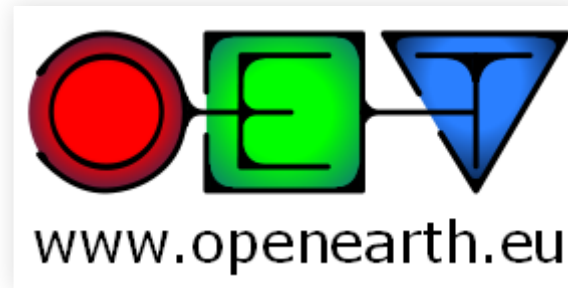


# OpenGLS Ser- vices

@ Delft Software Days 2014







Fedor Baart  
fedor.baart@deltares.nl





# Agenda

09:00 - 09:15	Introduction, Fedor Baart
09:15 - 10:30	PostGIS and Qgis, Frank Keppel
10:30 - 11:00	
11:00 - 12:30	Gridded data, Giorgio Santinelli
12:30 - 14:00	
14:00 - 15:30	Visualizing with KML, Kees den Heijer
15:30 - 16:00	
16:00 - 17:30	Services, Fedor Baart
17:30 - 18:30	



# History





# Open Source GIS

1980 Lagrid @ Westervelt

1983 Proj4 @ Evenden

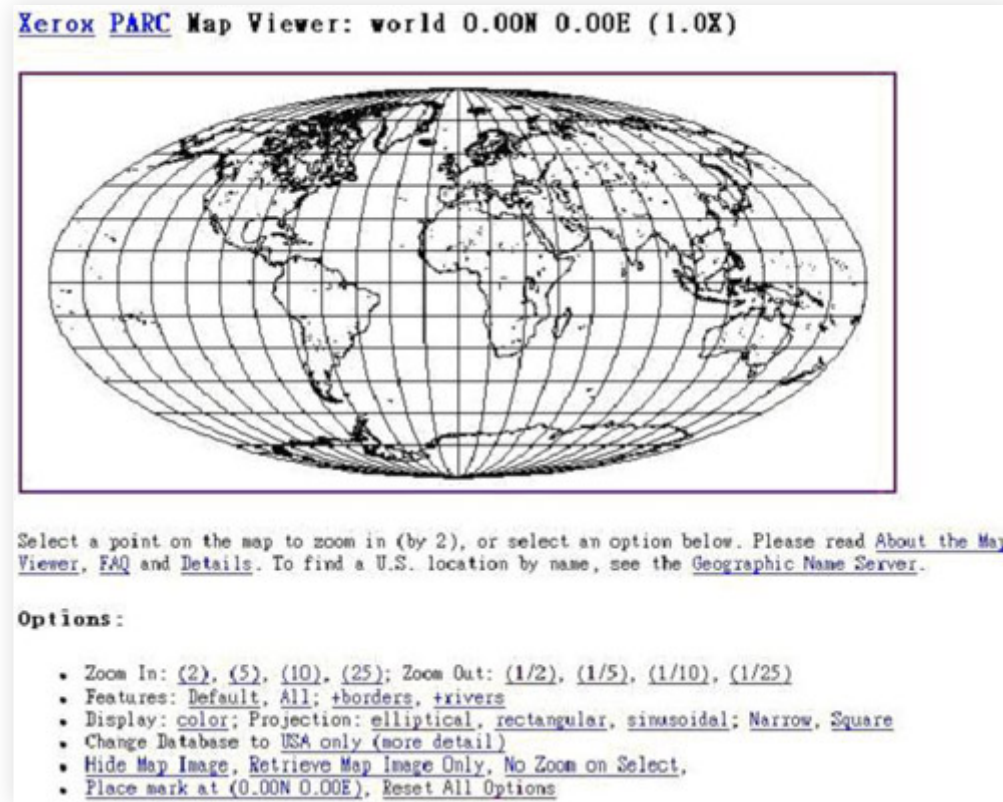
1983 GRASS GIS @ CERL

1994 www

1994 Open Geospatial Consortium



# First map



1993 @ Xerox



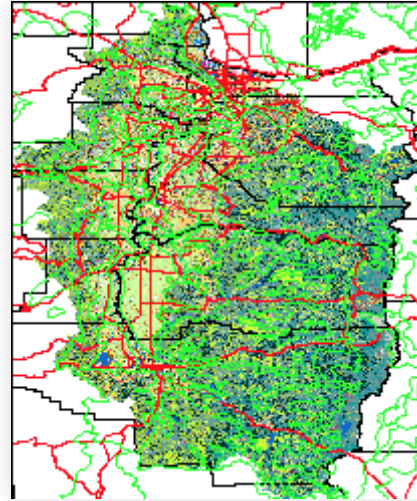
# Mapserver



1994 @ NASA/ForNet (open source since 1999)



