

O P E N R S C O E U O P P O R T U N I T Y

## Summer School

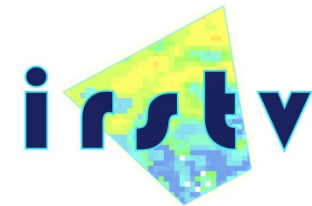
28<sup>th</sup> June - 9<sup>th</sup> July, 2010. Girona



Erasmus IP

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

### Partners Contribution



### Partner Collaboration



Universitat de Girona  
Departament de Geografia

O P E N R S C O E U O P P O R T U N I T Y

## XML, and GIS Web Services

Jeremy Morley,

Dr. Amir Pourabdollah

University of Nottingham

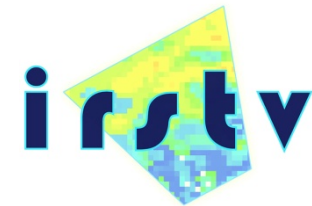
5<sup>th</sup> July, 2010. Girona



Erasmus IP

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Partners Contribution

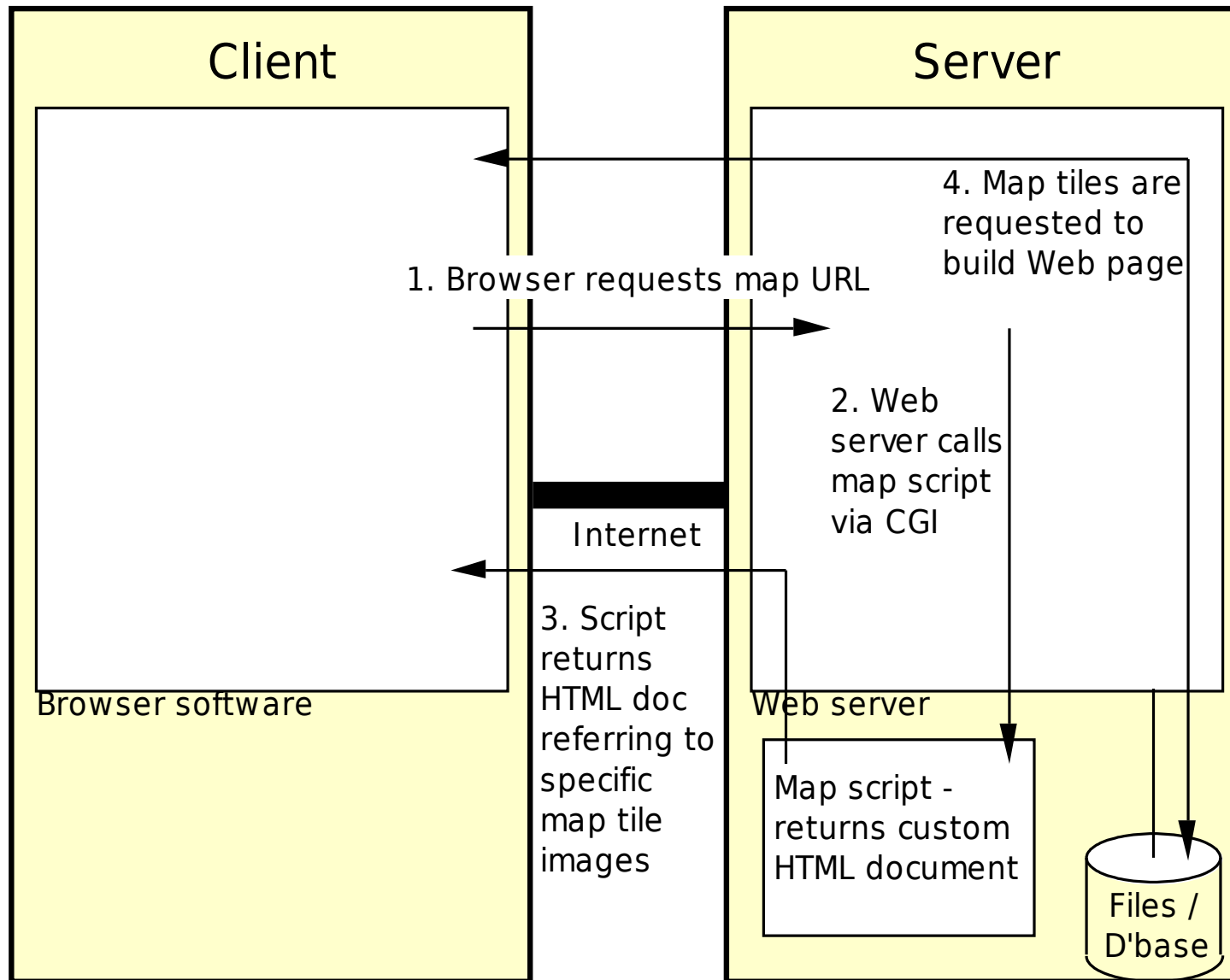


Partner Collaboration

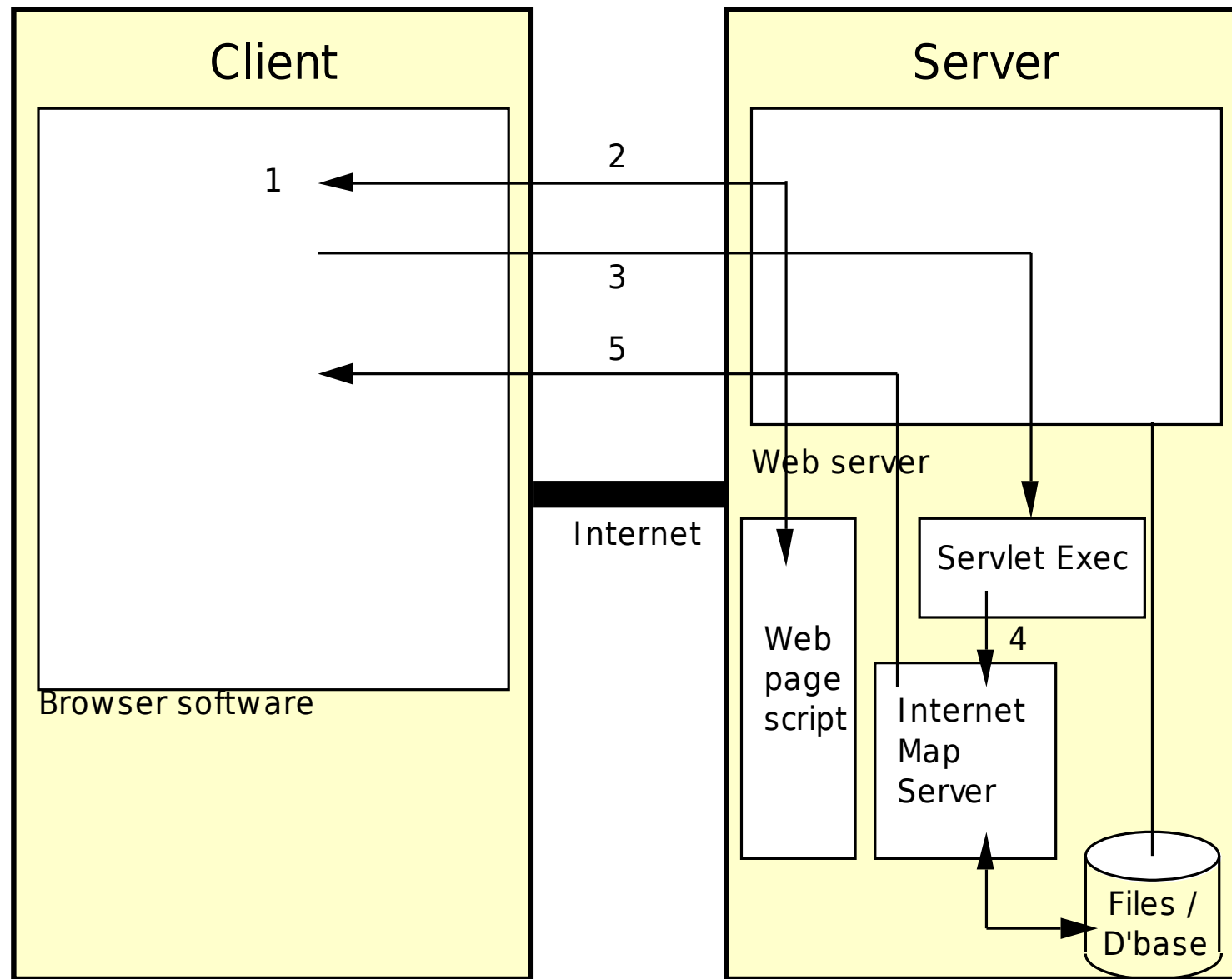


Universitat de Girona  
Departament de Geografia

# Simple server-side maps

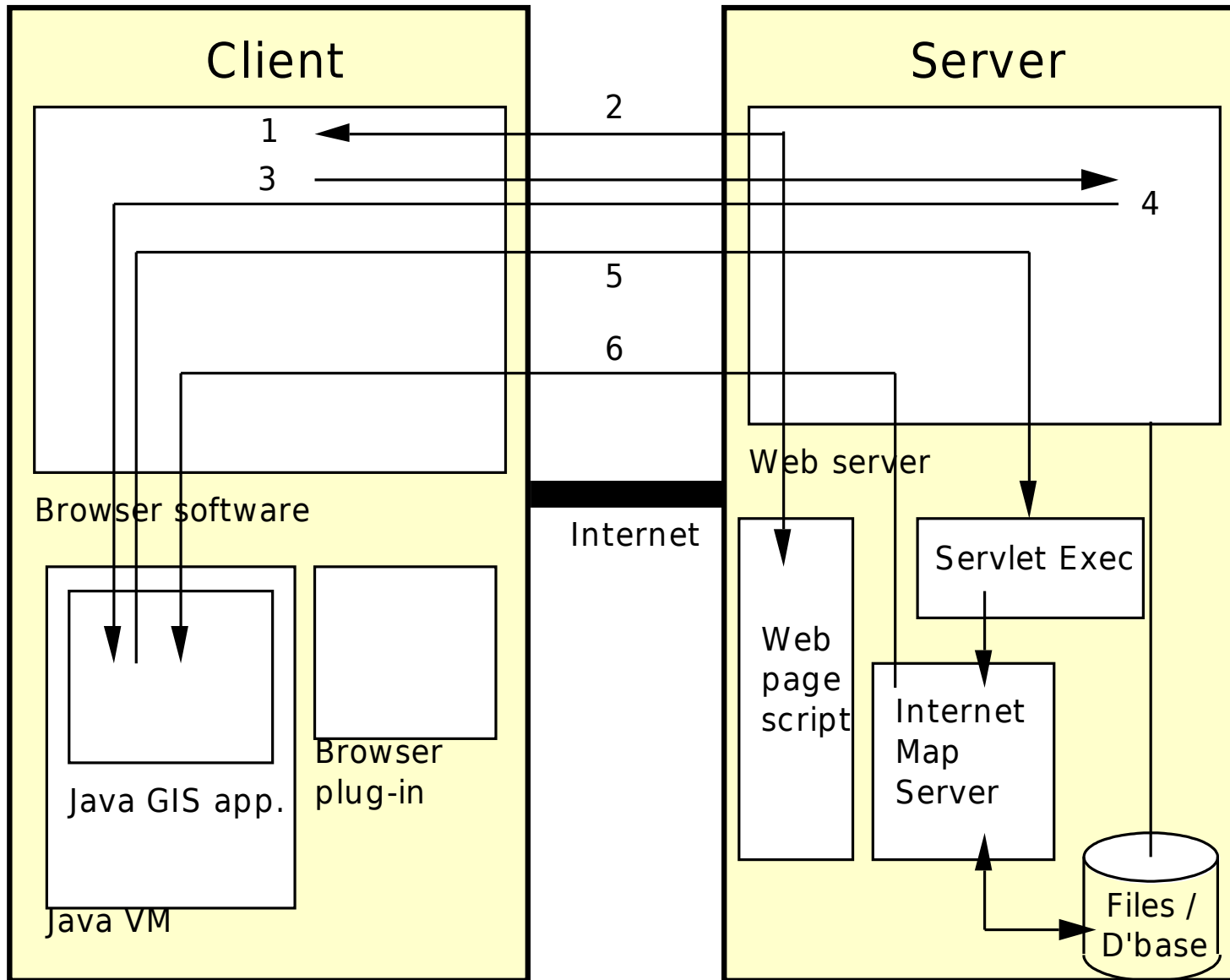


# Server-side interactive mapping



e.g. OGC WMS

# Client/server system



# Today

A lecture in two halves:

