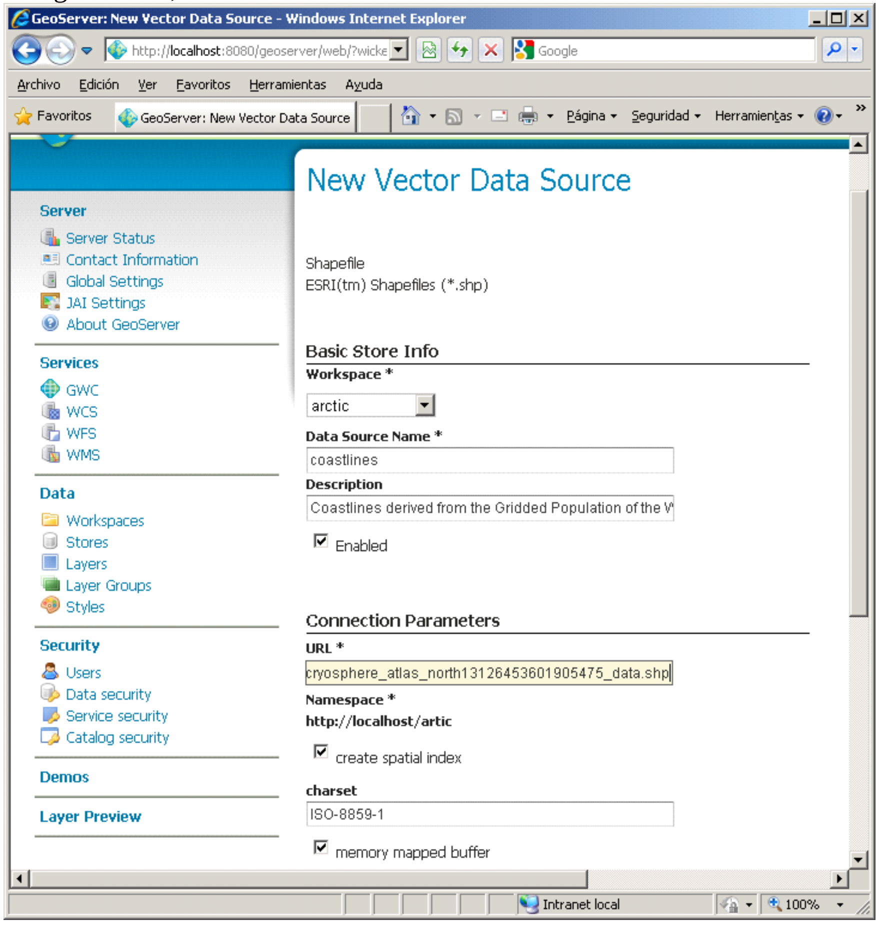
http://localhost:8080/geoserver/web/?wicket:bookmarkablePage=:org.geoserver.web.data.store.NewDataPage#

Before completing the following form, we have to copy the Shapefiles to the right folder. In our case, the default installation folder is called: C:\Program Files\GeoServer 2.0.1\data_dir\data\shapefiles. Remember that you have to copy at least the *.shp. *.dbf, *.shx, and also the *.prj, when available. In our case, these are the files we copied:



cryosphere_atlas_north13126453601905475_data.dbf
cryosphere_atlas_north13126453601905475_data.prj
cryosphere_atlas_north13126453601905475_data.README.txt
cryosphere_atlas_north13126453601905475_data.shp
cryosphere_atlas_north13126453601905475_data.shx

Now, we are able to fill the remaining fields of the form. In the Connection Parameters URL we introduced the full path and name of the file with .shp extension. After completing the form, we can click on "Save"



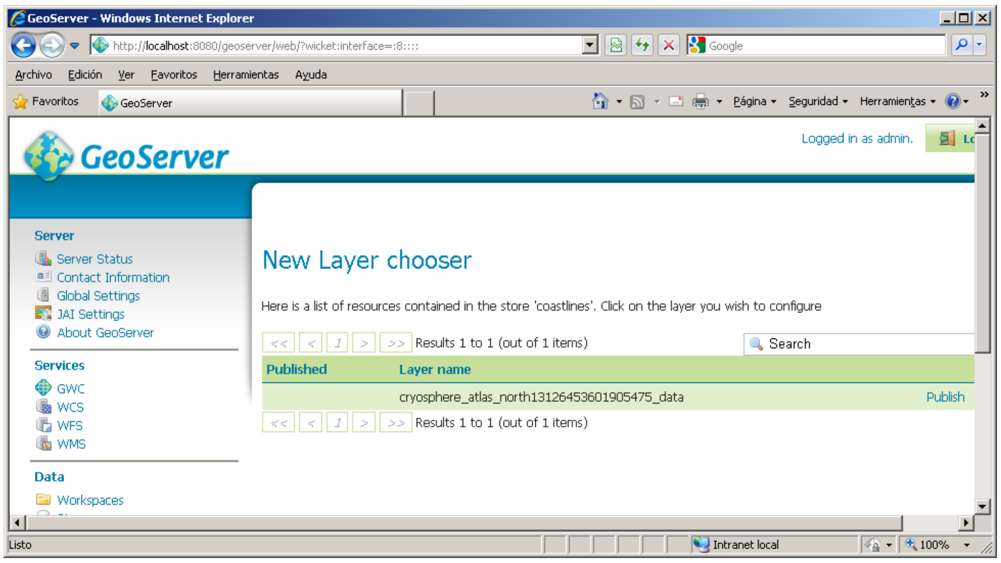
New Vector Data Source

Shapefile
ESRI(tm) Shapefiles (*.shp)

Basic Store Info
Workspace *
arctic
Data Source Name *
coastlines
Description
Coastlines derived from the Gridded Population of the W
☑ Enabled

Connection Parameters
URL *
cryosphere_atlas_north13126453601905475_data.shp
Namespace *
http://localhost/artic
☑ create spatial index
charset
ISO-8859-1
☑ memory mapped buffer

The following screen will appear:



New Layer chooser

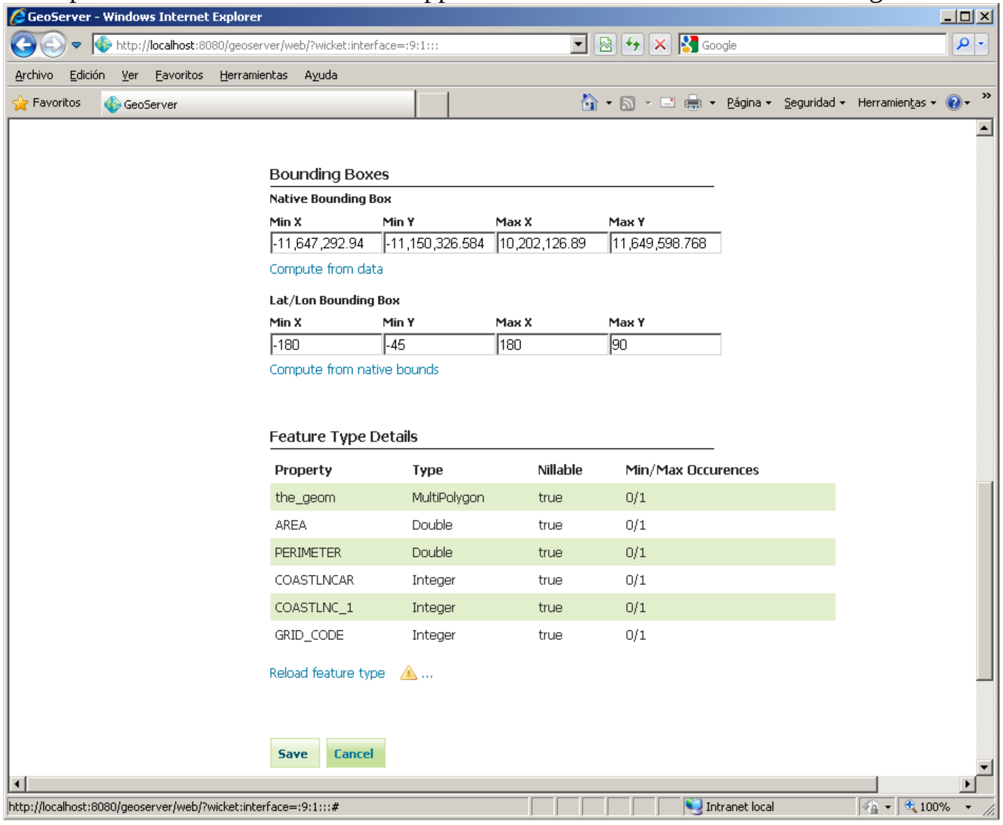
Here is a list of resources contained in the store 'coastlines'. Click on the layer you wish to configure

Results 1 to 1 (out of 1 items)

Published	Layer name	
	cryosphere_atlas_north13126453601905475_data	Publish

Results 1 to 1 (out of 1 items)

Press publish and another form will appear to enter the CRS and the bounding boxes.


GeoServer - Windows Internet Explorer
http://localhost:8080/geoserver/web/?wicket:interface=:9:1:::
Archivo Edición Ver Favoritos Herramientas Ayuda
Favoritos GeoServer
http://localhost:8080/geoserver/web/?wicket:interface=:9:1:::#
Intranet local
100%

Bounding Boxes

Native Bounding Box

Min X	Min Y	Max X	Max Y
-11,647,292.94	-11,150,326.584	10,202,126.89	11,649,598.768

Compute from data

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
-180	-45	180	90

Compute from native bounds

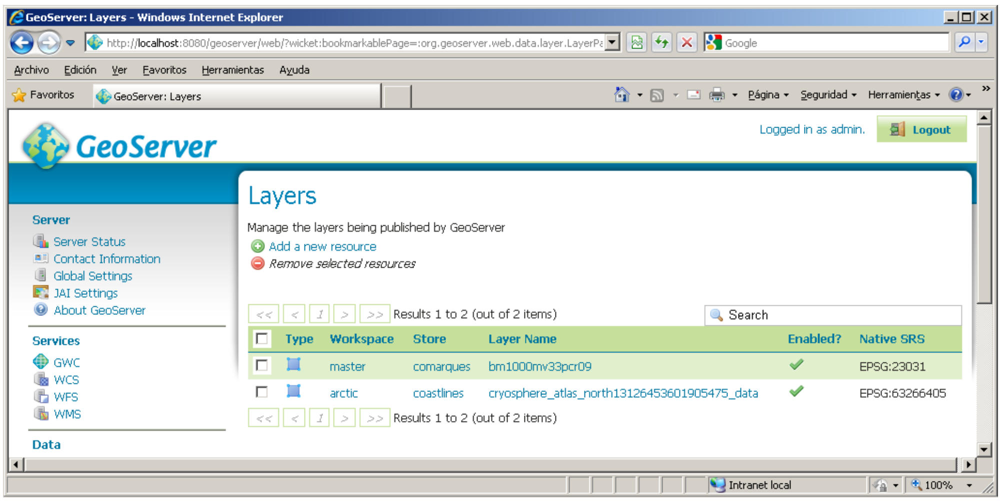
Feature Type Details

Property	Type	Nillable	Min/Max Occurences
the_geom	MultiPolygon	true	0/1
AREA	Double	true	0/1
PERIMETER	Double	true	0/1
COASTLNCAR	Integer	true	0/1
COASTLNC_1	Integer	true	0/1
GRID_CODE	Integer	true	0/1

Reload feature type ⚠ ...

Save Cancel

Once you press “Save”, you will see:


GeoServer: Layers - Windows Internet Explorer

http://localhost:8080/geoserver/web/?wicket:bookmarkablePage=:org.geoserver.web.data.layer.LayerPa

Archivo Edición Ver Favoritos Herramientas Ayuda

Favoritos | GeoServer: Layers

GeoServer — Logged in as admin. Logout

Server: Server Status, Contact Information, Global Settings, JAI Settings, About GeoServer

Services: GWC, WCS, WFS, WMS

Data

Layers

Manage the layers being published by GeoServer

Add a new resource

Remove selected resources

Results 1 to 2 (out of 2 items)

	Type	Workspace	Store	Layer Name	Enabled?	Native SRS
☐	▦	master	comarques	bm1000mv33pcr09	✓	EPSG:23031
☐	▦	arctic	coastlines	cryosphere_atlas_north13126453601905475_data	✓	EPSG:63266405

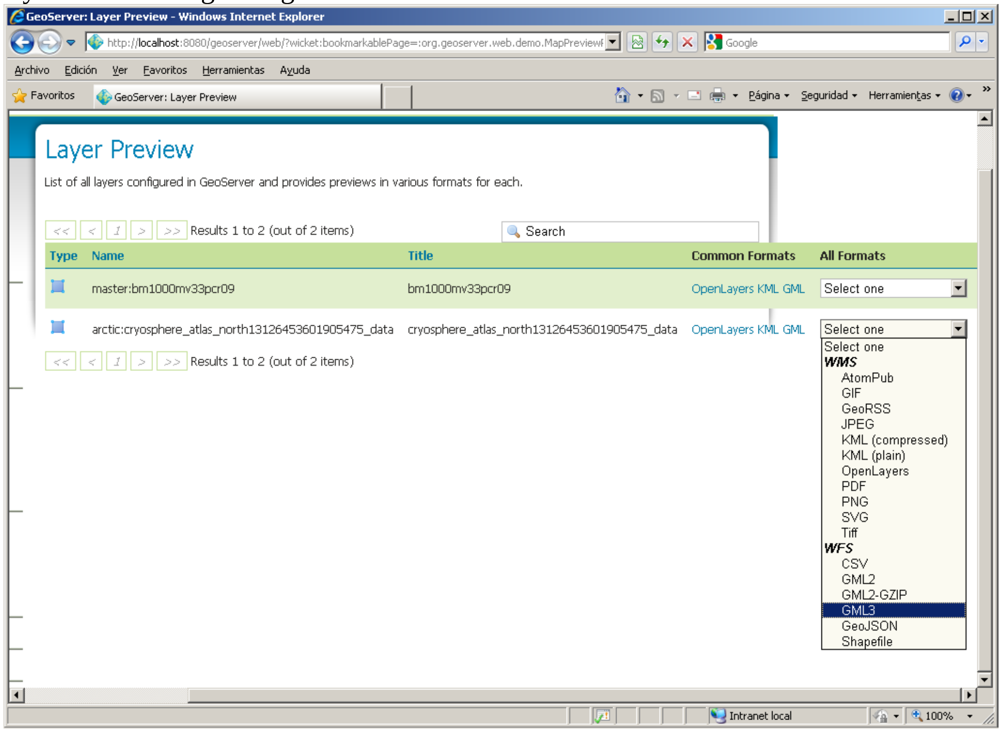
Results 1 to 2 (out of 2 items)

Intranet local | 100%

In theory, your data is ready to be seen in your WFS service.