# First process

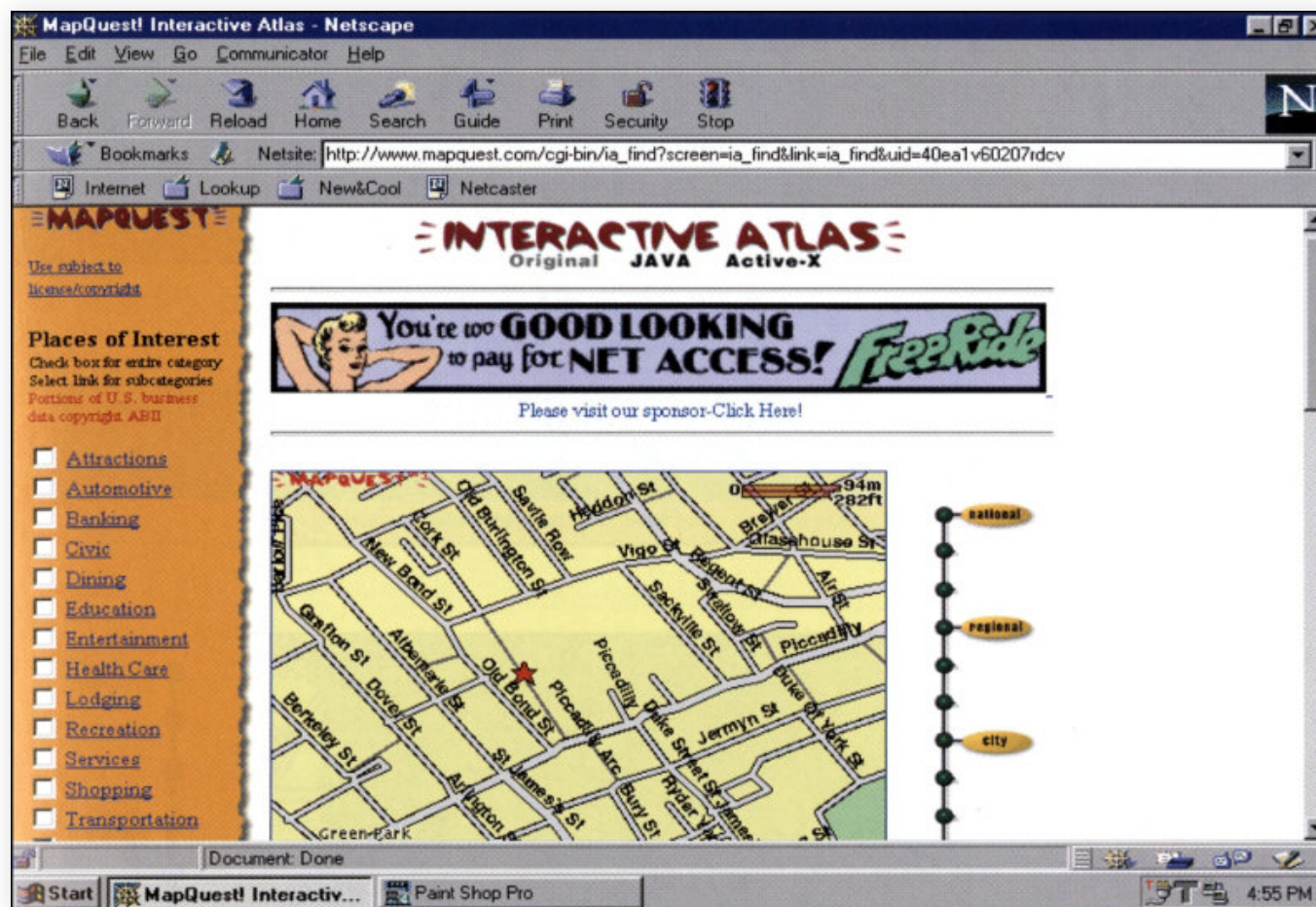


1995 @ Berkeley



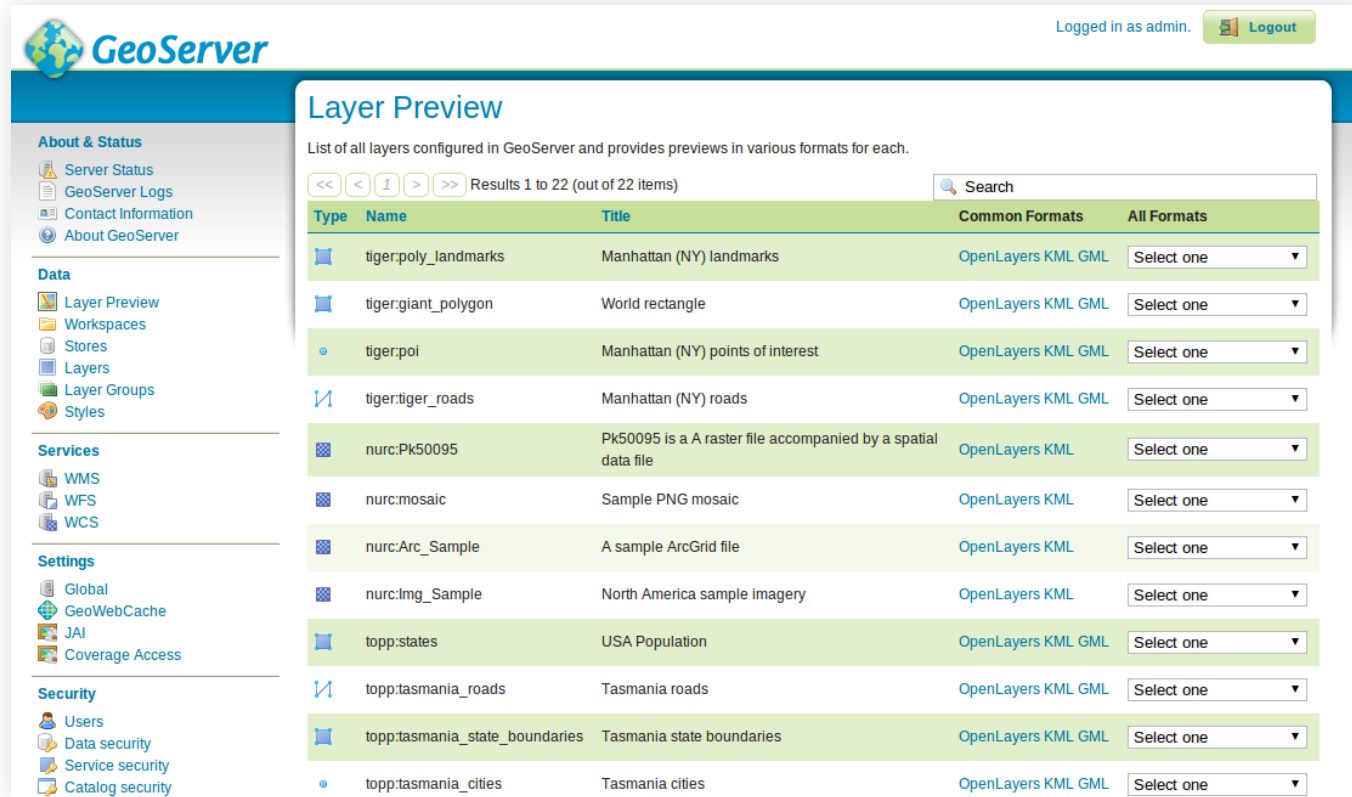


# Location based services: routing





# Spatial querying



The screenshot shows the GeoServer web interface. The top navigation bar includes the GeoServer logo, the text "Logged in as admin.", and a "Logout" button. The left sidebar contains a menu with sections: "About & Status" (Server Status, GeoServer Logs, Contact Information, About GeoServer), "Data" (Layer Preview, Workspaces, Stores, Layers, Layer Groups, Styles), "Services" (WMS, WFS, WCS), "Settings" (Global, GeoWebCache, JAI, Coverage Access), and "Security" (Users, Data security, Service security, Catalog security).

The main content area is titled "Layer Preview" and contains the text: "List of all layers configured in GeoServer and provides previews in various formats for each." Below this text is a pagination control showing "Results 1 to 22 (out of 22 items)" and a search input field.

Type	Name	Title	Common Formats	All Formats
	tiger:poly_landmarks	Manhattan (NY) landmarks	OpenLayers KML GML	Select one
	tiger:giant_polygon	World rectangle	OpenLayers KML GML	Select one
	tiger:poi	Manhattan (NY) points of interest	OpenLayers KML GML	Select one
	tiger:tiger_roads	Manhattan (NY) roads	OpenLayers KML GML	Select one
	nurc:Pk50095	Pk50095 is a A raster file accompanied by a spatial data file	OpenLayers KML	Select one
	nurc:mosaic	Sample PNG mosaic	OpenLayers KML	Select one
	nurc:Arc_Sample	A sample ArcGrid file	OpenLayers KML	Select one
	nurc:Img_Sample	North America sample imagery	OpenLayers KML	Select one
	topp:states	USA Population	OpenLayers KML GML	Select one
	topp:tasmania_roads	Tasmania roads	OpenLayers KML GML	Select one
	topp:tasmania_state_boundaries	Tasmania state boundaries	OpenLayers KML GML	Select one
	topp:tasmania_cities	Tasmania cities	OpenLayers KML GML	Select one