1. to introduce two important Web technologies:

- Hypertext Markup Language (HTML)
- Extensible Markup Language (XML)

1. to introduce the work of the Open Geospatial Consortium (OGC):

- Geographic Markup Language (GML)
- OpenGIS Web Services (OWS)

# HTML - A Markup Language

- HTML: Hypertext Markup Language
- Original concept: a markup system
- Document script contains stylistic suggestions
- Browser system interprets suggestions and lays out the page according to system capabilities
- Capabilities vary!
- HTML is not a page-layout language

# HTML - Tags

- Style marks are provided by tags
- A tag is a keyword plus optional attributes enclosed in angle brackets, e.g.

```
<A HREF="page2.html"> . . . </A>
```

- Most (but not all) styles require an opening and closing tag - a closing tag is marked by a slash, e.g.

```
<STRONG> . . . </STRONG>
```

- Some tags provide single-point effects and require no closing (or self-closing) tag, e.g.

```
<IMG SRC="head.gif" />
```

# Basic Document Structure

```
<HTML>
<!-- Comment text -->
<HEAD>
<!-- Header section -->
<TITLE> A simple document </TITLE>
</HEAD>
<BODY>
<!-- Main body of the document -->
This is the body of the text with
some <B>bold</B> text.
</BODY>
</HTML>
```

# XML – a data language

- XML (**eXtensible Markup Language**) is a serialization of textual data mostly used in the Web/Internet.
  - Labelled and Hierarchical data structure
- XML allows definition of tags, so we can describe our data in a way appropriate to its content

# An XML document!

(from w3schools.com)

```
<?xml version="1.0"?>  
<note>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this  
  weekend!</body>  
</note>
```

Also <http://www.w3schools.com/XML/simple.xml>

# XML - elements

- Elements are parts of a document
- Elements can be containers, with a mixture of *text* and *other elements*.
  - This element contains only text:  
`<description>This is text contained inside an element</description>`  
This element ("outer") contains both text and an element ("inner"):  
`<outer>this is text<inner>more text</inner>still more text</outer>`
- Elements can have attributes.
  - `<description language="English" encoding="Unicode">This is ...`
- Some elements are empty, and contribute information by their position and attributes.
  - `<outer>an element can be empty: <rightthere/></outer>`

Source– "Learning XML (Guide to) Creating Self-Describing Data", Erik T. Ray, January 2001, Publisher: O'Reilly & Assocs. Inc. ISBN 0-596-00046-4, 368 pages



# XML – elements (2)

- Elements are identified in XML files by tags, similar to HTML, and can have attributes
- How can a computer interpret XML? Two aspects:
  - Rules for XML document structure
  - Element & attribute definitions (**Schema**)
- Hence a 'good' XML document:
  - Follows the structure rules – is "**well formed**"
  - Conforms to its schema – is "**valid**"

# XML structure rules

1. Opening & closing parts of a tag set must contain the same name in the same case:
  - `<tag>...</tag>` *versus* `<tag>...</TAG>`
1. Empty tags are abbreviated:
  - `<br/>`
1. Opening tags must be balanced by closing tags
2. Tags must be nested correctly, e.g.
  - `<outer>outside <inner>in</inner></outer>`
1. All attribute values must be enclosed in quotes:
  - `<element id="value"> ... </element>`

# Example XML document (2)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE Book-Review SYSTEM "Book-Review.dtd">
<!-- End of prolog -->
<!-- Beginning of body -->
<Book-Review>
This book was great!
</Book-Review>
<!-- End of body -->
```

Adapted from: XML for Dummies, 2<sup>nd</sup> edition

# XML docs – standard elements

1. XML declaration
2. Document type declaration
3. Body
  - Starts with the *root element* (defines the basic sort of data – encloses the body of the document)
  - Contains the data content, structured into elements
  - Also comments: `<! - - ... - - >`

# XML Namespaces

- A prefix mechanism to avoid naming conflicts when using multiple vocabularies
- Optionally declared as an attribute of an element (or root element)
- Binding a prefix with a URL, just for uniqueness.