Step 2: Test your data

Testing the layer in GeoServer is very simple, because it incorporates tools to do that. In the Layer preview section, it is possible to test the WFS service by selecting the right layer and choosing the right service and format:


GeoServer: Layer Preview - Windows Internet Explorer

http://localhost:8080/geoserver/web/?wicket:bookmarkablePage=:org.geoserver.web.demo.MapPreviewP

Archivo Edición Ver Favoritos Herramientas Ayuda

Favoritos | GeoServer: Layer Preview

Layer Preview

List of all layers configured in GeoServer and provides previews in various formats for each.

Results 1 to 2 (out of 2 items)

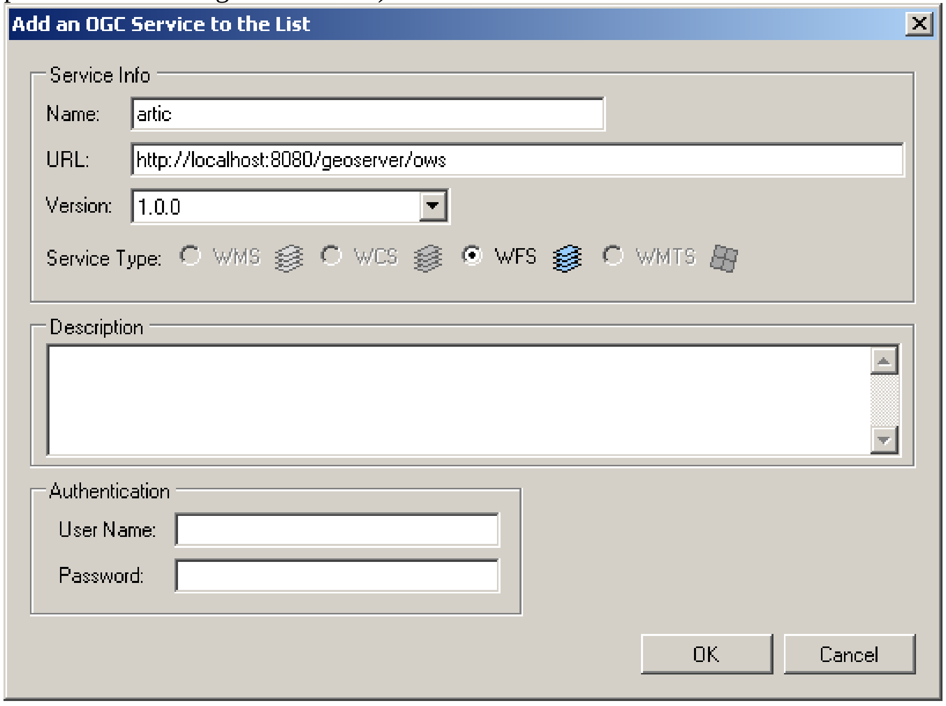
Type	Name	Title	Common Formats	All Formats
▦	master:bm1000mv33pcr09	bm1000mv33pcr09	OpenLayers KML GML	Select one
▦	arctic:cryosphere_atlas_north13126453601905475_data	cryosphere_atlas_north13126453601905475_data	OpenLayers KML GML	Select one

Results 1 to 2 (out of 2 items)

Intranet local | 100%

This way GeoServer generates a WFS request that automatically retrieves the dataset in GML3 format. In our case, it has generated http://localhost:8080/geoserver/ows?service=WFS&version=1.0.0&request=GetFeature &typeName=arctic:cryosphere_atlas_north13126453601905475_data&maxFeatures=50 &outputFormat=text/xml; subtype=gml/3.1.1
and we have got the whole polygon dataset as an XML file.

Also, in the Gaia v.3 application we can choose tools|add layer and then select the button with a "plus" symbol. Now we must indicate the name of the service (you can provide any name) and the URL of the GeoServer (by default: http://localhost:8080/geoserver/ows).


Add an OGC Service to the List

Service Info

Name: artic

URL: http://localhost:8080/geoserver/ows

Version: 1.0.0

Service Type: WMS WCS [✓ WFS] WMTS

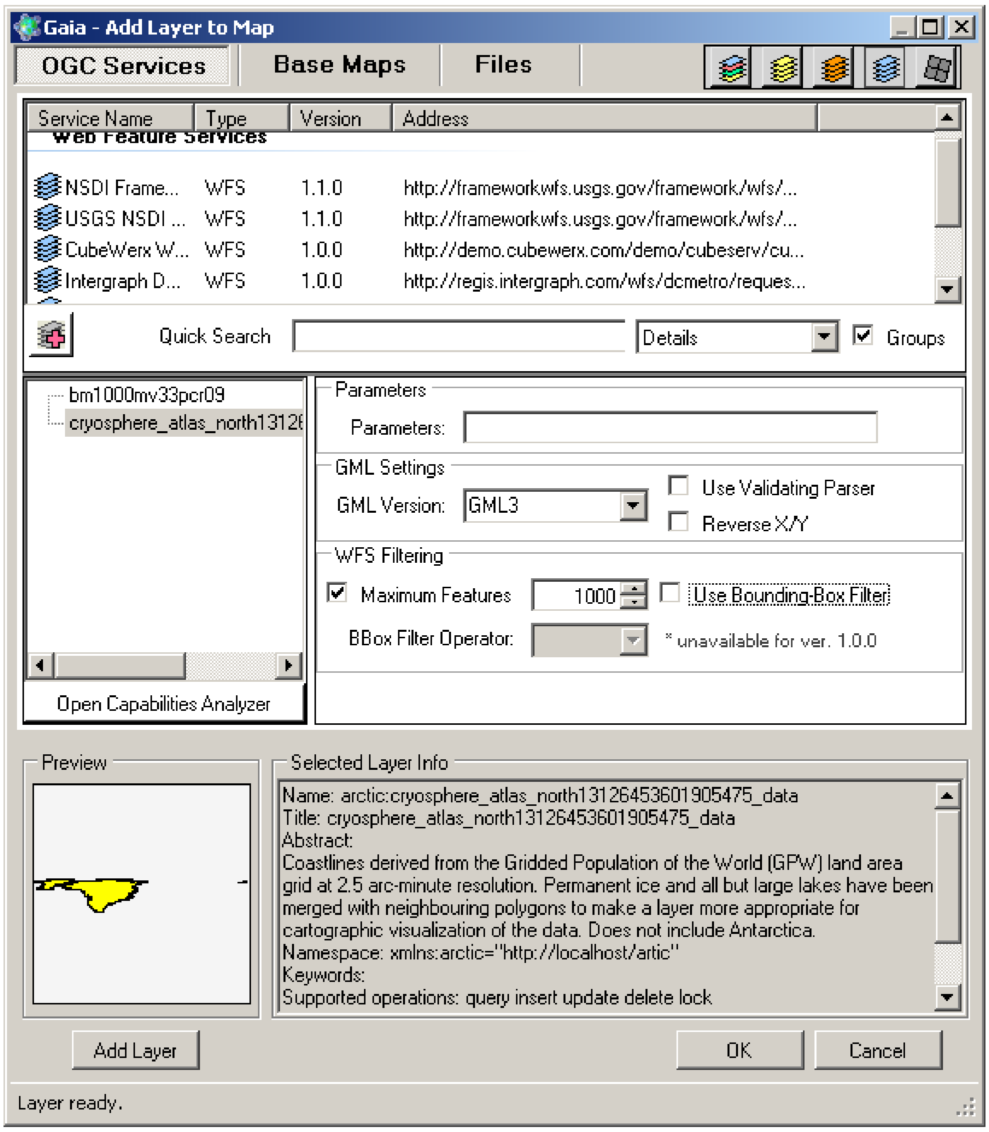
Description

Authentication

User Name:

Password:

After pressing ok, you will be able to select the cryosphere layer and the version of GML to use.



Service Name	Type	Version	Address
Web Feature Services			
NSDI Frame...	WFS	1.1.0	http://frameworkwfs.usgs.gov/framework/wfs/...
USGS NSDI ...	WFS	1.1.0	http://frameworkwfs.usgs.gov/framework/wfs/...
CubeWerx W...	WFS	1.0.0	http://demo.cubewerx.com/demo/cubeserv/cu...
Intergraph D...	WFS	1.0.0	http://regis.intergraph.com/wfs/dcmetro/reques...

Quick Search: Details; Groups: ✓

bm1000mv33pcr09
cryosphere_atlas_north1312

Open Capabilities Analyzer

Parameters
Parameters:

GML Settings
GML Version: GML3
Use Validating Parser: ☐
Reverse X/Y: ☐

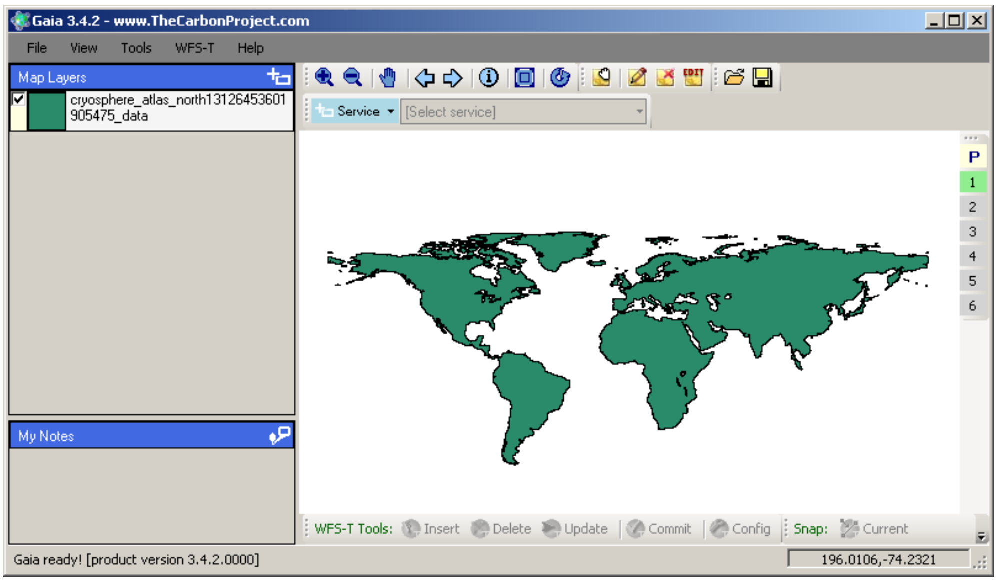
WFS Filtering
Maximum Features: ✓ 1000
Use Bounding-Box Filter: ☐
BBox Filter Operator:
* unavailable for ver. 1.0.0

Preview

Selected Layer Info
Name: arctic:cryosphere_atlas_north13126453601905475_data
Title: cryosphere_atlas_north13126453601905475_data
Abstract:
Coastlines derived from the Gridded Population of the World (GPW) land area grid at 2.5 arc-minute resolution. Permanent ice and all but large lakes have been merged with neighbouring polygons to make a layer more appropriate for cartographic visualization of the data. Does not include Antarctica.
Namespace: xmlns:arctic="http://localhost/artic"
Keywords:
Supported operations: query insert update delete lock

Layer ready.

and this is the result:

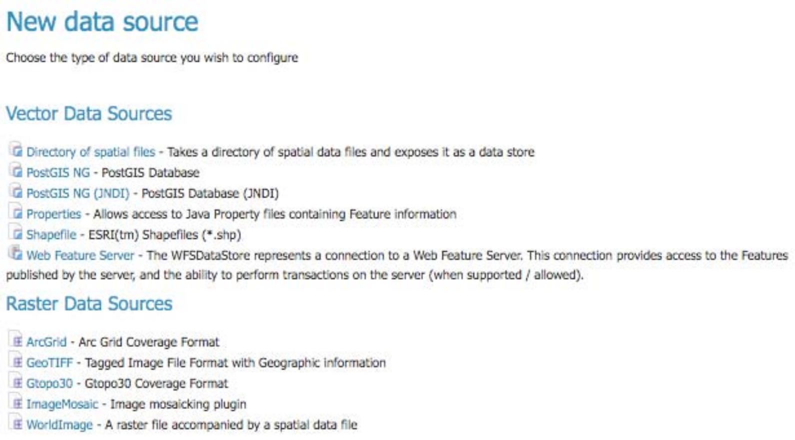


4.3.2.3 Case 3: Offering a PostGIS table with geoserver WFS

Step 1: Setup a PostGIS data

Create a new data store¶
The first step is to create a *data store* for the PostGIS database "nyc". The data store tells GeoServer how to connect to the database.