2000 @ Geoserver

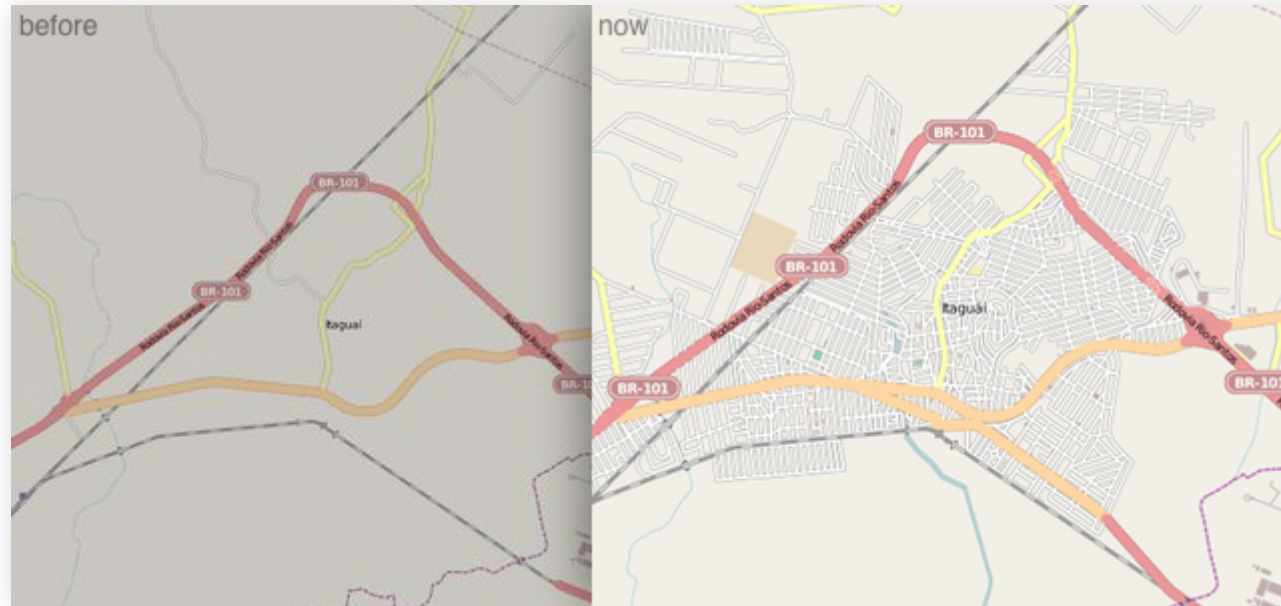


# NASA World Wind





# OpenStreetMap



2004 @ Steve Coast



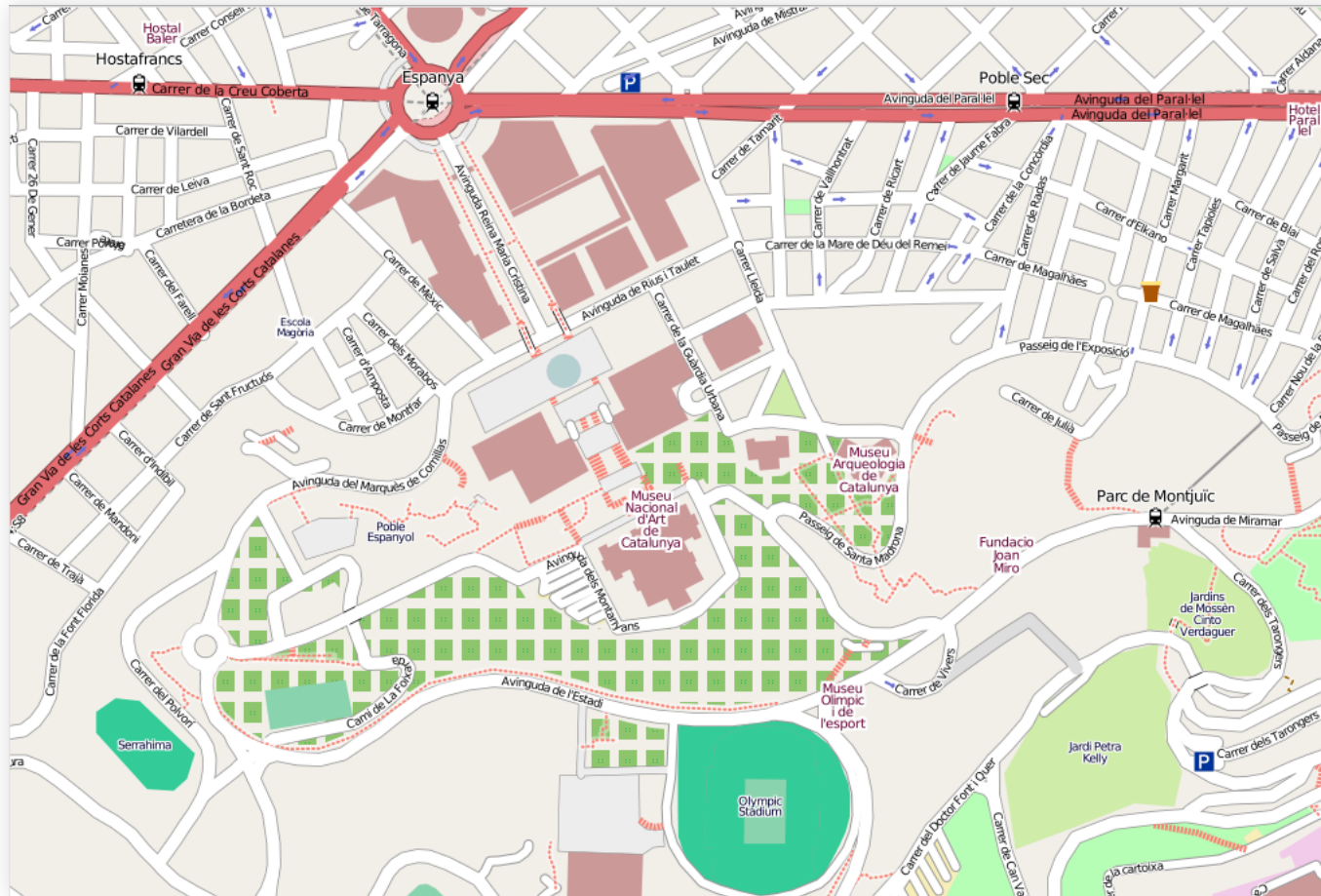


# OpenStreetMap

2004 @ Steve Coast



# Styling



2005 @ Mapnik



# Google Maps

Google Maps - from: 602 townsend street, san francisco, ca to: 345 park avenue, san jose, ca

http://maps.google.com/

Google Maps BETA

Maps Local Search Directions

Start address: 602 townsend street, san francisco, ca

End address: 345 park avenue, san jose, ca

Search

Maps Map - Satellite <sup>New!</sup>

Print Email Link to this page

Start address: 602 Townsend St  
San Francisco, CA 94103, USA

End address: 345 Park Ave  
San Jose, CA 95110, USA

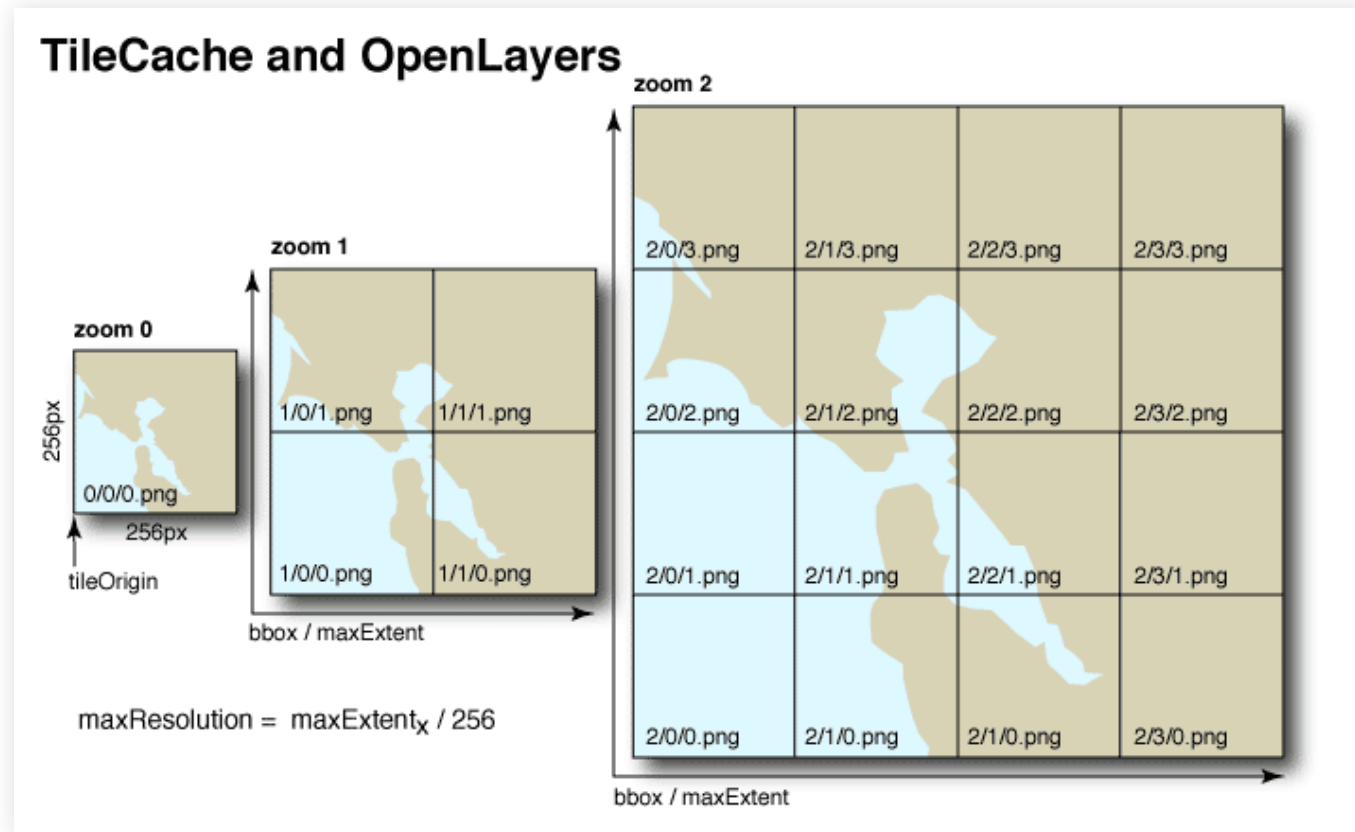
Distance: 47 mi (about 43 mins)