```
<book xmlns:lib="http://www.library.com">  
  <lib:Title>Sherlock Holmes</lib:Title>  
  <lib:Author>Arthur Conan Doyle</lib:Author>  
</book>
```

Source: [http://www.oracle.com/technology/pub/articles/srivastava\\_namespaces.html](http://www.oracle.com/technology/pub/articles/srivastava_namespaces.html)

# XML Schema

- The recommended method to define the validation rules of a XML instance (DTD is an alternative)
- Defining XML Schema *in XML* (unlike DTD)
- <schema> (root element), <element>, <attribute>, <sequence>, etc.
- .XSD files
- Document type declaration line in an XML instance defines which XSD to use.
- XSD Namespace (usually as “xs” prefix):  
<http://www.w3.org/2001/XMLSchema>
- Optionally defines a “target namespace” to be used in any XML instance.

# XML Schema

- Simple elements
  - `<xs:element name="xxx" type="yyy"/>`
  - type: `xs:string`, `xs:decimal`, `xs:integer`, ...
- Attributes of elements:
  - `<xs:attribute name="xxx" type="yyy"/>`
- Complex (nested) elements:
  - `<xs:complexType>...`
- Various restriction tags and attributes;
- And much more... full reference on <http://www.w3.org/2001/XMLSchema>

# XSD Example

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="shiporder">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="orderperson" type="xs:string"/>
        <xs:element name="shipto">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="address" type="xs:string"/>
              <xs:element name="city" type="xs:string"/>
              <xs:element name="country" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="item" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title" type="xs:string"/>
              <xs:element name="note" type="xs:string" minOccurs="0"/>
              <xs:element name="quantity" type="xs:positiveInteger"/>
              <xs:element name="price" type="xs:decimal"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="orderid" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Source, incl. other xsd:

[http://www.w3schools.com/Schema/schema\\_example.asp](http://www.w3schools.com/Schema/schema_example.asp)

# Example XML document

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

# XML tools online

- On-line tool to check if document well-formed:
  - <http://www.xml.com/pub/a/tools/ruwf/check.html>
- On-line tool to check if document well-formed and valid (i.e. checks against DTD too):
  - <http://www.xmlvalidation.com>
- Internet Explorer 5+ and the Firefox browser will both parse and display XML documents.
- Microsoft XSD reference:  
<http://msdn.microsoft.com/en-us/library/ms256235.aspx>

# OpenGeospatial Consortium\* and OpenGIS Web Services

\* Previously, the OpenGIS Consortium!



# OGC Objectives

1. Promote the use of “interoperable geoprocessing” throughout the Information Technology marketplace.

- WWW and e-commerce

- Telecommunications services

- Enterprise computing

1. Synchronize “geoprocessing technology” with commercial “Information Technology standards”

- Open systems

- Distributed processing

- Object oriented design

- Service-oriented technology

# OGC Objectives

3. Arrange cooperation of “GI product suppliers” and “GI users” to develop interoperable software interfaces.

Development v. requirements for component architectures

User-driven “business model” for interoperable applications

3. Involve the entire community in the “interoperability process”

Industry

Government

Academia

Standards bodies

# OGC Objectives

5. Provide an “industry forum” for “partnerships” and cooperative business development projects.

- Technology teams

- Joint marketing

- Industry forums

- Partnership development



# OpenGeospatial Technologies

- GML – The Geography Markup Language
  - Used as an interoperable standard for transmitting geographic data
  - Versions 2.1.x and 3.2.1 are important to us
- OpenGIS Web Services (OWS)
  - Web Map Service (WMS)
  - Web Coverage Service (WCS)
  - Web Feature Service (WFS)

# OWS - details

- **WMS – Web Map Service**
  - Provides rendered images of maps
  - Current version: 1.3
  - Also, Web Map Tile Service: WMTS 1.0