1. In a web browser navigate to http://localhost:8080/geoserver.
2. Navigate to Data Stores.



New data source

Choose the type of data source you wish to configure

Vector Data Sources

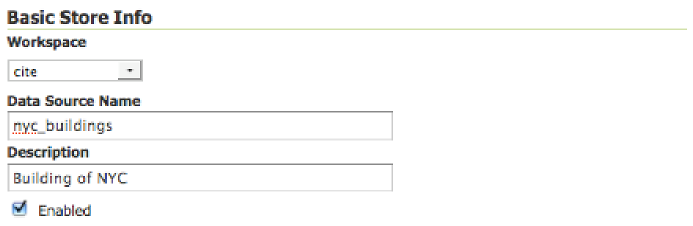
- Directory of spatial files - Takes a directory of spatial data files and exposes it as a data store
- PostGIS NG - PostGIS Database
- PostGIS NG (JNDI) - PostGIS Database (JNDI)
- Properties - Allows access to Java Property files containing Feature information
- Shapefile - ESRI(tm) Shapefiles (*.shp)
- Web Feature Server - The WFSDataStore represents a connection to a Web Feature Server. This connection provides access to the Features published by the server, and the ability to perform transactions on the server (when supported / allowed).

Raster Data Sources

- ArcGrid - Arc Grid Coverage Format
- GeoTIFF - Tagged Image File Format with Geographic Information
- Gtopo30 - Gtopo30 Coverage Format
- ImageMosaic - Image mosaicking plugin
- WorldImage - A raster file accompanied by a spatial data file

Adding a New Data Source

3. Create a new data store by clicking the PostGIS link.
4. Keeping the default *Workspace* enter *Basic Store Info* of Name and Description.


Basic Store Info

Workspace: cite

Data Source Name: nyc_buildings

Description: Building of NYC

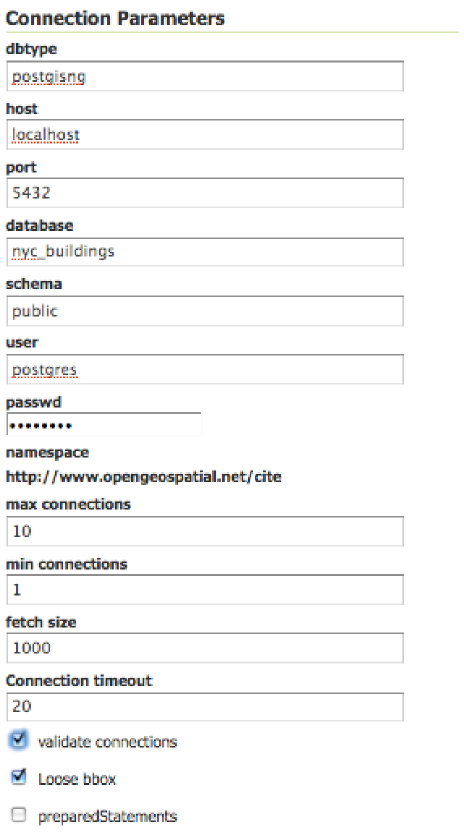
☑ Enabled

Basic Store Info

5. Specify the PostGIS database *Connection Parameters*

Option	**Description**
Workspace	Name of the workspace to contain the database. This will also be the prefix of any layer names created from tables in the database.
Data Source Name	Name of the database. This can be different from the name as known to PostgreSQL/PostGIS.
Description	Description of the database/store.
Enabled	Enables the store. If disabled, no data in the database will be served.
dbtype	Type of database. Leave this value as the default.
host	Host name where the database exists.
port	Port number to connect to the above host.
database	Name of the database as known on the host.
schema	Schema in the above database.
user	User name to connect to the database.
passwd	Password associated with the above user.
namespace	Namespace to be associated with the database. This field is altered by changing the workspace name.
max connections	Maximum amount of open connections to the database.
min connections	Minimum number of pooled connections.
fetch size	Number of records read with each interaction with the database.
Connection timeout	Time (in seconds) the connection pool will wait before timing out.
validate connections	Checks the connection is alive before using it.
Loose bbox	Performs only the primary filter on the bounding box. See the section on *Using loose bounding box* for details.
preparedStatements	Enables prepared statements.

6. Note The **username** and **password** parameters specific to the user who created the postgis database. Depending on how PostgreSQL is configured the password parameter may be unnecessary.


Connection Parameters

dbtype	postgisng
host	localhost
port	5432
database	nyc_buildings
schema	public
user	postgres
passwd	••••••••
namespace	http://www.opengeospatial.net/cite
max connections	10
min connections	1
fetch size	1000
Connection timeout	20

[✓] validate connections

[✓] Loose bbox

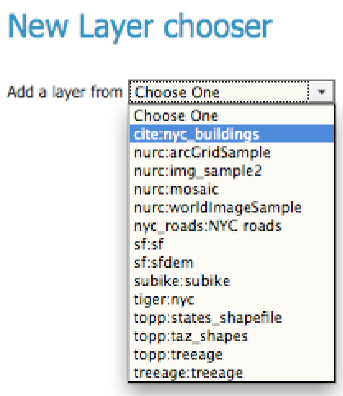
[ ] preparedStatements

Connection Parameters

7. Click the Save button

Layer Configuration¶

1. Navigate to *Data*‣*Layers*.
2. Select *Add a new resource* button.
3. From the *New Layer chooser* drop down menu, select cite:nyc_buidings.

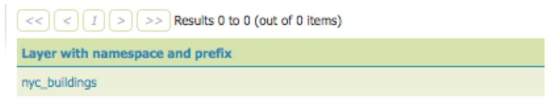

New Layer chooser

Add a layer from: Choose One

- Choose One
- cite:nyc_buildings
- nurc:arcGridSample
- nurc:img_sample2
- nurc:mosaic
- nurc:worldImageSample
- nyc_roads:NYC roads
- sf:sf
- sf:sfdem
- subike:subike
- tiger:nyc
- topp:states_shapefile
- topp:taz_shapes
- topp:treeage
- treeage:treeage

New Layer drop down selection

On the resulting layer row, select the Layer name nyc_buildings.

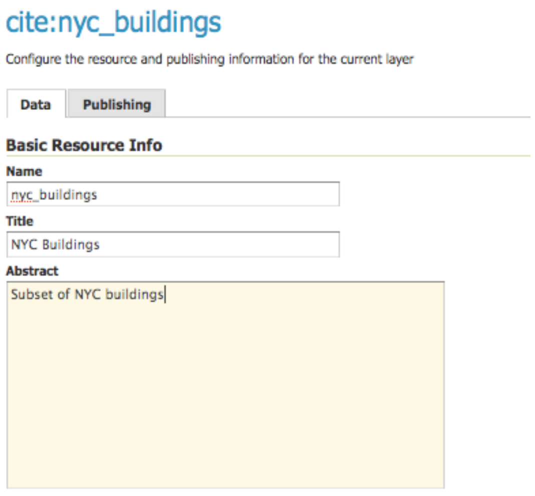


Layer with namespace and prefix

nyc_buildings

New Layer row

4. The following configurations define the data and publishing parameters for a layer. Enter the Basic Resource Info for nyc_buildings.



cite:nyc_buildings

Configure the resource and publishing information for the current layer

Basic Resource Info

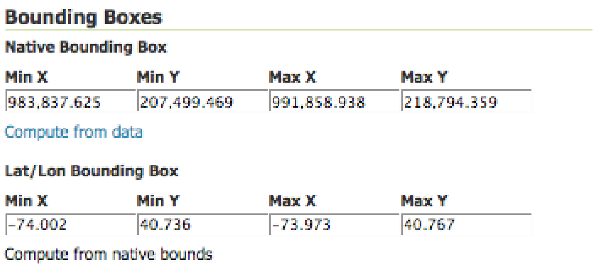
Name: nyc_buildings

Title: NYC Buildings

Abstract: Subset of NYC buildings

Basic Resource Info

5. Generate the database *bounds* by clicking the *Compute from data* and then *Compute from Native bounds.*



Bounding Boxes

Native Bounding Box

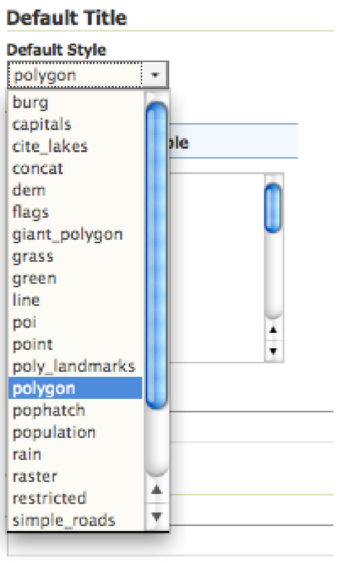
Min X	Min Y	Max X	Max Y
983,837.625	207,499.469	991,858.938	218,794.359

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
-74.002	40.736	-73.973	40.767

Generate Bounding Box

6. Set the layer's *style* by first moving over to the *Publishing* tab.
7. Then select *polygon* from the *Default Style* drop down list.

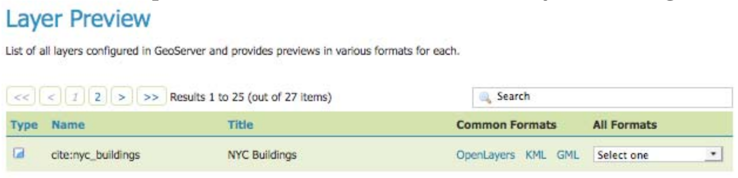


Select Default Style

8. Finalize your data and publishing configuration by scrolling to the bottom and clicking *Save*.

Preview the Layer¶

9. In order to verify that the nyc_building is published we will preview the layer. Navigate to the Map Preview and search for the cite:nyc_buildings



Layer Preview

List of all layers configured in GeoServer and provides previews in various formats for each.

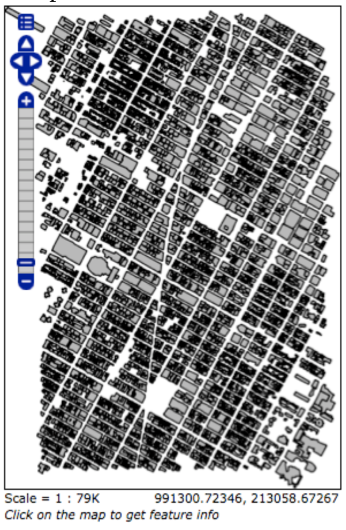
<< < 1 2 > >> Results 1 to 25 (out of 27 items)

Type	Name	Title	Common Formats	All Formats
	cite:nyc_buildings	NYC Buildings	OpenLayers KML GML	Select one

Layer Preview

10. Click on the OpenLayers link under the Common Formats column.

Success! An OpenLayers map should load with the default polygon style.



OpenLayers map of nyc_buildings

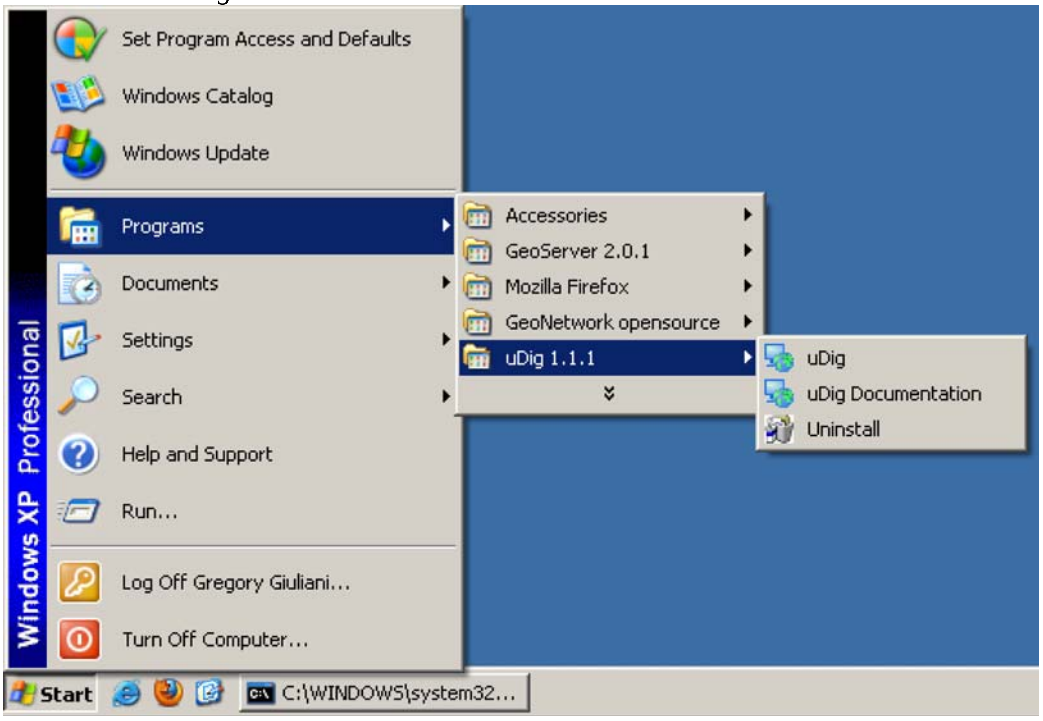
Step 2: Test your WFS Service in clients

You can test WFS service in various clients. In our case we will show how to access a WFS Service in uDig (http://udig.refractions.net)