Reverse directions

0. Head **northeast** from **Townsend St** - go **0.0 mi**
1. Turn **right** at **7th St** - go **0.5 mi**
2. Bear **left** at **Mississippi St** - go **0.2 mi**
3. Turn **left** at **Mariposa St** - go **0.1 mi**
4. Turn **right** into the **I-280 S** entry ramp to **Daly City/San Jose** - go **2.1 mi**
5. Take the **US-101 S** ramp to **San Jose** - go **40 mi**
6. Bear **right** onto the **CA-87** ramp - go **0.3 mi**



# Tilecache

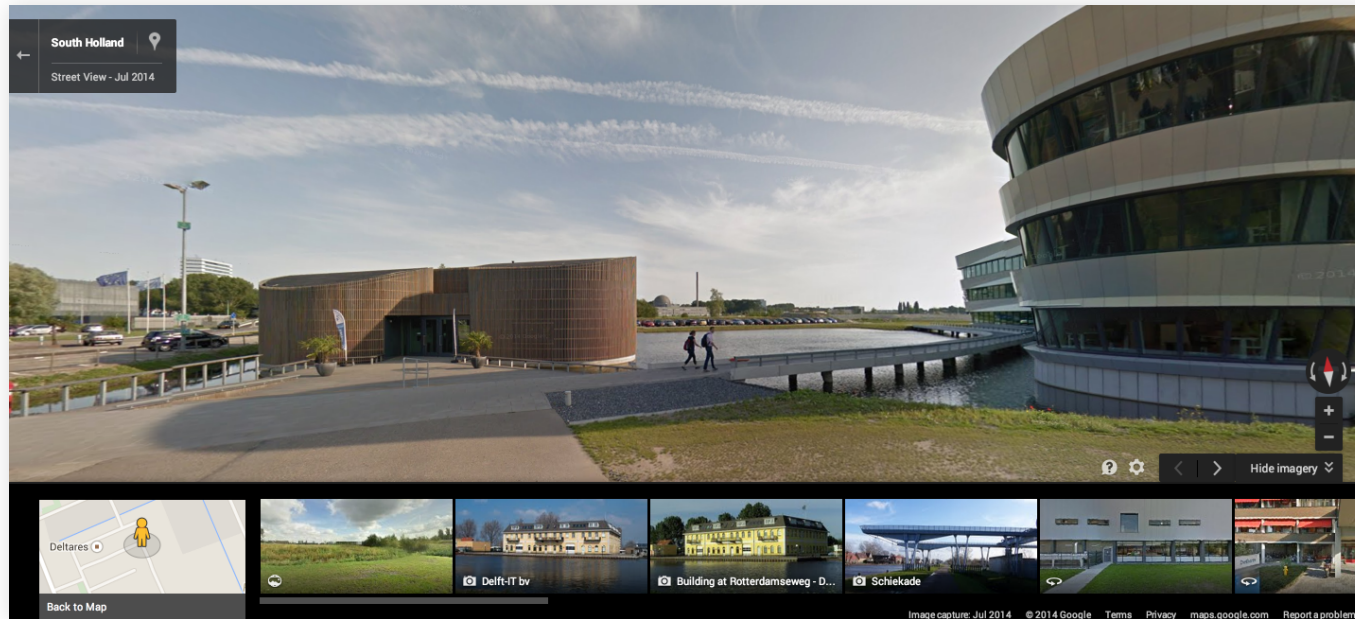


src: renderfast.com  
2006 @ Metacarta





# Streetview



2007 @ Google



# Sensors



2007 @ OGC



# INSPIRE

The screenshot shows the INSPIRE Geoportal website. At the top left is the European Commission logo. The main header reads "INSPIRE GEOPORTAL" and "Enhancing access to European spatial data". Below this is a navigation breadcrumb: "EUROPEAN COMMISSION > INSPIRE > INSPIRE GEOPORTAL > Discovery / Viewer". On the right, there are links for "Contact | Search | Legal notice" and a language dropdown set to "English (en)".

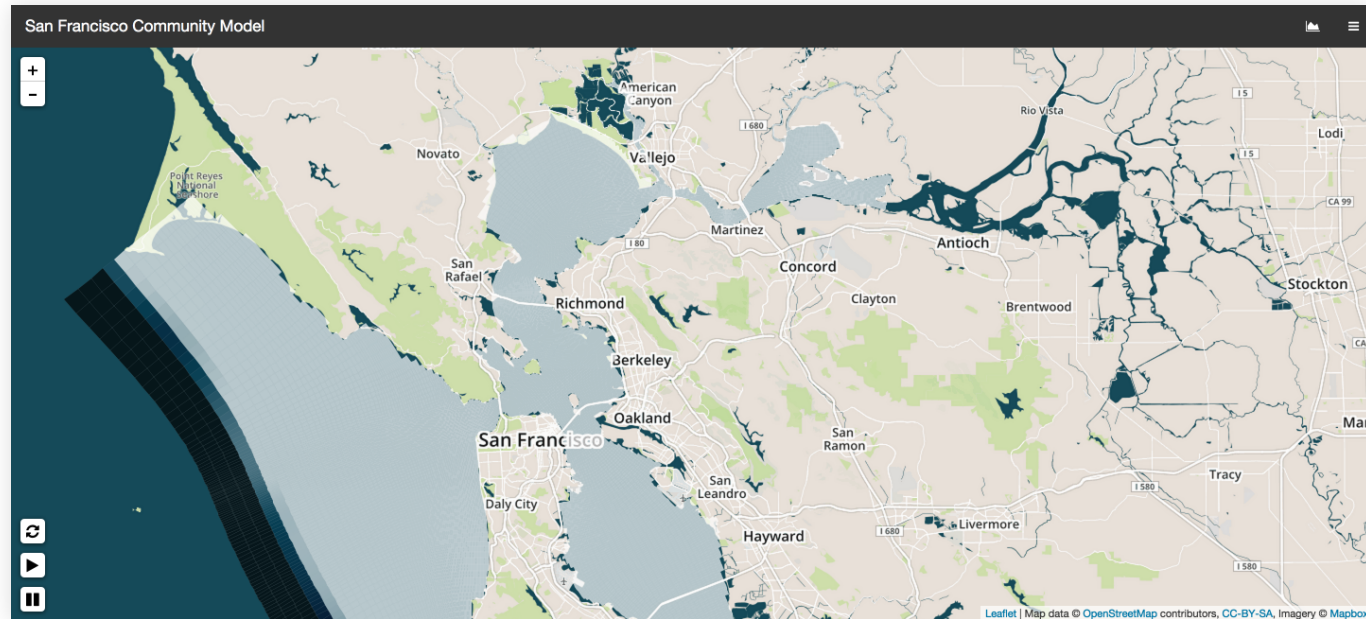
The main content area is divided into two columns. The left column features a search bar with "Find a place in: Europe" and a "Powered by GeoNames" label. Below the search bar is a map of Europe with a scale bar (1000 km / 1000 mi) and coordinates (25.14373, 71.52738). The right column has a search box with the placeholder "Enter your query in any language of the European Union" and an "Advanced Search" link. Below the search box are radio buttons for "datasets", "series", "services", "layers", and "download service datasets". A list of countries and categories follows, including Austria, Belgium, Croatia, Czech Republic, Denmark, Estonia, Finland, France, Greece, Iceland, Ireland, Latvia, Luxembourg, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, United Kingdom, and Addresses. Other categories include Administrative Units, Agricultural and Aquaculture Facilities, Area management/restriction/regulation zones and reporting units, Atmospheric Conditions, Buildings, Cadastral Parcels, Coordinate Reference Systems, Elevation, Energy Resources, Environmental Monitoring Facilities, Geographical Names, Geology, Habitats and Biotopes, Human Health and Safety, Hydrography, Land Cover, and Land Use. At the bottom of the right column, there are links for "Imagery / Base Maps / Earth Cover" and "Inland Waters".

At the bottom left of the main content area, it says "Active Layers: 0".