- **WFS – Web Feature Service**
  - Provides vector data on demand
  - Current version: 1.1
- **WCS – Web Coverage Service**
  - Provides raster data on demand
  - Current version: 1.1

# OGC Technologies (2)

- The OGC publish "specifications" that have been agreed by OGC members
- Current specifications can be found at:  
<http://www.opengeospatial.org/standards/is>
- These are implementation specs
  - written for a more technical audience and detail the interface structure between software components
- Predicated on abstract specs
  - the conceptual foundation for most OGC specification development activities
  - <http://www.opengeospatial.org/standards/as>

# OGC Web Mapping

- Three different services of interest:
  - Web Map Service (WMS), Web Feature Service (WFS), Web Coverage Service (WCS)
- All are designed as **server-side** mapping methods
- Map (and other) requests are made through standard HTML methods
  - URL
  - Parameters supplied within URL (GET) or hidden in document (POST)
  - One request gives one response

# OGC Web Mapping (2)

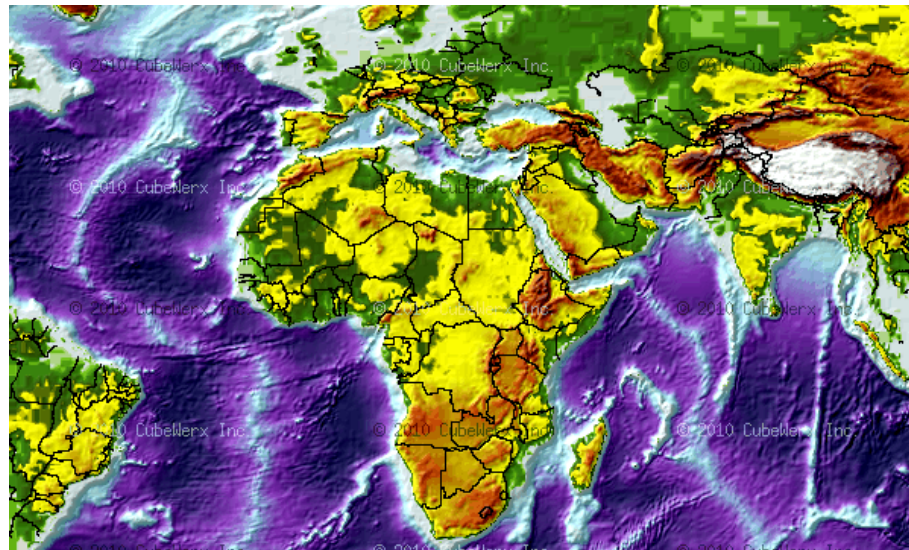
- In each URL, we can request one operation
- All four services provide a common operation: GetCapabilities
- GetCapabilities is important as it returns info on:
  - general information about the service itself and
  - specific information about the available datasets
- General information includes OGC spec level of server, spatial reference systems supported
- Specific information covers layers and their properties

# OGC Web Mapping (3)

- In the "GET" style of parameters, the parameters are included in the URL after a question mark (?), each separated by ampersands (&)
- Example URLs:
  - [http://atlas.gc.ca/cgi-bin/atlaswms\\_en?REQUEST=GetCapabilities](http://atlas.gc.ca/cgi-bin/atlaswms_en?REQUEST=GetCapabilities)
  - <http://demo.cubewerx.com/demo/cubeserv/cubeserv.cgi?SERVICE=WMS>
- Response to GetCapabilities will be an XML document

# WMS output example

[http://demo.cubewerx.com/demo/cubexplor/cubeserv.cgi?SERVICE=&CRS=EPSG%3A4326&BBOX=-36.78849055258625,-58.66810168664374,62.81670815058624,107.3405628186437&WIDTH=600&HEIGHT=360&LAYERS=Foundation.ETOPO2,Foundation.POLBNDL\\_1M](http://demo.cubewerx.com/demo/cubexplor/cubeserv.cgi?SERVICE=&CRS=EPSG%3A4326&BBOX=-36.78849055258625,-58.66810168664374,62.81670815058624,107.3405628186437&WIDTH=600&HEIGHT=360&LAYERS=Foundation.ETOPO2,Foundation.POLBNDL_1M)



# Other WMS Examples

- <http://iceds.ge.ucl.ac.uk/cgi-bin/icedswms?SERVICE=WMS&REQUEST=GetCapabilities>
- [http://iceds.ge.ucl.ac.uk/cgi-bin/icedswms?VERSION=1.1.1&REQUEST=GetMap&SRS=EPSG:4326&BBOX=-30,-35,112.86,65&WIDTH=600&HEIGHT=420&LAYERS=LANDSAT5&FORMAT=image/jpeg&BGCOLOR=0xffffffff&TRANSPARENT=TRUE&EXCEPTIONS=application/vnd.ogc.se\\_inimage](http://iceds.ge.ucl.ac.uk/cgi-bin/icedswms?VERSION=1.1.1&REQUEST=GetMap&SRS=EPSG:4326&BBOX=-30,-35,112.86,65&WIDTH=600&HEIGHT=420&LAYERS=LANDSAT5&FORMAT=image/jpeg&BGCOLOR=0xffffffff&TRANSPARENT=TRUE&EXCEPTIONS=application/vnd.ogc.se_inimage)

# OGC Web Mapping (4)

- Basic data types returned from services:
- WMS (none *required* by spec):
  - GIF, PNG, JPEG
  - Scalable Vector Graphics (SVG) or Web Computer Graphics Metafile (WebCGM) formats.
- WFS:
  - Default: GML 2
- WCS:
  - GeoTIFF, HDF-EOS, DTED, NITF, GML
  - Described in response to DescribeCoverage request

# Practical

- You get to try:
  - Creating a simple well-formed XML file of your own
  - Try interacting manually with WMS and WCS servers
  - Incorporate OWS layers into uDig

# GML

- GML is one of the older implementation standards
- Has evolved with time to add functionality
- Has also moved with the tide of XML
  - GML 1.0.0 – defined using DTDs
  - GML 2 – defined using XML Schema
  - GML 3 – defined using XML Schema, but includes more complex functionality, e.g. XLinks
- Reference:
  - OGC GML 3.2.1 specification document:  
[http://portal.opengeospatial.org/files/?artifact\\_id=20509](http://portal.opengeospatial.org/files/?artifact_id=20509)
  - Schema and DTDs: <http://schemas.opengis.net/gml/>

# GML 2.1.2

- "XML for Simple Features"
- Spec: <http://www.opengeospatial.org/docs/02-069.pdf>