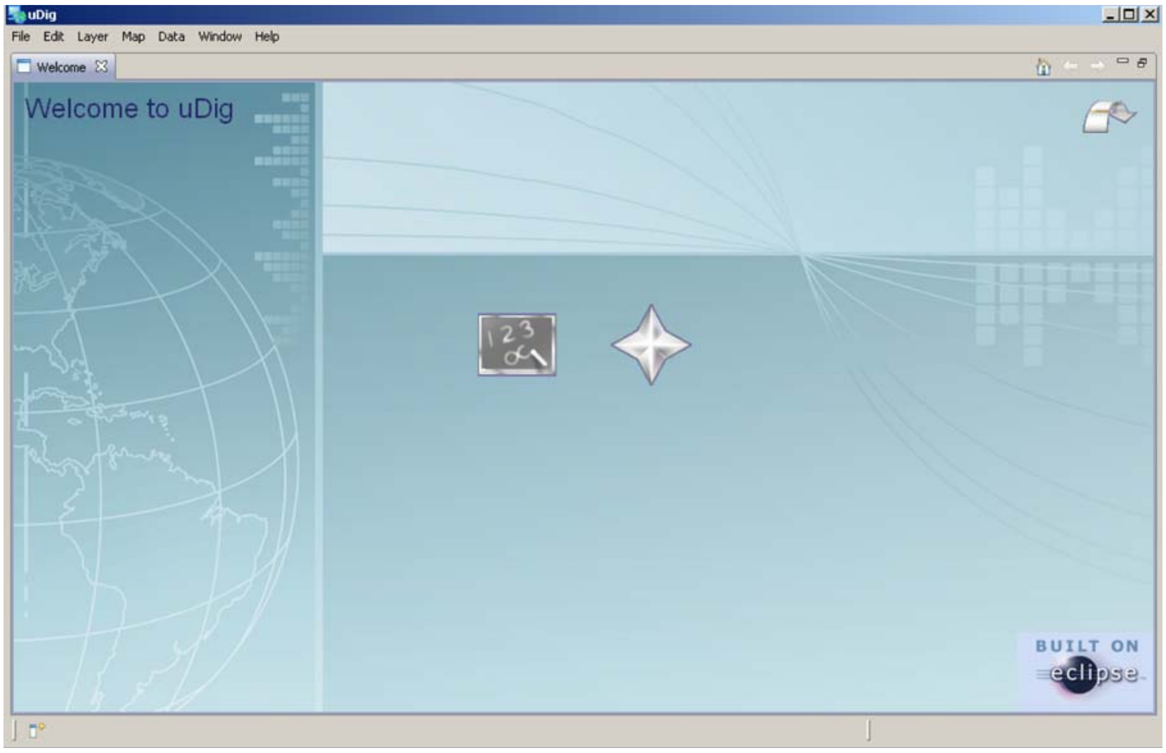
We asume that uDig is installed on the computer.

Getting started

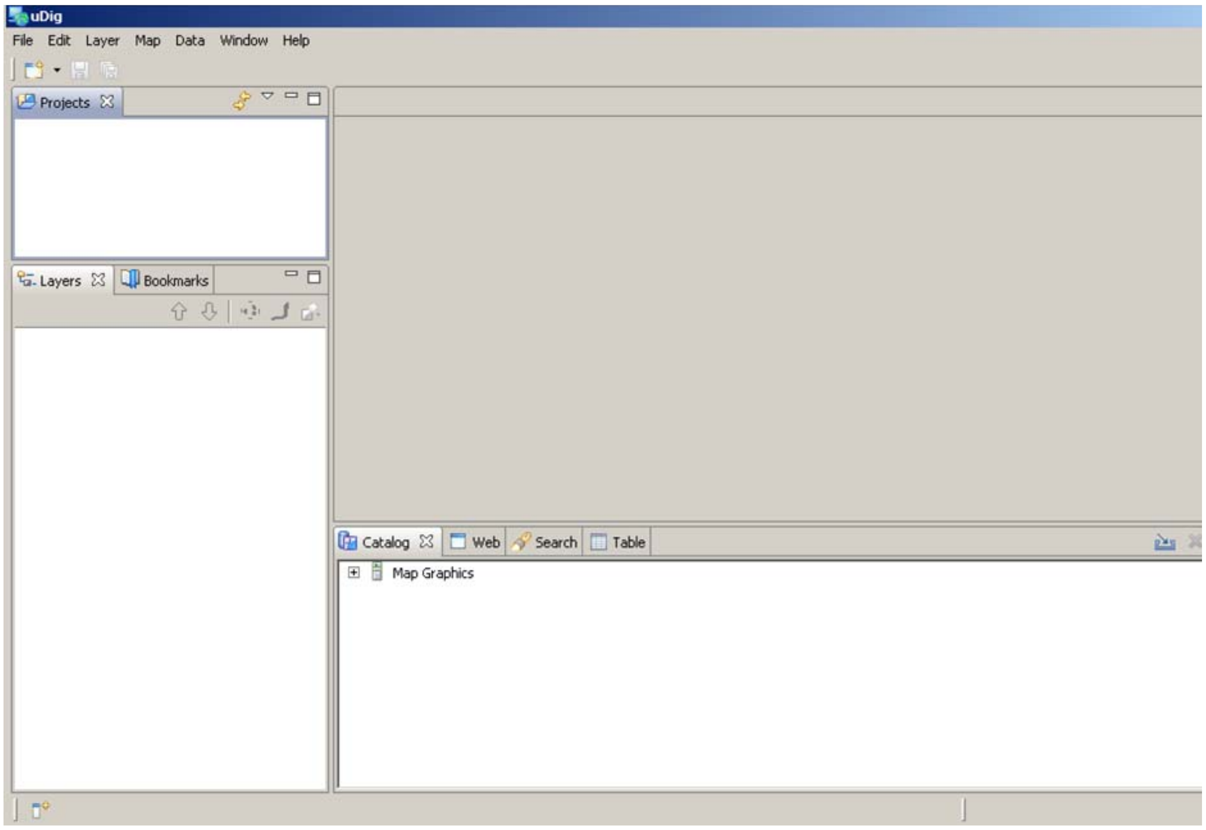
1. After completing the installation, run the uDig application from the Windows *Start > Programs menu*.



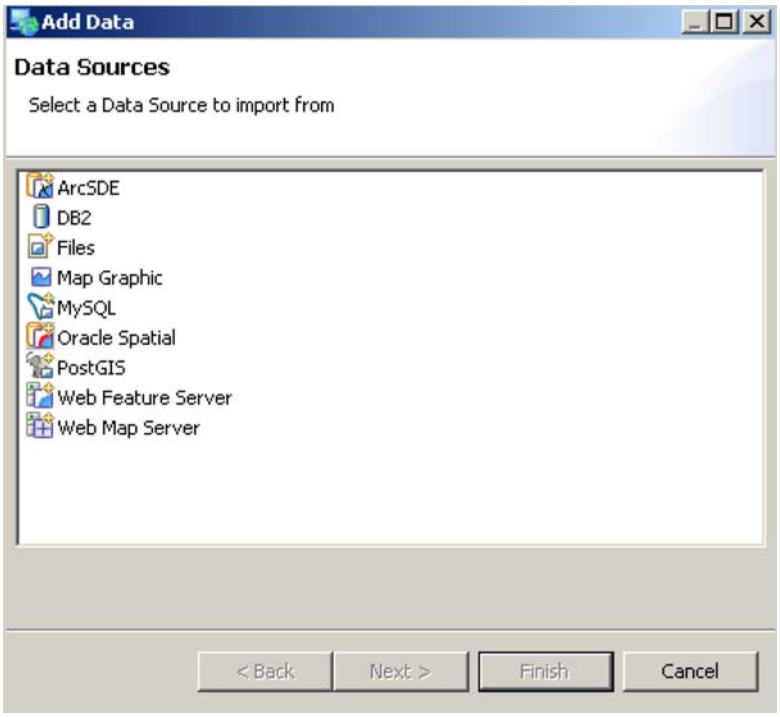
2. Initially you are presented with the welcome screen.
3. The welcome screen contains a link to the Getting Started tutorial from the online documentation and a link to the Official Website.



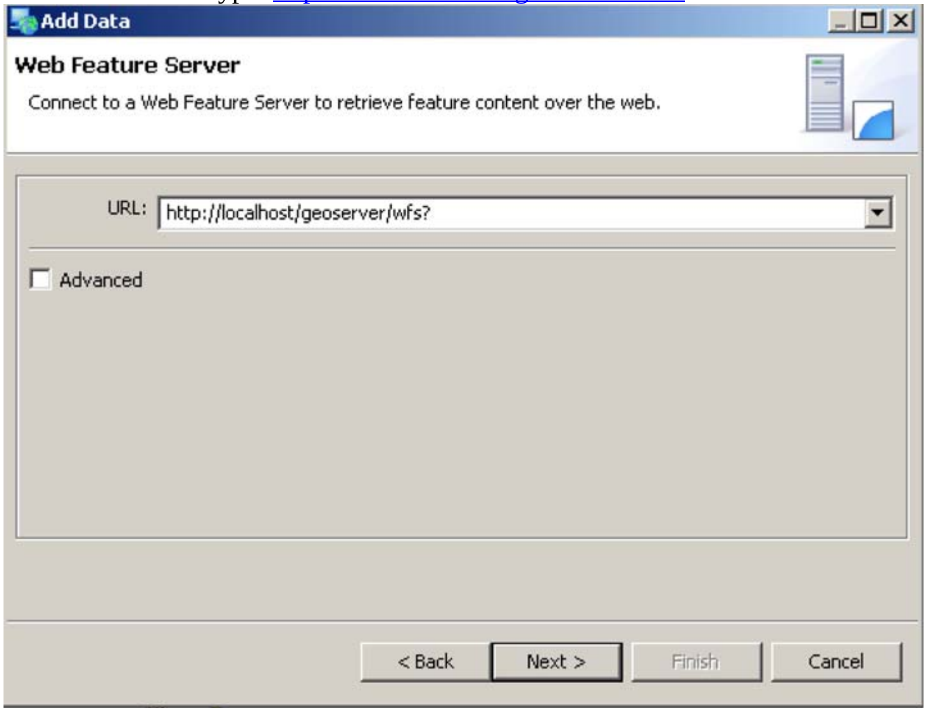
4. To continue click on the Workbench arrow in the top right corner of the uDig welcome screen.
5. You are now ready to start to work with uDig.



Start uDig, under the File menu select *New* > *New Map* to create a new map.
Make sure your local GeoServer is started and ready to go.
Select *Layer* > *Add*
This will bring up a Wizard allowing you to choose which Data source you want to use..
Select Web Feature Server.



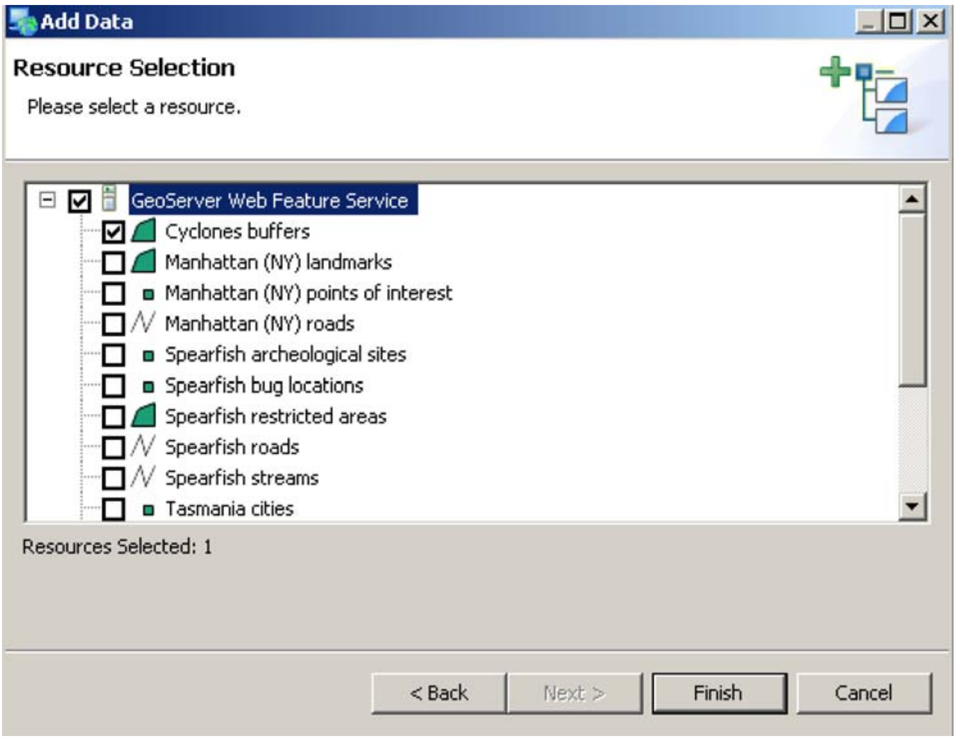
This open a window where you have to insert the URL of a web feature server you want to access. In our case type: http://localhost:8080/geoserver/wfs? and click Next.


Add Data
Web Feature Server
Connect to a Web Feature Server to retrieve feature content over the web.
< Back
Next >
Finish
Cancel

URL: http://localhost/geoserver/wfs?

☐ Advanced

You have now access to all WFS layers available on our GeoServer. Select Manhattan (NY) landmarks and the press the Finish button. The layer will be drawn on your map.


