Geospatial Data Processing with Stetl

www.stetl.org

Just van den Broecke

GeoPython 2016

Muttenz - Switserland

June 24, 2016

www.justobjects.nl

About Me

Independent Open Source Geospatial Professional

+ Secretary OSGeo Dutch Local Chapter
+ Member of the Dutch OpenGeoGroep

Just van den Broecke
just@justobjects.nl
www.justobjects.nl

OPEN GEO GROEP

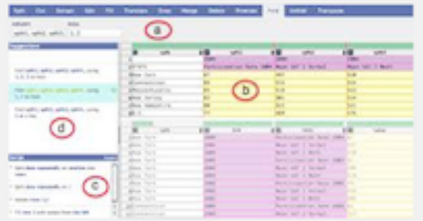
OSGeo.nl
Nederlandstalige afdeling

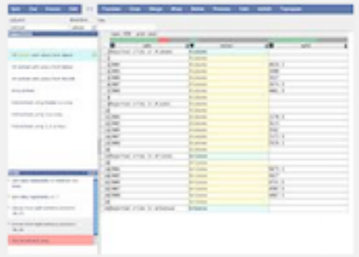
Agenda

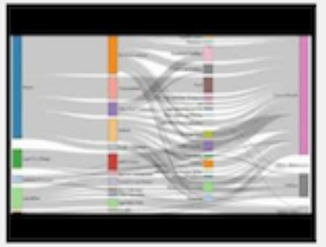
- Spatial ETL
- Stetl
  - Concepts
  - Cases
  - Status
- Q & A

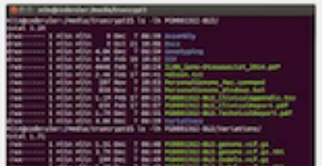
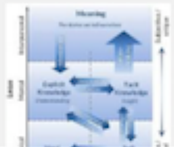
Spatial ETL