- Simple features
  - " 'simple' geometries for which coordinates are defined in two dimensions and the delineation of a curve is subject to linear interpolation " (from spec.)
- Simple vector data only + feature collections

# Some GML 2.1.2 design goals

- Permit the easy integration of spatial and non-spatial data, separated from data presentation
- Data is XML-encoded;
- Be able to readily link spatial (geometric) elements to other spatial or non-spatial elements.
- Provide a set common geographic modelling objects to enable interoperability of independently developed applications.



# Relevance

- Ordnance Survey of GB use GML as the (only) format for supply of MasterMap data
- The OS take GML 2.1.2 as their starting point
- They use the GML 2.1.2 schema as a basis but build more complex feature models on top.  
Hence the large scale MasterMap product has its own associated XML Schema definitions:
  - <http://www.ordnancesurvey.co.uk/oswebsite/xml/schema/index.html>
  - From postCode through to heightAboveDatum types

# GML 3.0.0

- GML 3.0.0 spec. was published on 29<sup>th</sup> January 2003
- Addresses the following needs that were not addressed or adequately met by the previous versions:
  - represent geospatial phenomena in addition to simple 2D linear features, including features with complex, non-linear, 3D geometry, features with 2D topology, features with temporal properties, dynamic features, coverages, and observations;
  - provide more explicit support for properties of features and other objects whose value is complex
  - represent spatial and temporal reference systems, units of measure and standards information;
  - use reference system, units and standards information in the representation of geospatial phenomena, observations, and values;
  - represent default styles for feature and coverage visualization.



# GML 3.0.x (cont.)

- Note in particular the addition of coverages
- Coverages are the OGC raster model
- GML 3.0.0 schema are eight times the size of the GML 2.1.2
- However not all GML 3.0.0 schema definitions will be needed for most applications
- It is possible to take a subset to build on
- Ref.: "Geography Mark-Up Language: foundation for the geo-web", R. Lake, D.S. Burggraf, M. Trninic, L. Rae; John Wiley & Sons., Chichester. ISBN:0-470-87154-7

# Standards Convergence

- ISO (the International Organisation for Standardization) has been working on geographic data
  - ISO TC/211
- 3.0.0 brings GML in line with:
  - ISO DIS 19107:  
Geographic Information – Spatial Schema
  - ISO DIS 19108:  
Geographic Information – Temporal Schema
  - ISO DIS 19118: Geographic Information – Encoding
  - ISO DIS 19123: Geographic Information – Coverages

# GML 3.1+

- GML Version 3.1.0:
  - adds new geometries
  - is more compliant with the ISO/TC 211 family of specifications, and
  - contains some items for increased efficiency and simplicity
- GML Version 3.1.1:
  - fixes bugs (e.g. the schema are now valid!)
  - first 'working' version of GML 3...
- GML Version 3.2.1:
  - ISO 19136 standard (2007) + fixed bugs

# *De facto* standards

- Google
  - Google Maps
  - Google Earth
- Bing Maps
  - was Microsoft Virtual Earth, Windows Live Local !
- Important because they provide simple Javascript APIs to use maps on your own web site.
  - <http://www.google.com/apis/maps/documentation/>
  - <http://dev.live.com/virtualearth/sdk/>

# KML

- Originally, Keyhole Markup Language: XML language focused on geographic visualization
- An XML application KML to:
  - Specify icons and labels to identify locations on the planet surface
  - Create different camera positions to define unique views for each of your features
  - Use image overlays attached to the ground or screen
  - Define styles to specify feature appearance
  - Write HTML descriptions of features, including hyperlinks and embedded images
  - Use folders for hierarchical grouping of features
  - Dynamically fetch and update KML files from remote or local network locations
  - Fetch KML data based on changes in the 3D viewer
  - <http://code.google.com/apis/kml/>

# Example KML