Add Data

Resource Selection

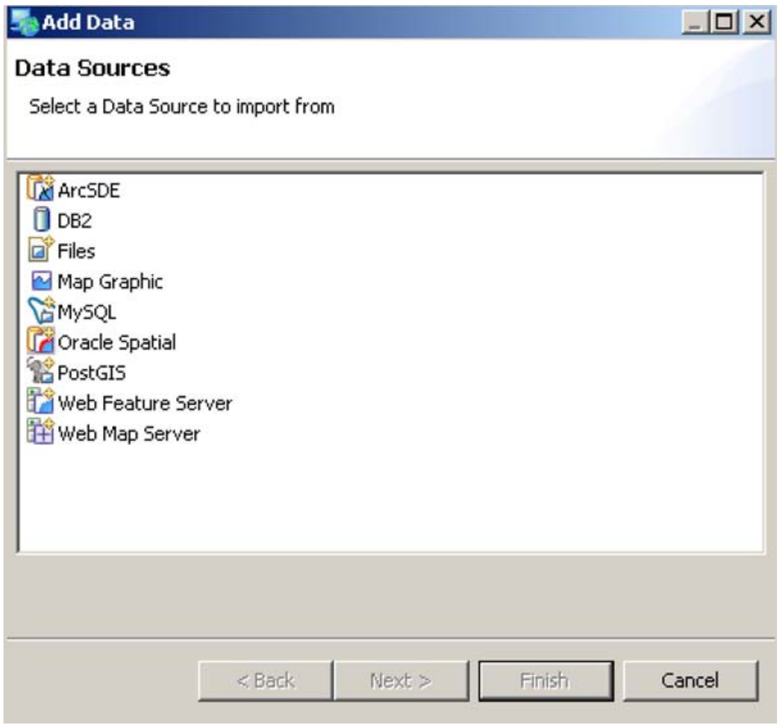
Please select a resource.

- [✓] GeoServer Web Feature Service
  - [✓] Cyclones buffers
  - [ ] Manhattan (NY) landmarks
  - [ ] Manhattan (NY) points of interest
  - [ ] Manhattan (NY) roads
  - [ ] Spearfish archeological sites
  - [ ] Spearfish bug locations
  - [ ] Spearfish restricted areas
  - [ ] Spearfish roads
  - [ ] Spearfish streams
  - [ ] Tasmania cities

Resources Selected: 1

Accessing the service

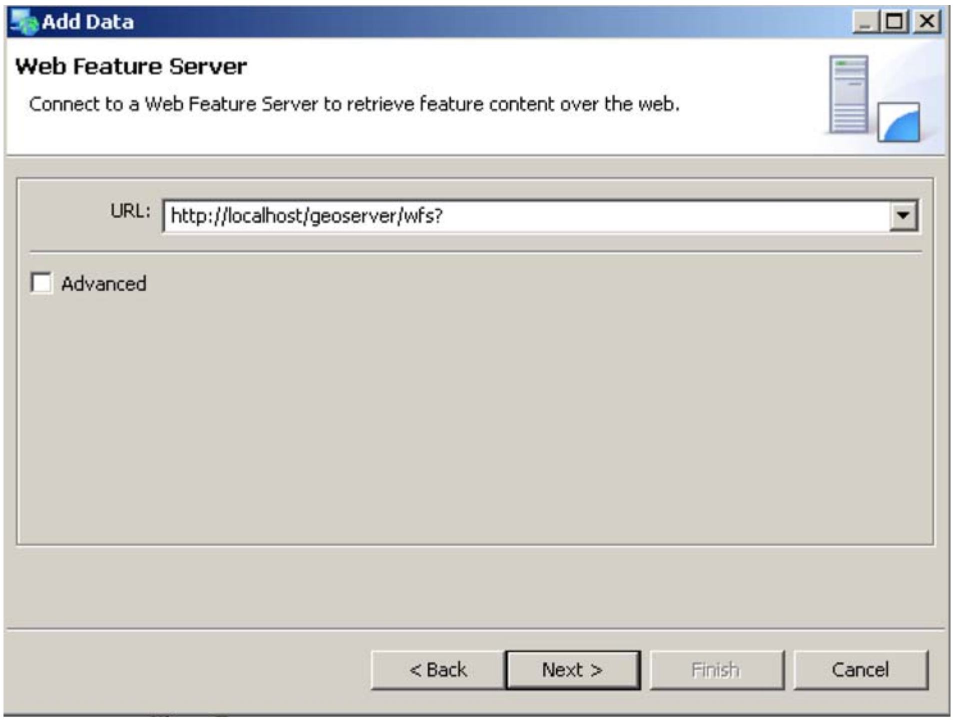
7. Start uDig, under the File menu select New > New Map to create a new map.
8. Make sure your local GeoServer is started and ready to go.
9. Select Layer > Add
10. This will bring up a Wizard allowing you to choose which Data source you want to use. Select **Web Feature Server**.


Add Data

Data Sources

Select a Data Source to import from

11. This open a window where you have to insert the URL of a web feature server you want to access. In our case type: http://localhost:8080/geoserver/wfs? and click **Next**.



Add Data

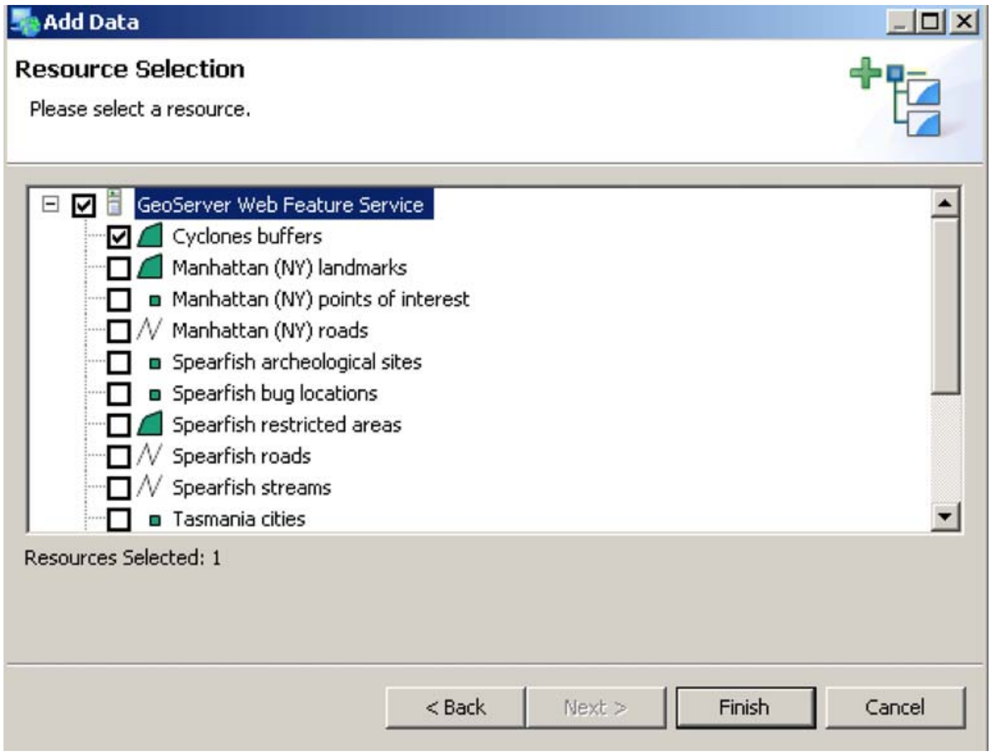
Web Feature Server

Connect to a Web Feature Server to retrieve feature content over the web.

URL: http://localhost/geoserver/wfs?

☐ Advanced

12. You have now access to all WFS layers available on our GeoServer. Select Manhattan (NY) landmarks and the press the **Finish** button. The layer will be drawn on your map.


Add Data

Resource Selection

Please select a resource.

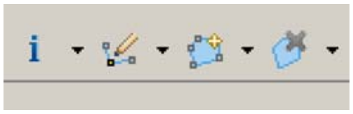
- ☑ GeoServer Web Feature Service
  - ☑ Cyclones buffers
  - ☐ Manhattan (NY) landmarks
  - ☐ Manhattan (NY) points of interest
  - ☐ Manhattan (NY) roads
  - ☐ Spearfish archeological sites
  - ☐ Spearfish bug locations
  - ☐ Spearfish restricted areas
  - ☐ Spearfish roads
  - ☐ Spearfish streams
  - ☐ Tasmania cities

Resources Selected: 1

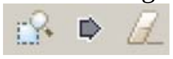
4.3.2.4 Step 6: Updating data

In Step 3 we have accessed a WFS layer. Now we can show the editing (transactional) capabilities of a WFS service.

1. Select *Navigation* > *Show All* from the menu bar.
11. Zoom to a specific buffer
12. Select Manhattan (NY) landmarks in the Layers view.
13. Choose to the Edit Geometry tool from the tool bar



14. Use the Edit Geometry tool to select a buffer. The buffer will change color and develop “vertex handles”.
15. Move the vertex handles as you want.
16. Once finished, press the **Commit Changes** button in the tool bar to send your changes off to the Web Feature Server.



Your changes have been registered.

4.3.2.5 Step 7: Optimizing the service performance and maintenance

To optimize service performance various considerations must be taken into account.
An interesting paper discuss these issues: http://opengeo.org/publications/geoserver-production/

Java consideration
http://docs.geoserver.org/stable/en/user/production/java.html

Container consideration
http://docs.geoserver.org/stable/en/user/production/container.html
Configuration consideration
http://docs.geoserver.org/stable/en/user/production/config.html
Data considerations
http://docs.geoserver.org/stable/en/user/production/data.html
Other
http://docs.geoserver.org/stable/en/user/production/misc.html

4.3.2.6 Step 8: Service errors and troubleshooting

Please refer to: http://docs.geoserver.org/stable/en/user/production/troubleshooting.html

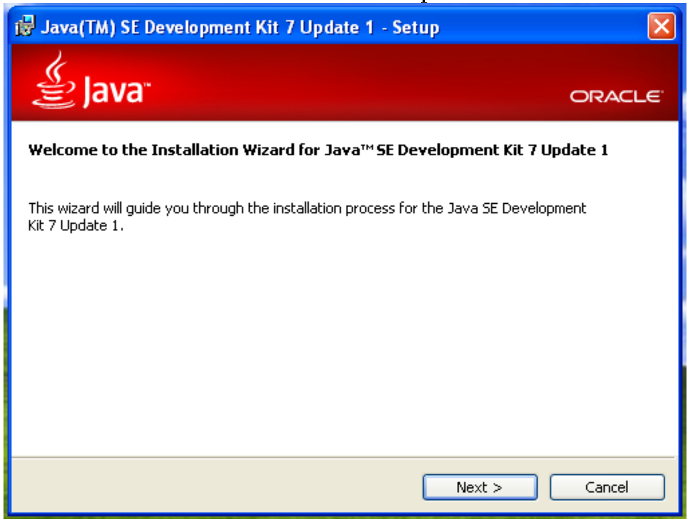
4.3.2.7 Step 9: Register the service with GEOSS

This step is related to the CSR tutorial -> make a link to it.

4.3.3 Offering a shapefile with deegree

Deegree version 3.0 offers the possibility to have a WMS and WFS service simultaneously with a single application. It also provides WPS and CSW support. The easiest way to start with Deegree is setting up the demo version that includes Utah datasets. This version is the one we are going to use.

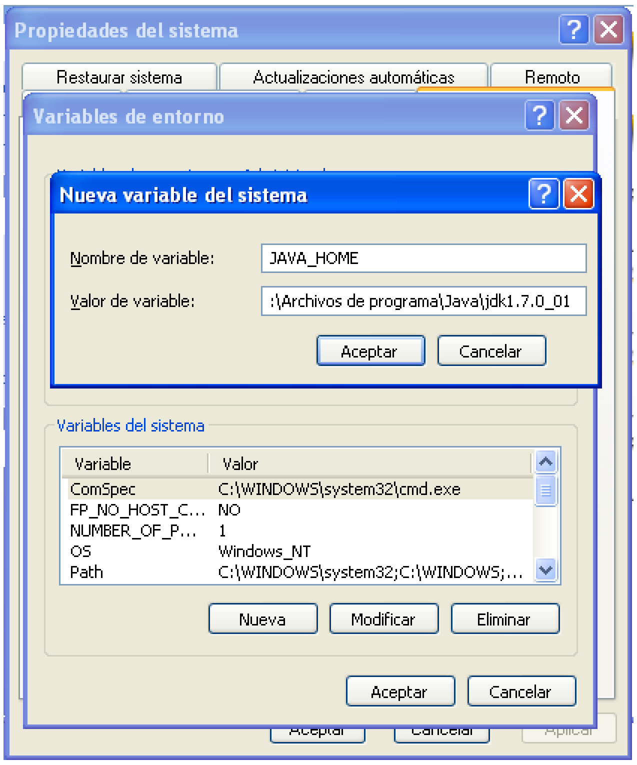
Before setting up degree, you must setup the JDK java virtual machine. Please do not rely on the usual JRE because it will not work. We have downloaded the version jdk-7u1-windows-i586.exe from the Oracle website http://www.oracle.com/index.html



Welcome to the Installation Wizard for Java™ SE Development Kit 7 Update 1

This wizard will guide you through the installation process for the Java SE Development Kit 7 Update 1.

Once the Java setup is completed, it is important to define the JAVA_HOME path. This is done by including a new environment variable in the system by creating it in the advanced tab of the system properties.