Data Wrangling

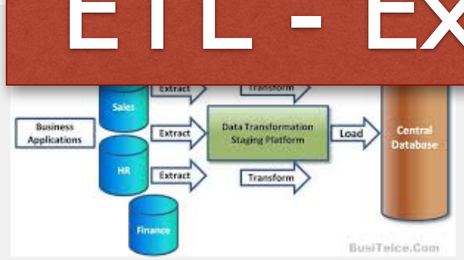


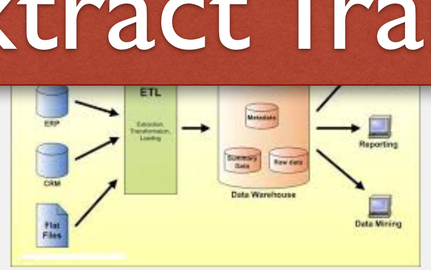


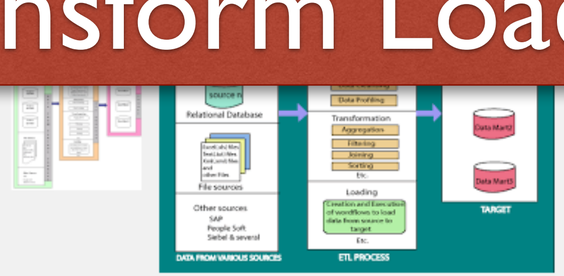


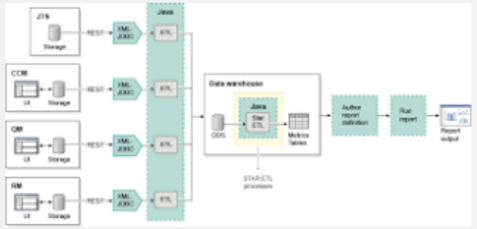



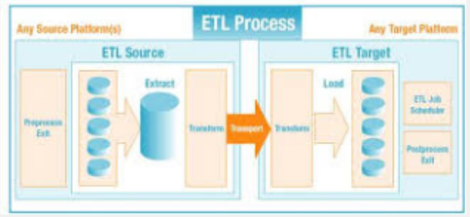
ETL - Extract Transform Load

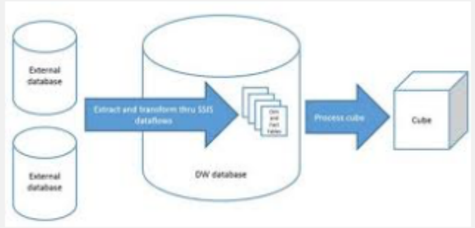


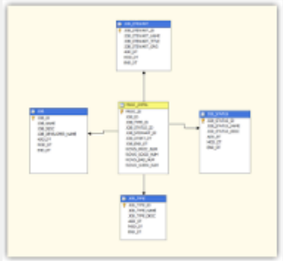


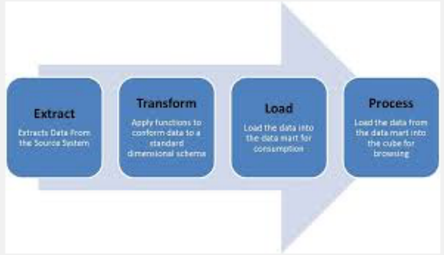