2007 @ EU



# Stylesheets

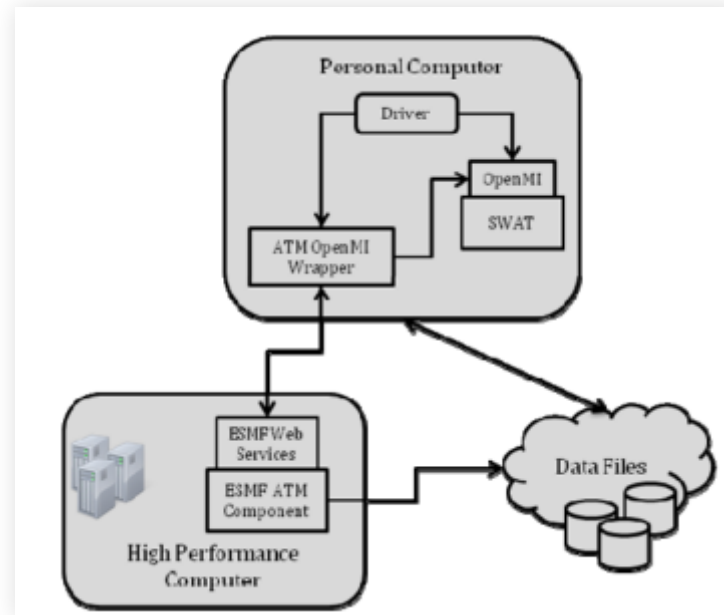


2008 @ Mapnik & Tilemill





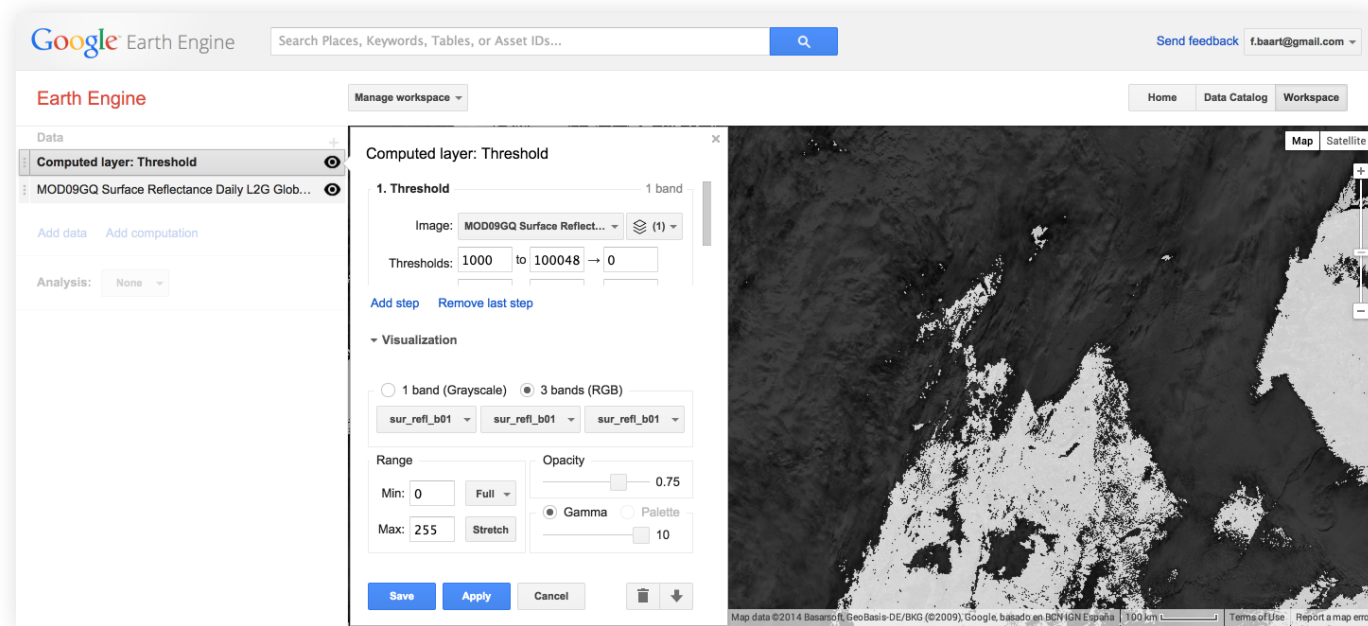
# Model as a service



2010 @ ESMF (Saint)