Propiedades del sistema

Variables de entorno

Nueva variable del sistema

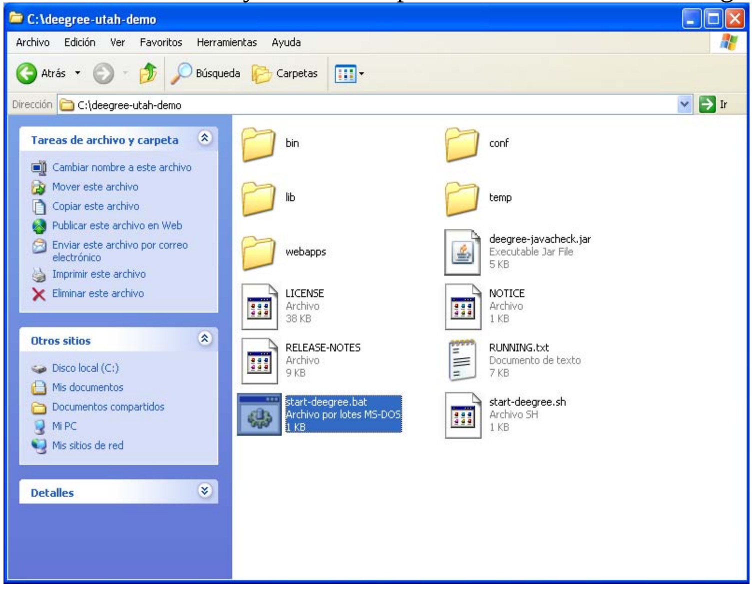
Nombre de variable: JAVA_HOME

Valor de variable: :\Archivos de programa\Java\jdk1.7.0_01

Variables del sistema

Variable	Valor
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	1
OS	Windows_NT
Path	C:\WINDOWS\system32;C:\WINDOWS;...

You can know download Deegree 3.0. We have done it directly from http://download.deegree.org/deegree3/deegree-utah-demo-3.0.4.zip and decompressed the zip file to the c: root directory. The decompression creates the following folder.


C:\deegree-utah-demo

Dirección C:\deegree-utah-demo

bin, conf, lib, temp, webapps

deegree-javacheck.jar Executable Jar File 5 KB

LICENSE Archivo 38 KB

NOTICE Archivo 1 KB

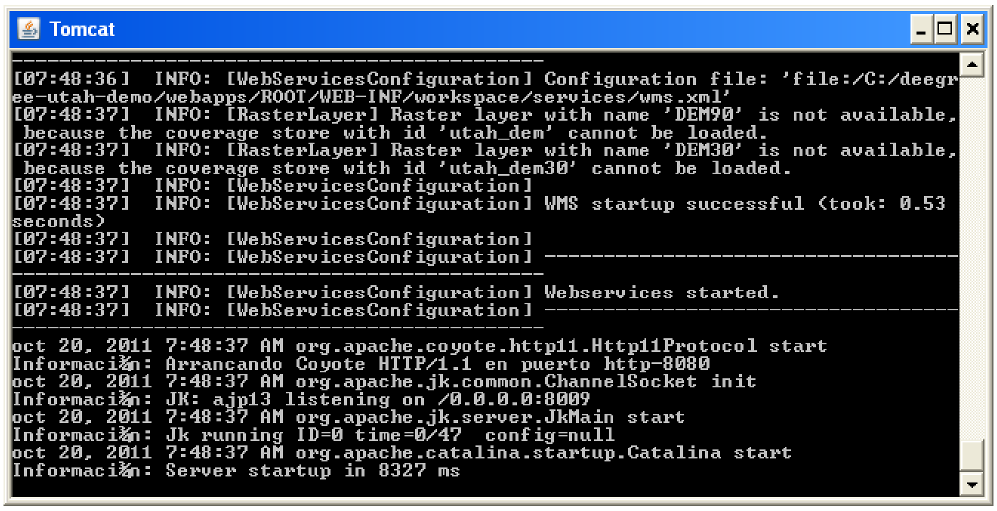
RELEASE-NOTES Archivo 9 KB

RUNNING.txt Documento de texto 7 KB

start-deegree.bat Archivo por lotes MS-DOS 1 KB

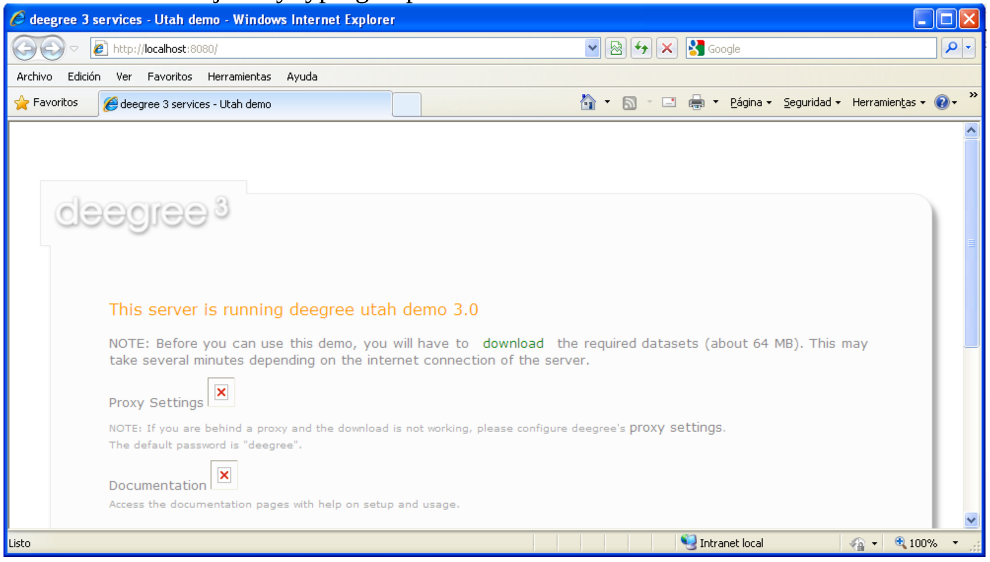
start-deegree.sh Archivo SH 1 KB

Now you can open a Windows console (CMD), change directory to "c:\deegree_utah_demo" and run "start-deegree.bat". The result is another console window with a tomcat report about the web service starting process. Leave this window open, unless you want to close the service.



```
-------------------------------------------------------------
[07:48:36]  INFO: [WebServicesConfiguration] Configuration file: 'file:/C:/deegr
ee-utah-demo/webapps/ROOT/WEB-INF/workspace/services/wms.xml'
[07:48:37]  INFO: [RasterLayer] Raster layer with name 'DEM90' is not available,
 because the coverage store with id 'utah_dem' cannot be loaded.
[07:48:37]  INFO: [RasterLayer] Raster layer with name 'DEM30' is not available,
 because the coverage store with id 'utah_dem30' cannot be loaded.
[07:48:37]  INFO: [WebServicesConfiguration]
[07:48:37]  INFO: [WebServicesConfiguration] WMS startup successful (took: 0.53
seconds)
[07:48:37]  INFO: [WebServicesConfiguration]
[07:48:37]  INFO: [WebServicesConfiguration] -----------------------------------
-------------------------------------------------------------
[07:48:37]  INFO: [WebServicesConfiguration] Webservices started.
[07:48:37]  INFO: [WebServicesConfiguration] -----------------------------------
-------------------------------------------------------------
oct 20, 2011 7:48:37 AM org.apache.coyote.http11.Http11Protocol start
Informaci¾n: Arrancando Coyote HTTP/1.1 en puerto http-8080
oct 20, 2011 7:48:37 AM org.apache.jk.common.ChannelSocket init
Informaci¾n: JK: ajp13 listening on /0.0.0.0:8009
oct 20, 2011 7:48:37 AM org.apache.jk.server.JkMain start
Informaci¾n: Jk running ID=0 time=0/47  config=null
oct 20, 2011 7:48:37 AM org.apache.catalina.startup.Catalina start
Informaci¾n: Server startup in 8327 ms
```

Now Deegree is running in your machine under the port 8080, and you can get access to the main interface just by typing http://localhost:8080 in a web browser:



This server is running deegree utah demo 3.0

NOTE: Before you can use this demo, you will have to download the required datasets (about 64 MB). This may take several minutes depending on the internet connection of the server.

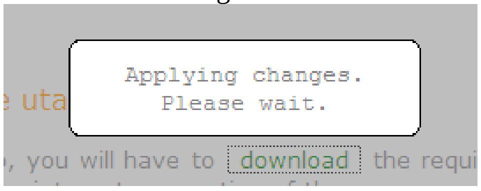
Proxy Settings

NOTE: If you are behind a proxy and the download is not working, please configure deegree's proxy settings.
The default password is "deegree".

Documentation

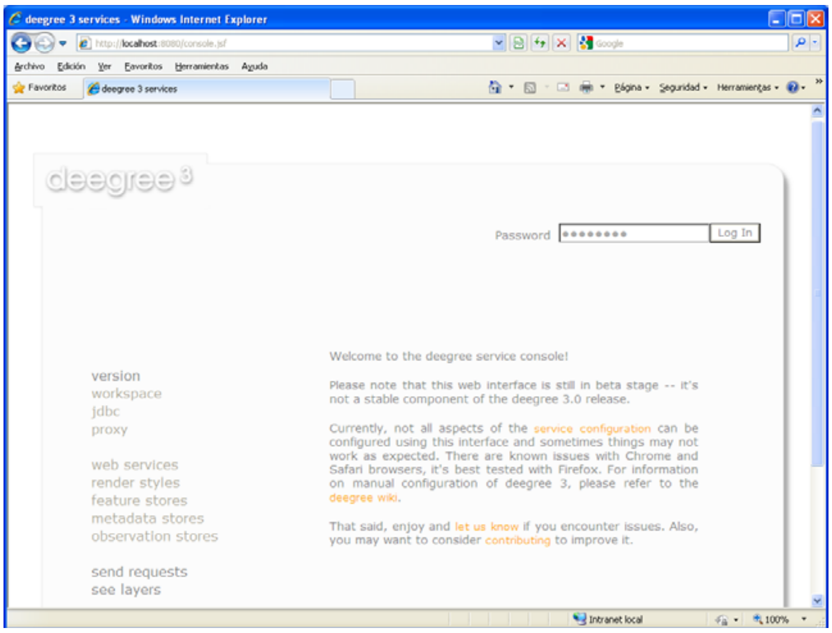
Access the documentation pages with help on setup and usage.

Now, you have to click on "download" to get the data that is needed to run this demo:



Applying changes.
Please wait.

After the download is completed, new options will appear in the same screen. Among the new ones, you will see "administrative console" that allows you to configure your server and to add new data:



Welcome to the deegree service console!

Please note that this web interface is still in beta stage -- it's not a stable component of the deegree 3.0 release.

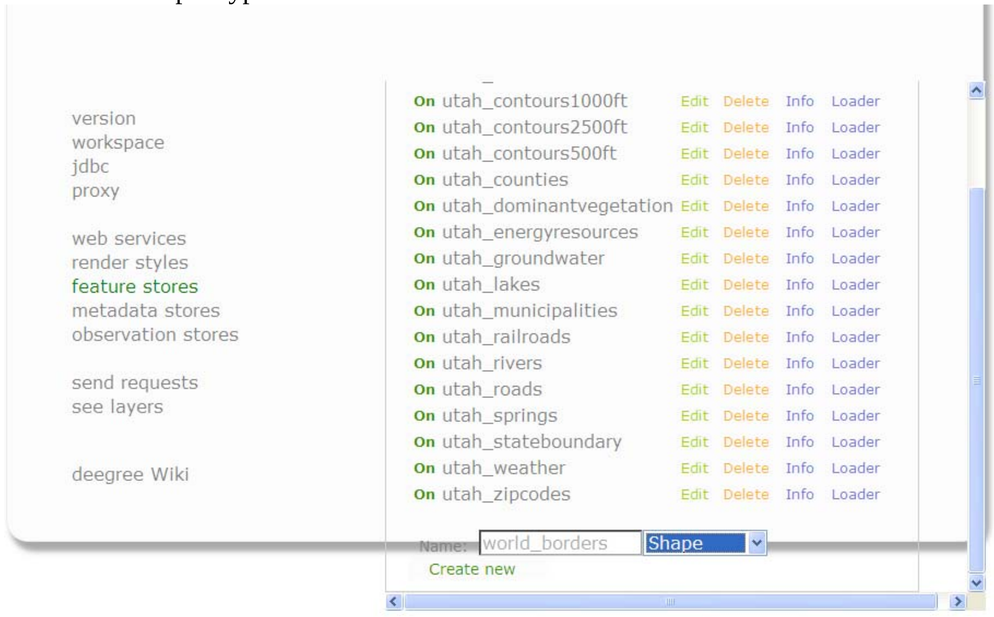
Currently, not all aspects of the service configuration can be configured using this interface and sometimes things may not work as expected. There are known issues with Chrome and Safari browsers, it's best tested with Firefox. For information on manual configuration of deegree 3, please refer to the deegree wiki.

That said, enjoy and let us know if you encounter issues. Also, you may want to consider contributing to improve it.

To get in, you need to know the default password: "deegree".