```
<?xml version="1.0" encoding="UTF-8" ?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Folder>
    <LookAt>
      <longitude>5.08</longitude>
      <latitude>44.54</latitude>
      <range>3300000</range>
    </LookAt>
    <name>ICEDS hill-shaded SRTM service</name>
    <open>1</open>
    <NetworkLink>
      <name>SRTM hillshaded topography</name>
      <Url>
        <href>http://mapdev.esrin.esa.int:8080/gproxy.php?
REQUEST=GetMap&SERVICE=WMS&LAYERS=srtm&VERSION=1.1.1&FORMAT=JPEG&SER
VER=http://iceds.ge.ucl.ac.uk/cgi-bin/icedswms</href>
      </Url>
      <visibility>1</visibility>
      <viewRefreshMode>onStop</viewRefreshMode>
    </NetworkLink>
  </Folder>
</kml>
```

# Example KML (2)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Placemark>
  <name>SketchUp Model of Macky Auditorium</name>
  <description>University of Colorado, Boulder; model created by
  NoÃ«l Nemcik.</description>
  <Model id="model_4">
    <altitudeMode>relativeToGround</altitudeMode>
    <Location>
      <longitude>-105.272774533734</longitude>
      <latitude>40.009993372683</latitude>
      <altitude>0</altitude>
    </Location>
    <Link>
      <href>files/CU Macky.dae</href>
    </Link>
  </Model>
</Placemark>
</kml>
```

# KML's future

- Google have a beta version of KML 2.2
- Now adopted at version 2.2 by the OGC
- Hence after this version, KML will be developed within the OGC consensus system
  - hence not just Google driving future development
  - is this Google being generous, or tech. disruption?
- OGC plan to make KML & GML more compatible:  
<http://www.opengeospatial.org/standards/kml>