# Earth Engine



2010 @ Google



# Vectors and 3D

transitions  
2011 @ W3



# Vectors and 3D

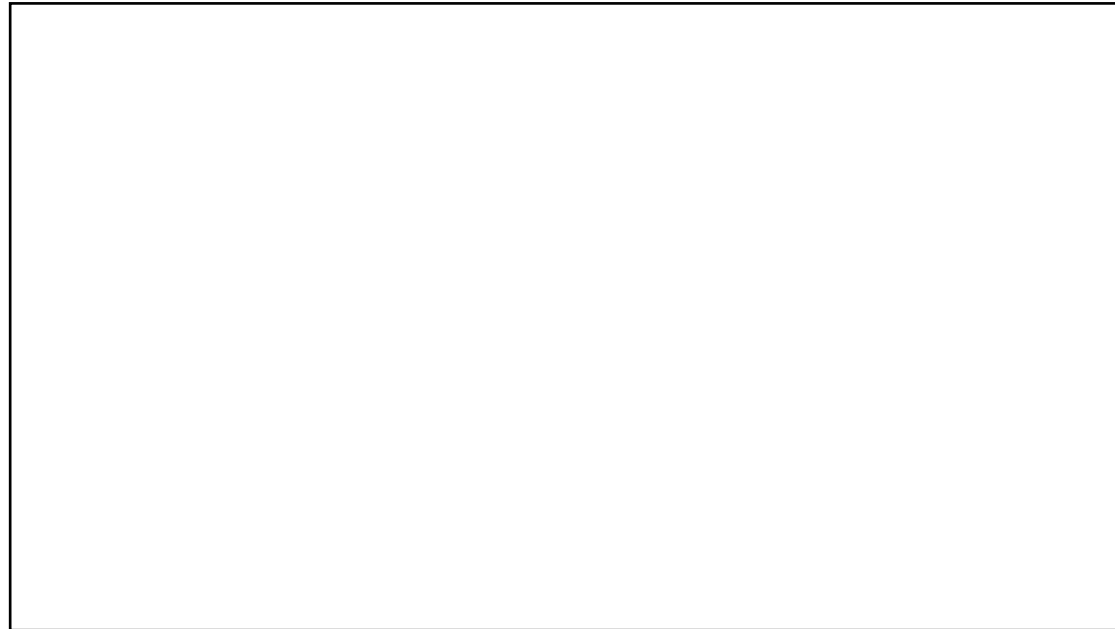


2011 @ W3





# Animations



2011 @ W3

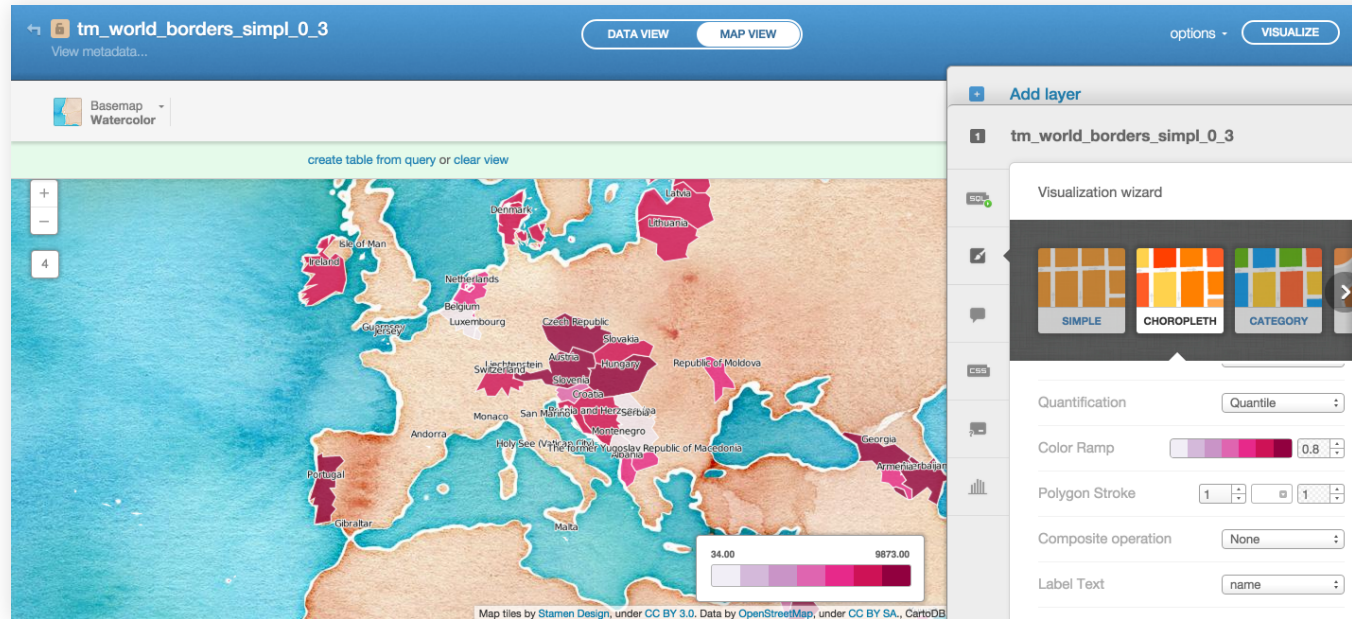


# Animations

2011 @ W3



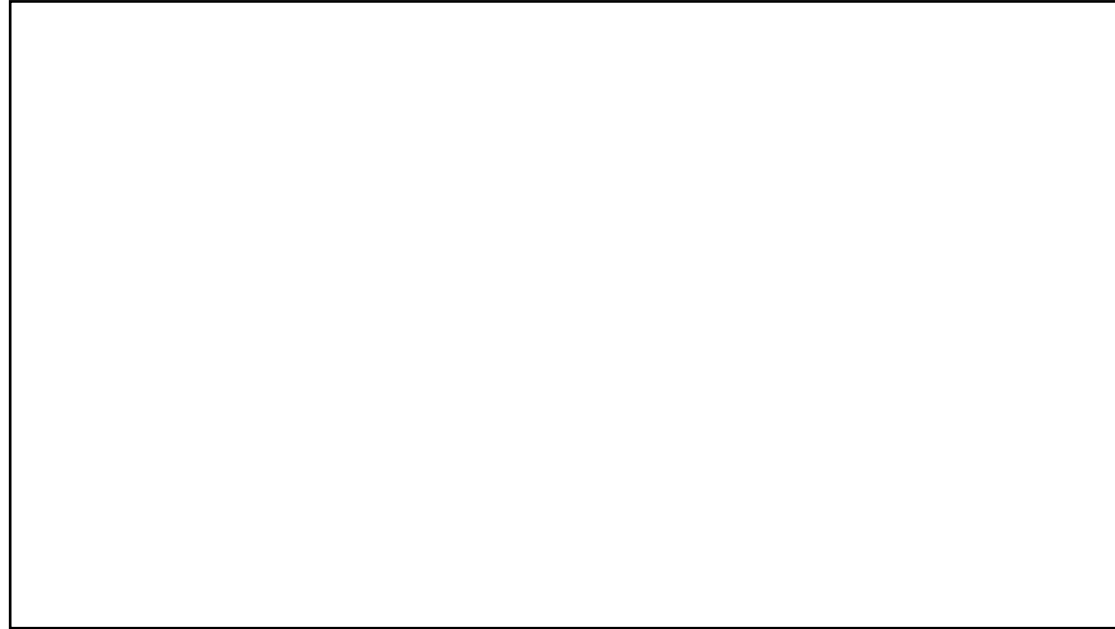
# Map Content Management



2011 @ CartoDB



# Interactive models

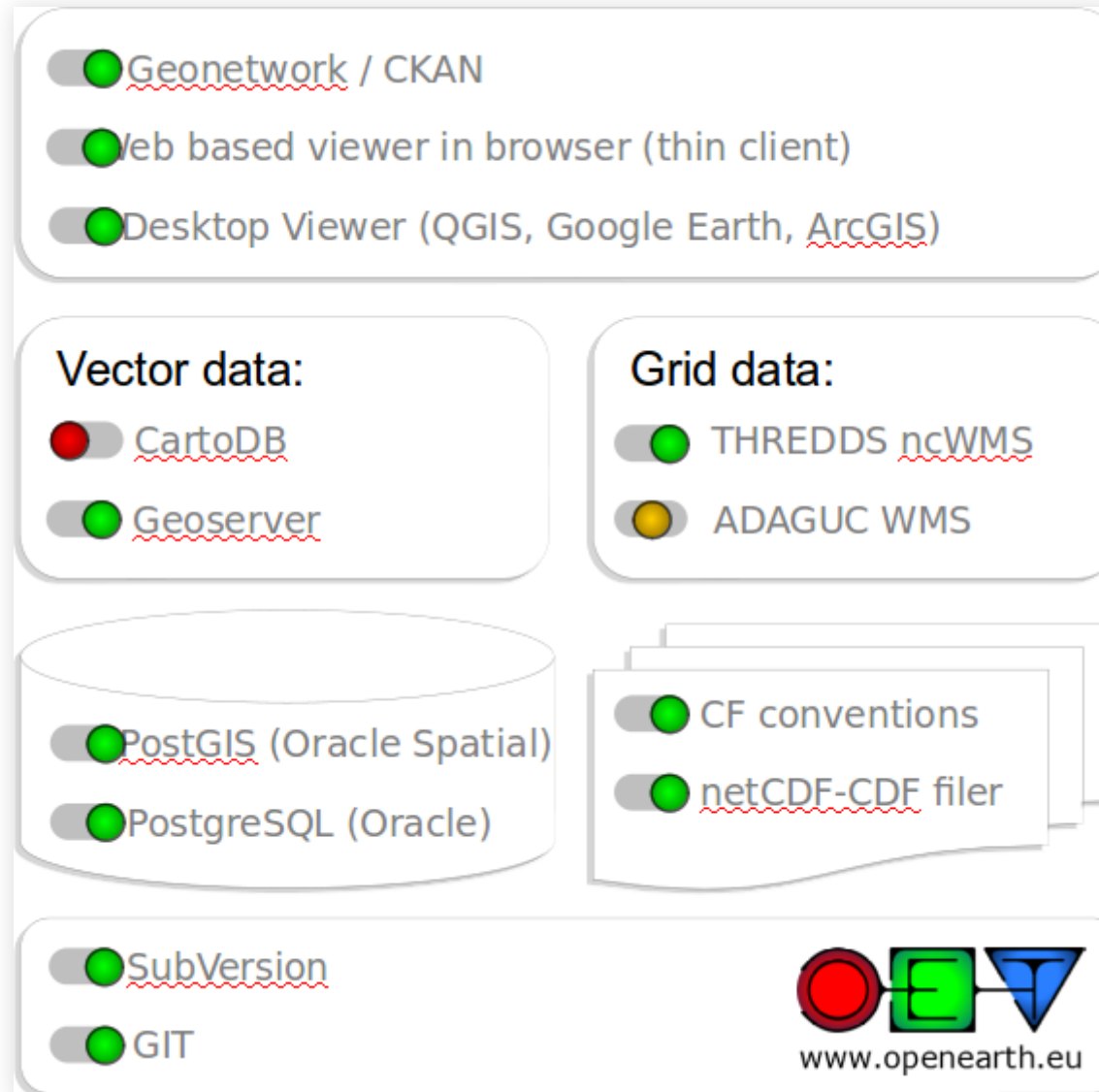


2014 @ 3Di





# OpenEarth stack





# Future

- Sensors
- Push/Messages
- Global processing
- Model setup



# OGC Service



# Web Map Services

- Get map in bitmap form
- Get information for 1 location
- Custom bounding box
- Map rendered by server on request
- Query by time and elevation
- Coordinate transformation





# GetCapabilities

```
<wms_capabilities>
  <service>
    <title>OpenEarth test server</title>
    <maxwidth>2048</maxwidth>
  </service>
  <capability>
    <request>
      <getmap>
        <format>image/png</format>
        <dcptype><http><get><onlineresource xlink:type="simple" xlink:
      </getmap>
    </request>
    <layer>
      <crs>EPSG:4326</crs>
      <title>daily averaged meteo parameter.</title>
    </layer>
  </capability>
```



# Implementations

- Geoserver
- Mapserver
- ncWMS
- ADAGUC



# GetMap

[http://geoport.whoi.edu/thredds/wms/bathy/srtm30plus\\_v6?](http://geoport.whoi.edu/thredds/wms/bathy/srtm30plus_v6?)