Step 1: Adding new data

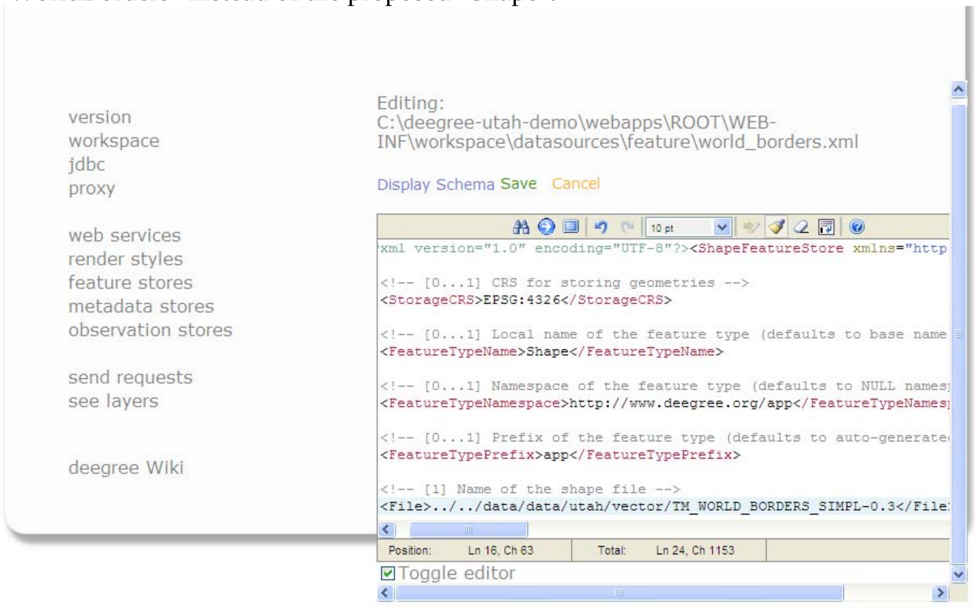
At this point, we are going to create a new feature store with the name "world_borders", which is a "Shape" type.



```
On utah_contours1000ft        Edit Delete Info Loader
On utah_contours2500ft        Edit Delete Info Loader
On utah_contours500ft         Edit Delete Info Loader
On utah_counties              Edit Delete Info Loader
On utah_dominantvegetation    Edit Delete Info Loader
On utah_energyresources       Edit Delete Info Loader
On utah_groundwater           Edit Delete Info Loader
On utah_lakes                 Edit Delete Info Loader
On utah_municipalities        Edit Delete Info Loader
On utah_railroads             Edit Delete Info Loader
On utah_rivers                Edit Delete Info Loader
On utah_roads                 Edit Delete Info Loader
On utah_springs               Edit Delete Info Loader
On utah_stateboundary         Edit Delete Info Loader
On utah_weather               Edit Delete Info Loader
On utah_zipcodes              Edit Delete Info Loader

Name: world_borders  Shape
Create new
```

The system creates a new configuration file for this layer. We have to manually edit the position of the vector shape file that we are going to publish. We have downloaded a world borders dataset from: http://thematicmapping.org/downloads/TM_WORLD_BORDERS_SIMPL-0.3.zip. To simplify the process we are going to decompress the data directly into the same

directory where the Utah data is: C:\deegree-utah-demo\webapps\ROOT\WEB-INF\workspace\data\data\utah\vector.
In the next screen, we have to change the content of the <file> element to include the relative position of the shp file using '/' as slashes (instead of back slashes) and without the extension ".shp". We also have to change the <FeatureTypeName> to a data type name that will be representative in the wfs capabilities. In our case we introduced "WorldBorders" instead of the proposed "Shape".

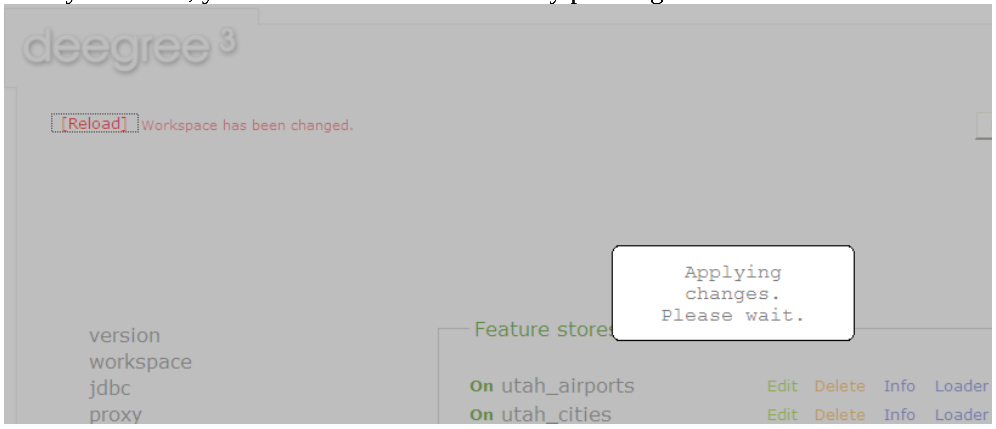


Editing:
C:\deegree-utah-demo\webapps\ROOT\WEB-INF\workspace\datasources\feature\world_borders.xml

```
xml version="1.0" encoding="UTF-8"?><ShapeFeatureStore xmlns="http

<!-- [0...1] CRS for storing geometries -->
<StorageCRS>EPSG:4326</StorageCRS>

<!-- [0...1] Local name of the feature type (defaults to base name
<FeatureTypeName>Shape</FeatureTypeName>

<!-- [0...1] Namespace of the feature type (defaults to NULL names
<FeatureTypeNamespace>http://www.deegree.org/app</FeatureTypeNames

<!-- [0...1] Prefix of the feature type (defaults to auto-generate
<FeatureTypePrefix>app</FeatureTypePrefix>

<!-- [1] Name of the shape file -->
<File>../../data/data/utah/vector/TM_WORLD_BORDERS_SIMPL-0.3</File
```

Position: Ln 16, Ch 63 Total: Ln 24, Ch 1153

Once you save it, you have to reload the server by pressing a small red linked text.

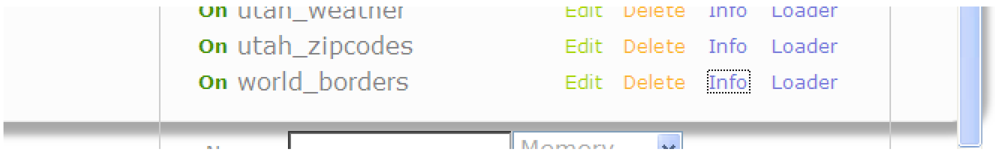


[Reload] Workspace has been changed.

Applying changes. Please wait.

On utah_airports
On utah_cities

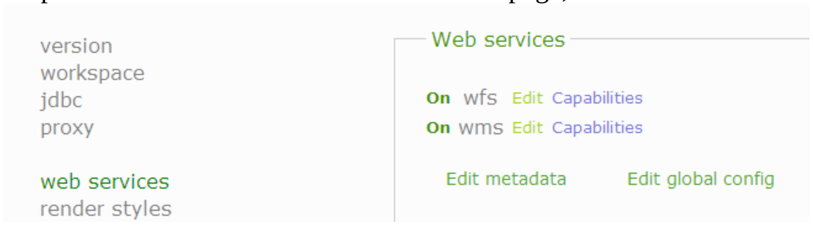
Finally, the layer becomes available at the end of the list below the others.



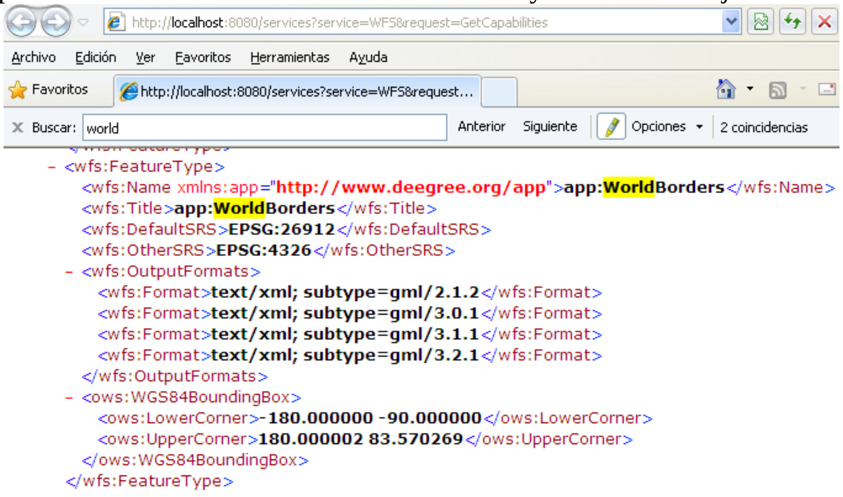
On utah_weather
On utah_zipcodes
On world_borders

Step 2: Testing the layer

The first evidence that the system is working properly is that we can see the layer in the capabilities document. Accessing the capabilities document is as simple as clicking in the text "Capabilities" from in the "web services" subpage, next to "On wfs Edit":



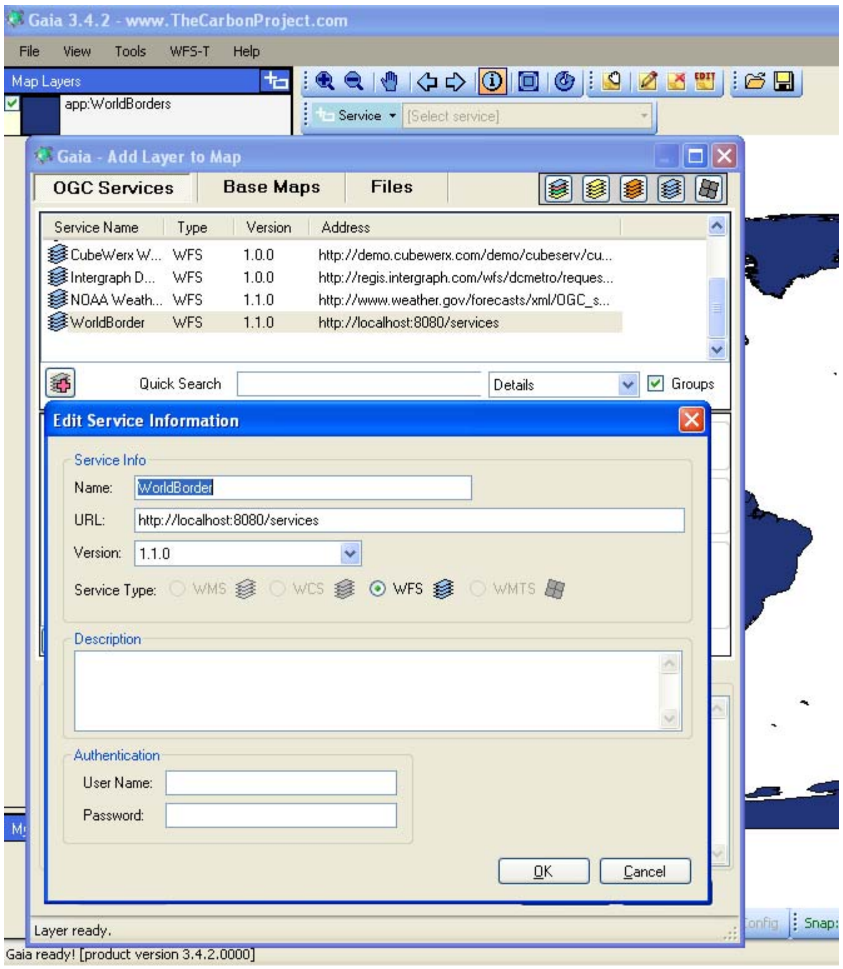
The capabilities document includes the name of the layer that we have just included:


http://localhost:8080/services?service=WFS&request=GetCapabilities
Archivo Edición Ver Favoritos Herramientas Ayuda
Favoritos
Buscar: world
Anterior Siguiente Opciones 2 coincidencias

```
- <wfs:FeatureType>
    <wfs:Name xmlns:app="http://www.deegree.org/app">app:WorldBorders</wfs:Name>
    <wfs:Title>app:WorldBorders</wfs:Title>
    <wfs:DefaultSRS>EPSG:26912</wfs:DefaultSRS>
    <wfs:OtherSRS>EPSG:4326</wfs:OtherSRS>
  - <wfs:OutputFormats>
      <wfs:Format>text/xml; subtype=gml/2.1.2</wfs:Format>
      <wfs:Format>text/xml; subtype=gml/3.0.1</wfs:Format>
      <wfs:Format>text/xml; subtype=gml/3.1.1</wfs:Format>
      <wfs:Format>text/xml; subtype=gml/3.2.1</wfs:Format>
    </wfs:OutputFormats>
  - <ows:WGS84BoundingBox>
      <ows:LowerCorner>-180.000000 -90.000000</ows:LowerCorner>
      <ows:UpperCorner>180.000002 83.570269</ows:UpperCorner>
    </ows:WGS84BoundingBox>
  </wfs:FeatureType>
```

Once again, we use Gaia v3 to illustrate that the resulting dataset works perfectly. Gaia can be downloaded from: http://www.thecarbonproject.com/gaia.php and requires Microsoft NET framework 2.0 to start.

When the program is started, the only thing we have to do is to press the icon with the plus symbol from the dark blue bar, and add a new server by clicking again in the multilayer with a plus symbol icon. In our installation, the name of the service is http://localhost:8080/services but, in a real implementation, we are going to use the ip name of the machine instead of “localhost”.