service=WMS&

request=GetMap&

layers=topo&

version=1.3.0&

CRS=epsg:4326&

bbox=50,0,55,10&

width=256&height=256&

styles=&

format=image/png



# Example

[http://geoport.whoi.edu/thredds/wms/bathy/srtm30plus\\_v6](http://geoport.whoi.edu/thredds/wms/bathy/srtm30plus_v6)



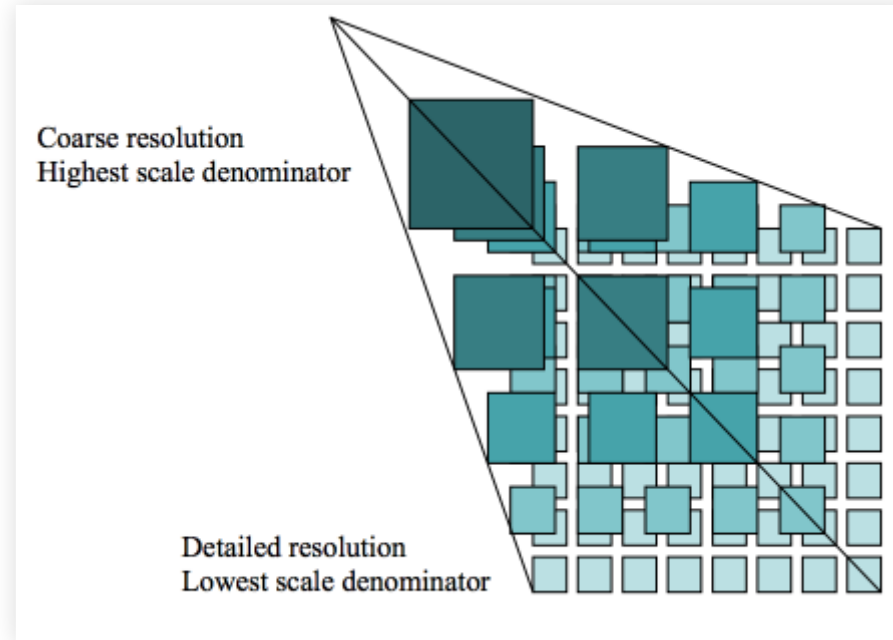


# Extensions

- COLORSCALERANGE
- LOGSCALE



# WMTS





# UTFGrid





# Challenges

Video

Streaming

Voxels

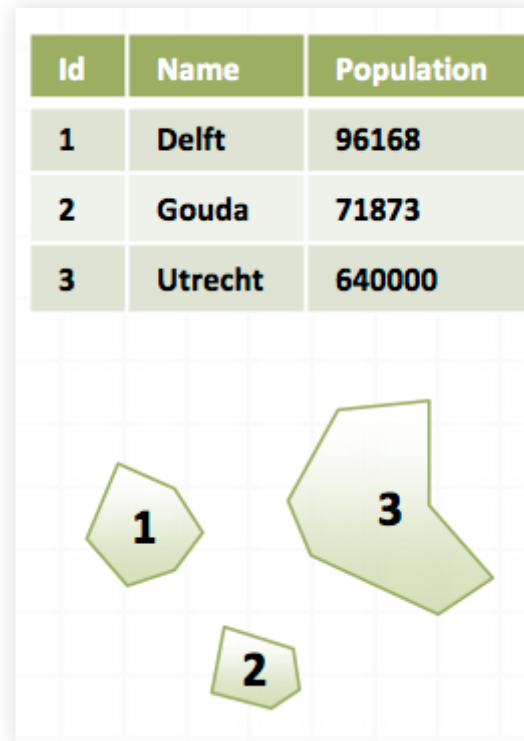




# Vector



# Everything is a feature





*The question "What is a feature?" leads directly to a philosophical rabbit hole which deposits the unwary questioner in a wonderland from which it is difficult to return.*

Nordgren, 2006



# Web Feature Service

- Get list of features
- Get table with feature info
- Custom bounding box
- Not rendered
- Coordinate transformation





# GetFeature

```
http://geo.vliz.be/geoserver/wfs?  
request=GetFeature&  
service=wfs&  
typeName=World:worldcities&  
bbox=51,2,55,5&
```



# Coverages

*Feature that acts as a function to return values from its range for any direct position within its spatiotemporal domain*

a grid



# Web Coverage Service

- Get a grid file
- Custom bounding box
- Not rendered
- Coordinate transformation



# GetCoverage

[http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus\\_v6?](http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6?)

request=GetCoverage

&version=1.0.0

&service=WCS

&format=geotiff

&coverage=topo

&BBOX=0,50,10,55





# OPeNDAP vs WCS

array or spatial?

[10.1111/j.1467-9671.2012.01312.x](https://doi.org/10.1111/j.1467-9671.2012.01312.x)



# Other services

- Sensor Observation Service
- Open Model Interface
- Catalogue Service for the Web
- OPeNDAP



# Client libraries

- OGR/GDAL (C,python,R)
- OWSLib (python)
- geotools (java)



# Client software

- QGis (desktop)
- uDig (desktop)
- Leaflet (web)
- OpenLayers (web)





# Web Processing Services



# Processes

- Call a function
- Define input variables/files
- Define output variables/files
- Custom bounding box
- Can be rendered or not
- Start process
- Chain processes



# Typical examples

- Spatial operations
- Conversions
- Process @ data
- Simple models
- Facade to complex models