Gaia 3.4.2 - www.TheCarbonProject.com

Gaia - Add Layer to Map

Service Name	Type	Version	Address
CubeWerx W...	WFS	1.0.0	http://demo.cubewerx.com/demo/cubeserv/cu...
Intergraph D...	WFS	1.0.0	http://regis.intergraph.com/wfs/dcmetro/reques...
NOAA Weath...	WFS	1.1.0	http://www.weather.gov/forecasts/xml/OGC_s...
WorldBorder	WFS	1.1.0	http://localhost:8080/services

Edit Service Information

Service Info

Name: WorldBorder

URL: http://localhost:8080/services

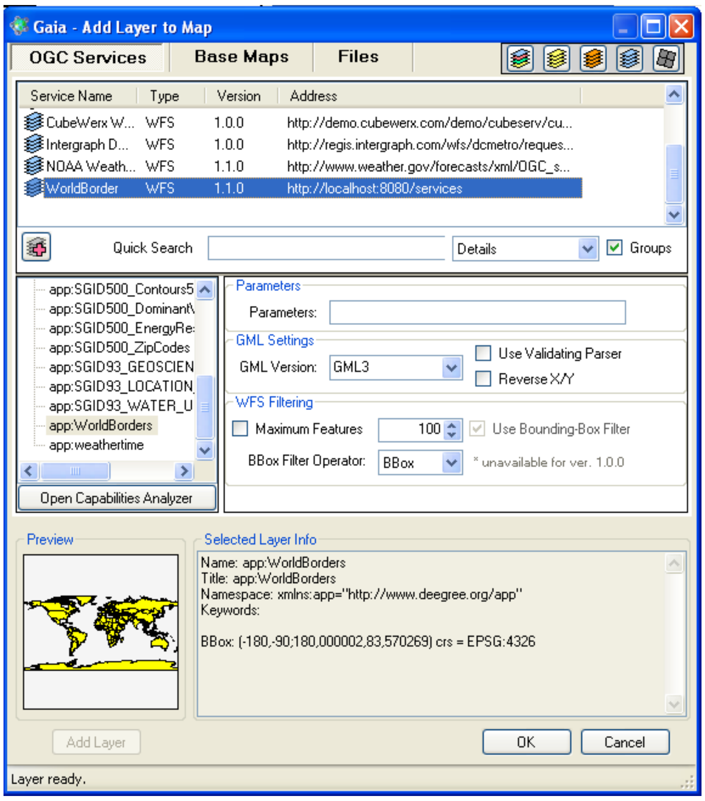
Version: 1.1.0

Service Type: WMS WCS [✓ WFS] WMTS

Layer ready.

Gaia ready! [product version 3.4.2.0000]

After an arbitrary "name" is introduced and the "version" 1.1.0 is selected, we can click "OK" to select the relevant layer that, in this particular case is app:WorldBorders. If you are doing itcorrectly, you will see a preview of the layer:



Service Name	Type	Version	Address
CubeWerx W...	WFS	1.0.0	http://demo.cubewerx.com/demo/cubeserv/cu...
Intergraph D...	WFS	1.0.0	http://regis.intergraph.com/wfs/dcmetro/reques...
NOAA Weath...	WFS	1.1.0	http://www.weather.gov/forecasts/xml/OGC_s...
WorldBorder	WFS	1.1.0	http://localhost:8080/services

app:SGID500_Contours5
app:SGID500_Dominant\
app:SGID500_EnergyRe:
app:SGID500_ZipCodes
app:SGID93_GEOSCIEN
app:SGID93_LOCATION
app:SGID93_WATER_U
app:WorldBorders
app:weathertime

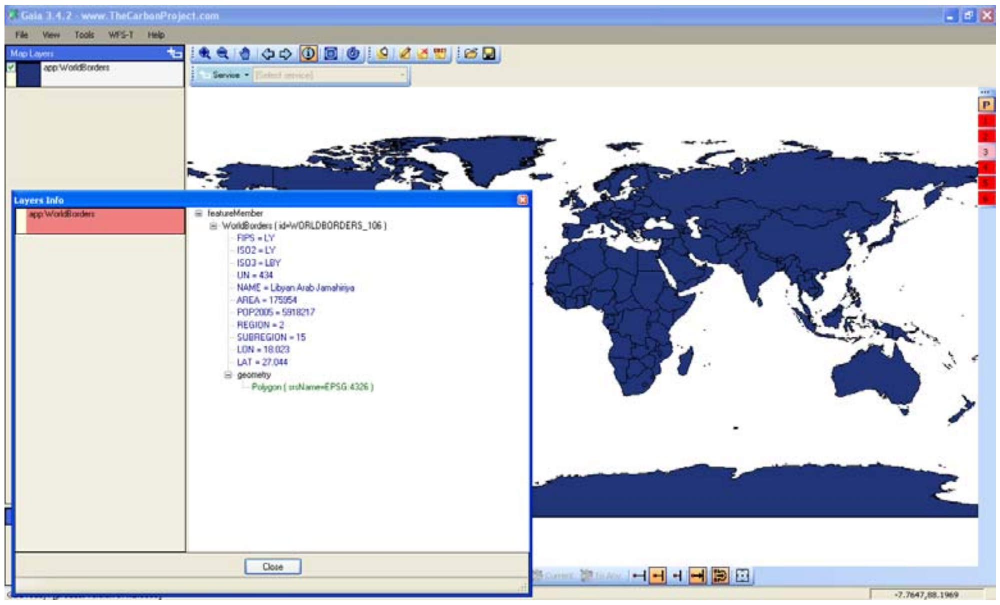
Parameters
Parameters:
GML Settings
GML Version: GML3
Use Validating Parser
Reverse X/Y
WFS Filtering
Maximum Features 100
Use Bounding-Box Filter
BBox Filter Operator: BBox
* unavailable for ver. 1.0.0

Selected Layer Info
Name: app:WorldBorders
Title: app:WorldBorders
Namespace: xmlns:app="http://www.deegree.org/app"
Keywords:

BBox: (-180,-90;180,000002,83,570269) crs = EPSG:4326

Layer ready.

Since this is a WFS service, the elements are individualized and we can select one of them to find the attributes.



```
featureMember
  WorldBorders ( id=WORLDBORDERS_106 )
    FIPS = LY
    ISO2 = LY
    ISO3 = LBY
    UN = 434
    NAME = Libyan Arab Jamahiriya
    AREA = 175954
    POP2005 = 5918217
    REGION = 2
    SUBREGION = 15
    LON = 18.023
    LAT = 27.044
    geometry
      Polygon ( srsName=EPSG:4326 )
```

This illustration shows world vector map and the result of a query by location for Libya.

4.3.4 Setting up a community updated layer with WFS-T cubewebx

The section 2.2.5 Transactional WFS Service Interface. Updating features describes how a transactional WFS can be used. The example uses a demonstrative dedicated portal provided by Cubewerx during the Haiti earthquake that is still operational at the URL http://portal.cubewerx.com/cubewerx/cubeserv/cubeserv.cgi?CONFIG=haiti_vgi&DATASTORE=vgi). Please refer to that section for details

Apendix A Troubleshooting

Old GML files can point to and old schema repository such as: http://schemas.opengeospatial.net/gml/2.1.2/feature.xsd. The new OGC schema repository is http://schemas.opengis.net/gml/2.1.2.1/feature.xsd

ShapeChange "No application schema found" error.

Appendix B FAQ

Appendix C Glossary/Acronyms

For the purposes of this document, the following terms and definitions have been taken from the OpenGIS Web Feature Service 2.0 Interface Standard, Open Geospatial Consortium Inc.

attribute <XML>
name-value pair contained in an element (4.6)
[ISO 19136:2007, definition 4.1.3]
NOTE In this document an attribute is an XML attribute unless otherwise specified.

client
software component that can invoke an operation (4.17) from a server (4.28)
[ISO 19128:2005, definition 4.1]

coordinate
one of a sequence of n numbers designating the position of a point in n-dimensional

space
[ISO 19111:2007, definition 4.5]

coordinate reference system
coordinate system (4.5) that is related to an object by a datum
[ISO 19111:2007, definition 4.8]

coordinate system
set of mathematical rules for specifying how coordinates (4.3) are to be assigned to points
[ISO 19111:2007, definition 4.10]

element <XML>
basic information item of an XML document containing child elements, attributes (4.1) and character data

[ISO 19136:2007, definition 4.1.23]

feature
abstraction of real world phenomena
[ISO 19101:2002, definition 4.11]
NOTE A feature can occur as a type or an instance. The term "feature type" or "feature instance" should be used when only one is meant.

feature identifier
identifier that uniquely designates a feature (4.7) instance OGC 09-025r1 and ISO/DIS 19142

filter expression
predicate expression encoded using XML
[ISO 19143, definition 4.11]

interface
named set of operations (4.17) that characterize the behaviour of an entity
[ISO 19119:2005, definition 4.2]
join predicate
filter expression (4.9) that includes one or more clauses that constrain properties from two different entity types
[ISO 19143, definition 4.16]
NOTE In this International Standard, the entity types will be feature (4.7) types.

join tuple
set of two or more object instances that satisfy a filter that includes join predicates (4.11)
NOTE In this International Standard, the object instances will be feature (4.7) instances.

local resource
resource that is under the direct control of a system
NOTE In this International Standard, the system is a web feature service and the resource is held in a data store that is directly controlled by that service.

locator attribute
attribute (4.1) whose values is a reference to a local resource (4.13) or remote resource (4.20)
NOTE In XML, this attribute is commonly called an href and contains a URI reference to the remote resource (see W3C XLink).

Multipurpose Internet Mail Extensions (MIME) type
media type and subtype of data in the body of a message that designates the native representation (canonical form) of such data
[IETF RFC 2045:1996]

namespace <XML>
collection of names, identified by a URI reference which are used in XML documents as element (4.6) names and attribute (4.1) names
[W3C XML Namespaces:1999]

operation
specification of a transformation or query that an object may be called to execute
[ISO 19119:2005, definition 4.3] OGC 09-025r1 and ISO/DIS 19142 6 Copyright © 2010 Open Geospatial Consortium

property
facet or attribute of an object, referenced by a name
[ISO 19143:2010, definition 4.21]

resource
asset or means that fulfils a requirement
[ISO 19143, definition 4.23]
NOTE In this International Standard, the resource is a feature (4.7), or any identifiable component of a feature (e.g. a property of a feature)

remote resource
resource that is not under direct control of a system
NOTE In this International Standard, the system is a web feature service. The resource is not held in any data store that is directly controlled by that service and thus cannot be directly retrieved by the service.

request
invocation of an operation (4.17) by a client (4.2)
[ISO 19128:2005, definition 4.10]

relocate
<reference> update a reference to a resource that has been moved or copied to a new location
EXAMPLE A server (4.28) is generating a response (4.24) to a GetFeature request (4.21), it has to copy a referenced feature (4.7) into the response document and the server has to "relocate" the original link contained in the referencing feature to the copy placed in the response document.

resolve
retrieval of a referenced resource and its insertion into a server-generated response document
NOTE The insertion may be accomplished by either replacing the reference in-line with a copy of the resource or by relocating the reference to point to a copy of the resource that has been placed in the response document.

response
result of an operation (4.17) returned from a server (4.28) to a client (4.2)
[ISO 19128:2005, definition 4.11]

response model
schema (4.26) defining the properties of each feature (4.7) type that can appear in the response (4.24) to a query operation (4.17)
NOTE This is the schema of feature types that a client (4.2) can obtain using the DescribeFeatureType operation (see Clause 9).

schema
formal description of a model OGC 09-025r1 and ISO/DIS 19142
[ISO 19101:2002, definition 4.25]
NOTE In general, a schema is an abstract representation of an object's characteristics and relations to other objects. An XML schema represents the relationship between the attributes (4.1) and elements (4.6) of an XML object (for example, a document or a portion of a document).

schema <XML Schema>
collection of schema (4.26) components within the same target namespace (4.16)
[ISO 19136:2007, definition 4.1.54]
EXAMPLE Schema components of W3C XML Schema are types, elements (4.16), attributes (4.1), groups, etc.

server
particular instance of a service (4.29)
[ISO 19128:2005, definition 4.12]
service
distinct part of the functionality that is provided by an entity through interfaces (4.10)
[ISO 19119:2005, definition 4.1]

service metadata
metadata describing the operations (4.17) and geographic information available at a server (4.28)
[ISO 19128:2005, definition 4.14]

traversal
<XML>
using or following an XLink link for any purpose
[W3C XLink:2001]

tuple
ordered list of values
[ISO 19136:2007, definition 4.1.63]
NOTE In this International Standard, the order list will generally be a finite sequence for features (4.7), each of a specific feature type.

Uniform Resource Identifier
unique identifier for a resource, structured in conformance with IETF RFC 2396
[ISO 19136:2007, definition 4.1.65]
NOTE The general syntax is <scheme>::<scheme-specified-part>. The hierarchical syntax with a namespace (4.16) is <scheme>://<authority><path>?<query> OGC 09-025r1 and ISO/DIS 19142

Abbreviated terms

CEOP Critical Earth Observation Priorities

CEOS Committee on Earth Observation Satellites

CGI Common Gateway Interface

CIM Core ISO Metadata

CRS Coordinate Reference System

CSR Components and Services Registry

DCP Distributed Computing Platform

DCP Distributes Computer Platforms

DSTF Data Sharing Task Force

DTD Document Type Definition

EO Earth Observation

EOP Earth Observation Product

EPSG European Petroleum Survey Group

FES Filter Encoding Specification

GCI CT GEOSS Common Infrastructure Coordination Team

GEO Group on Earth Observations

GEOSS Global Earth Observing System of Systems

GIGAS Project Global Earth Observing System of Systems

GML Geography Markup Language

HTTP Hypertext Transfer Protocol

HTTPS Secure Hypertext Transfer Protocol

ILAF OGC

ISO International Organisation for Standardisation

IETF Internet Engineering Task Force

KVP Keyword-value pairs

MIME Multipurpose Internet Mail Extensions

OGC Open Geospatial Consortium

OWS OGC Web Service

PSC The MapServer Project Steering Committee

RPC Remote Procedure Call

REST Representational State Transfer

SDI Spatial Data Infrastructure (1990's)

SIF Standards and Interoperability Forum

SQL Structured Query Language

SOAP Simple Object Access Protocol

UML Unified Modelling Language

URI Uniform Resource Identifier

URL Uniform Resource Locator

URN	Uniform Resource Name
VSP	Vendor Specific Parameter
WFS	Web Feature Service
WMS	Web Map Service
WCS	Web Coverage Service
WSDL	Web Services Description Language
W3C	World Wide Web Consortium
XML	Extensible Markup Language
Web Aliases	Web-accessible directories

Appendix E Summary of Important Links