# Operations

- GetCapabilities
- DescribeProcess
- Execute

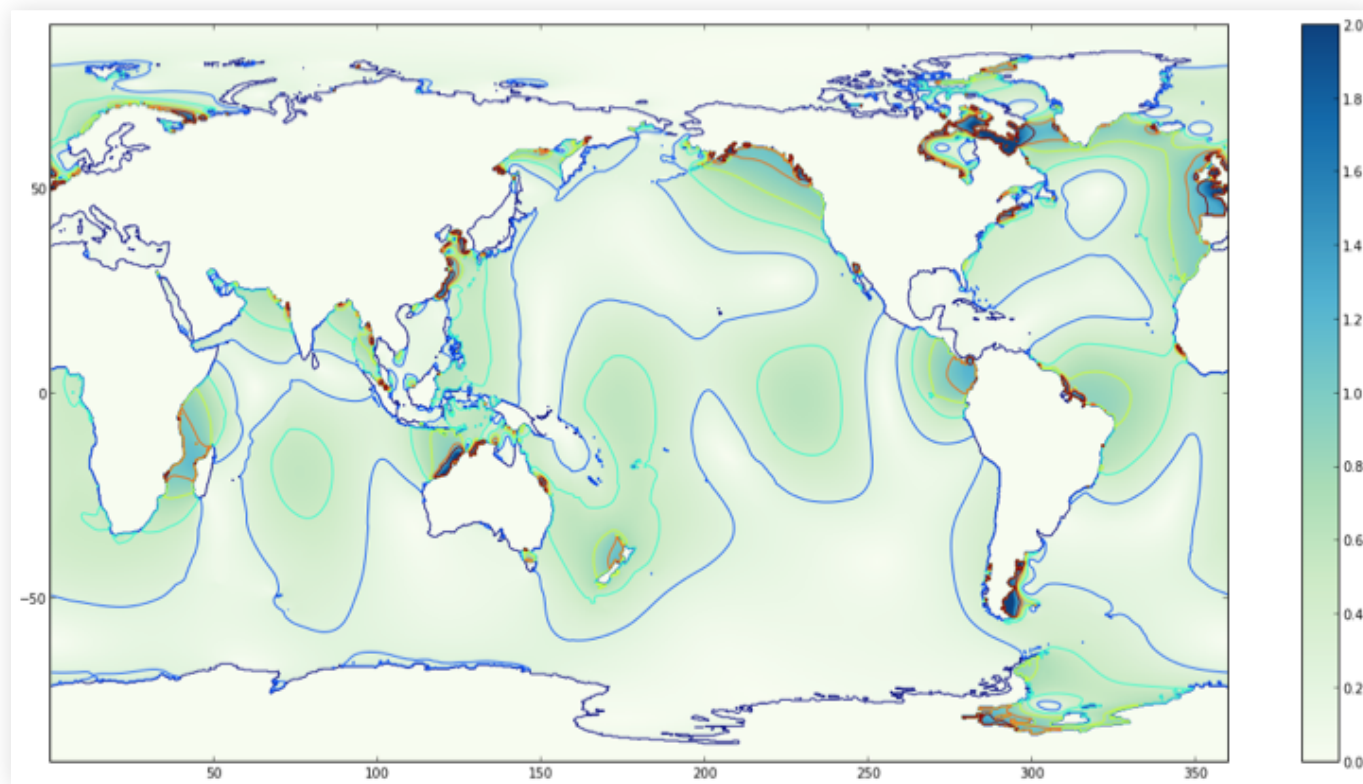




# Tide Example

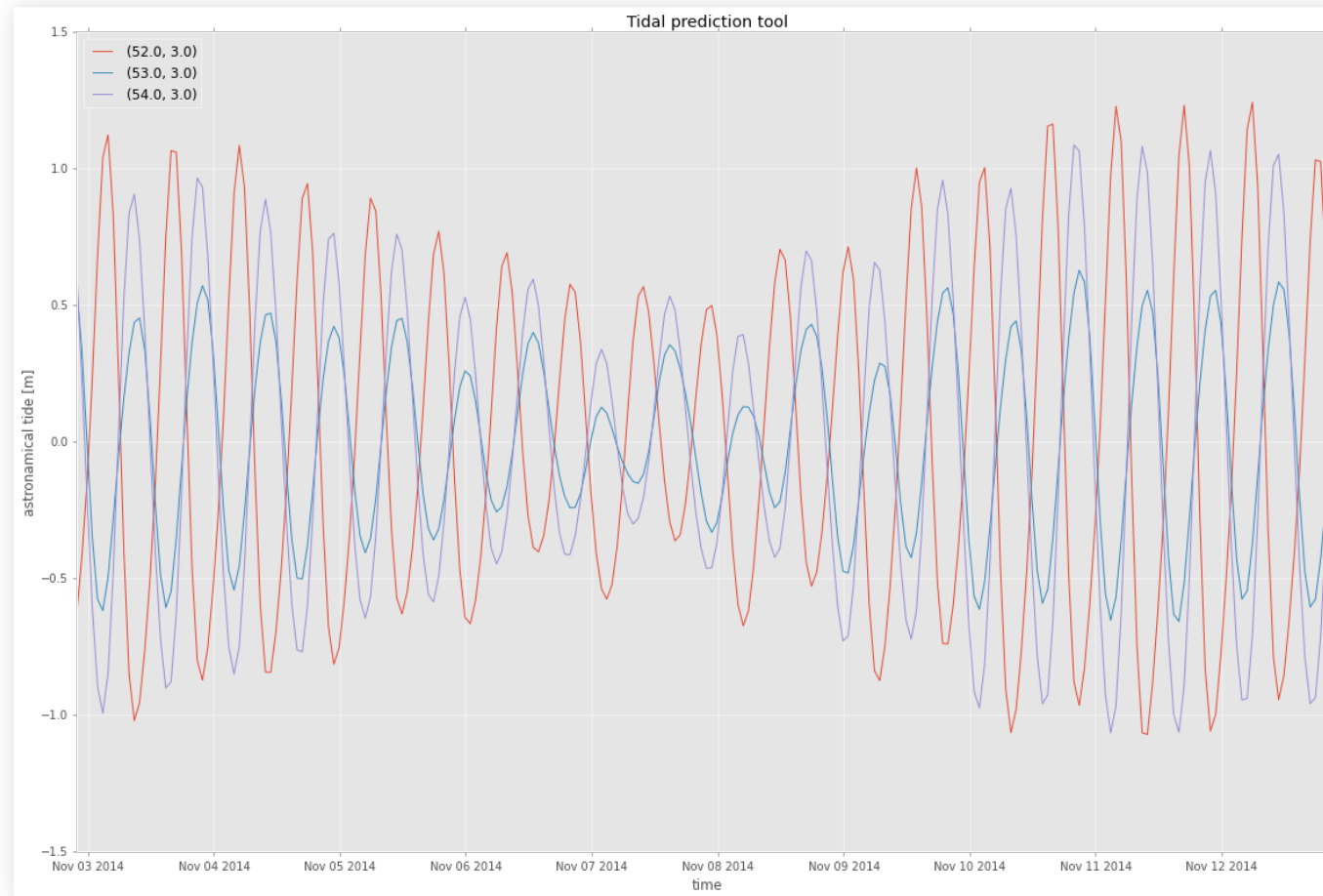


# OSU Tidal constituents





# Tidal predict





# GetCapabilities

```
<wps:capabilities service="WPS" version="1.0.0">
  <ows:serviceidentification>
    <ows:title>OpenEarth WPS server</ows:title>
    <ows:accessconstraints>none</ows:accessconstraints>
  </ows:serviceidentification>
  <ows:serviceprovider>
    <ows:providername>Deltares</ows:providername>
  </ows:serviceprovider>
  <wps:processofferings>
    <wps:process wps:processversion="0.1">
      <ows:identifier>tidal_predict</ows:identifier>
      <ows:title>Tidal prediction tool</ows:title>
    </wps:process>
  </wps:processofferings>
</wps:capabilities>
```





# DescribeProcess

```
<wps:processdescriptions service="WPS" version="1.0.0" xml:lang="en-CA">
  <processdescription>
    <ows:identifier>tidal_predict</ows:identifier>
    <ows:title>Tidal prediction tool</ows:title>
    <datainputs>
      ...<identifier>startdate</identifier>...
    </datainputs>
    <processoutputs>
      <output>
        <ows:identifier>tide</ows:identifier>
        <ows:title>Calculated water level for requested locations and date</ows:title>
        <complexoutput>
          ...<mimetype>text/csv</mimetype>...
        </complexoutput>
      </output>
    </processoutputs>
  </processdescription>
</wps:processdescriptions>
```



# Input/Output

- Title/Abstract
- Description
- Unit
- Default
- Min/max occurrence

## Types

- Literal (string, number)
- Bounding Box
- Complex (raster, vector)



# Execute

```
<wps:executerresponse>
<wps:process wps:processversion="0.1">
  <ows:identifier>tidal_predict</ows:identifier>
  <wps:status creationtime="2014-11-02T19:50:39Z">
    <wps:processsucceeded>PyWPS Process tidal_predict successfully calculated<
  </wps:status>
  <wps:processoutputs>
    <wps:output>
      <ows:title>Calculated water level for requested locations and date</ows:
      <wps:data>
        <wps:complexdata mimetype="text/csv">
date,h,lat,lon
2014-11-02 19:50:37,0.3735264572229592,3.0,52.0
        </wps:complexdata>
      </wps:data>
    </wps:output>
  </wps:processoutputs>
```



# Server Implementations

- Zoo
- PyWPS
- Geoserver
- 52North





# Tools and languages

- JTS
- GRASS
- Orfeo
- Sextante
- python, R, java



# Implementing a process

```
class Process(WPSProcess):
    def __init__(self):
        WPSProcess.__init__(self, "tidal_predict",
                             title="Tidal prediction tool")
    def execute(self):
        """execute a tidal prediction"""
        date = self.date.getValue()
        location = self.location.getValue()
        df = openearthtools.physics.tide.predict(location, date=date)
        self.tide.setValue(df.to_json())
```



# Client Implementations

- OpenLayers
- OWSLib
- Geotools (unsupported)



# Client Example

```
// OpenLayers example
client = new OpenLayers.WPSCient({
  servers: {
    opengeo: 'http://demo.opengeo.org/geoserver/wps '
  }
});

intersect = client.getProcess('opengeo', 'JTS:intersection');
intersect.configure({
  inputs: {
    a: features,
    b: geometry
  },
  success: function(outputs) {
    map.baseLayer.addFeatures(outputs.result);
  }
});
```





# Setting up a server

1. Get OSX/linux computer with VirtualBox
2. Install ansible
3. Install vagrant
4. Download OpenEarth stack @ github
5. `$ vagrant up wps`



# Missing features

- Stop a process
- REST
- Spatial types (everything is complex)
- Layout hints/classes
- Variable relations
- Push/notifications



# Planned in WPS 2

Fall of 2014?

- Pause, Resume, Dismiss
- REST?



# Excercises





# WMS + WFS

wfs/wms client

# WPS

wps client