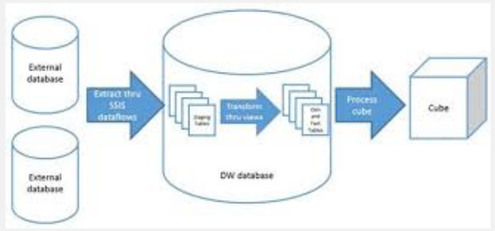


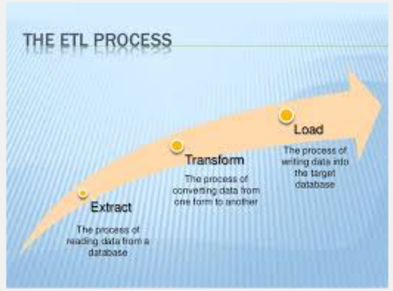


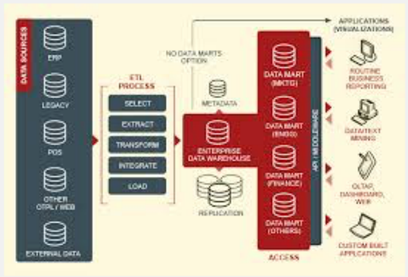


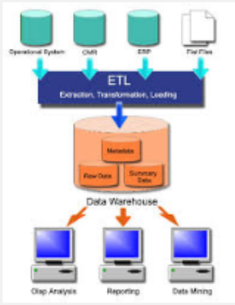


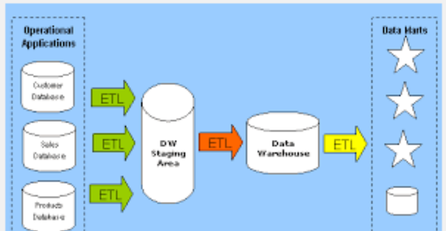


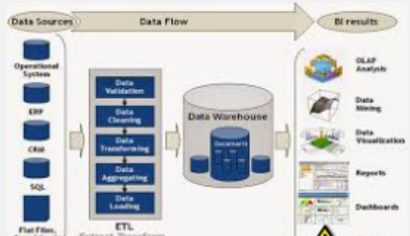




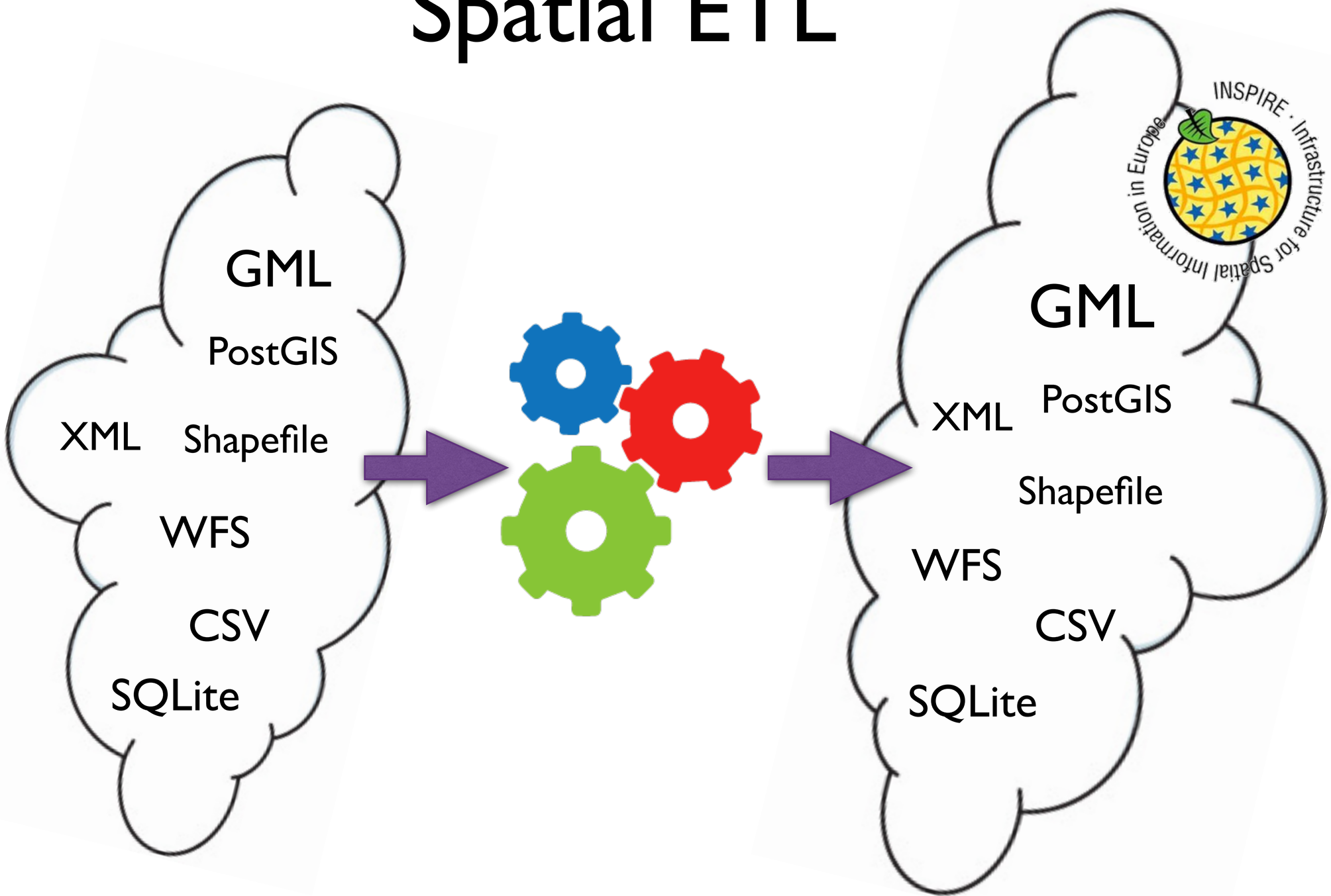








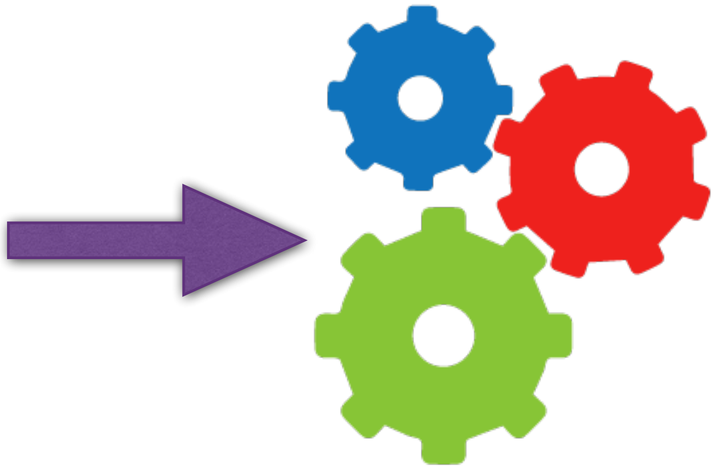
Spatial ETL

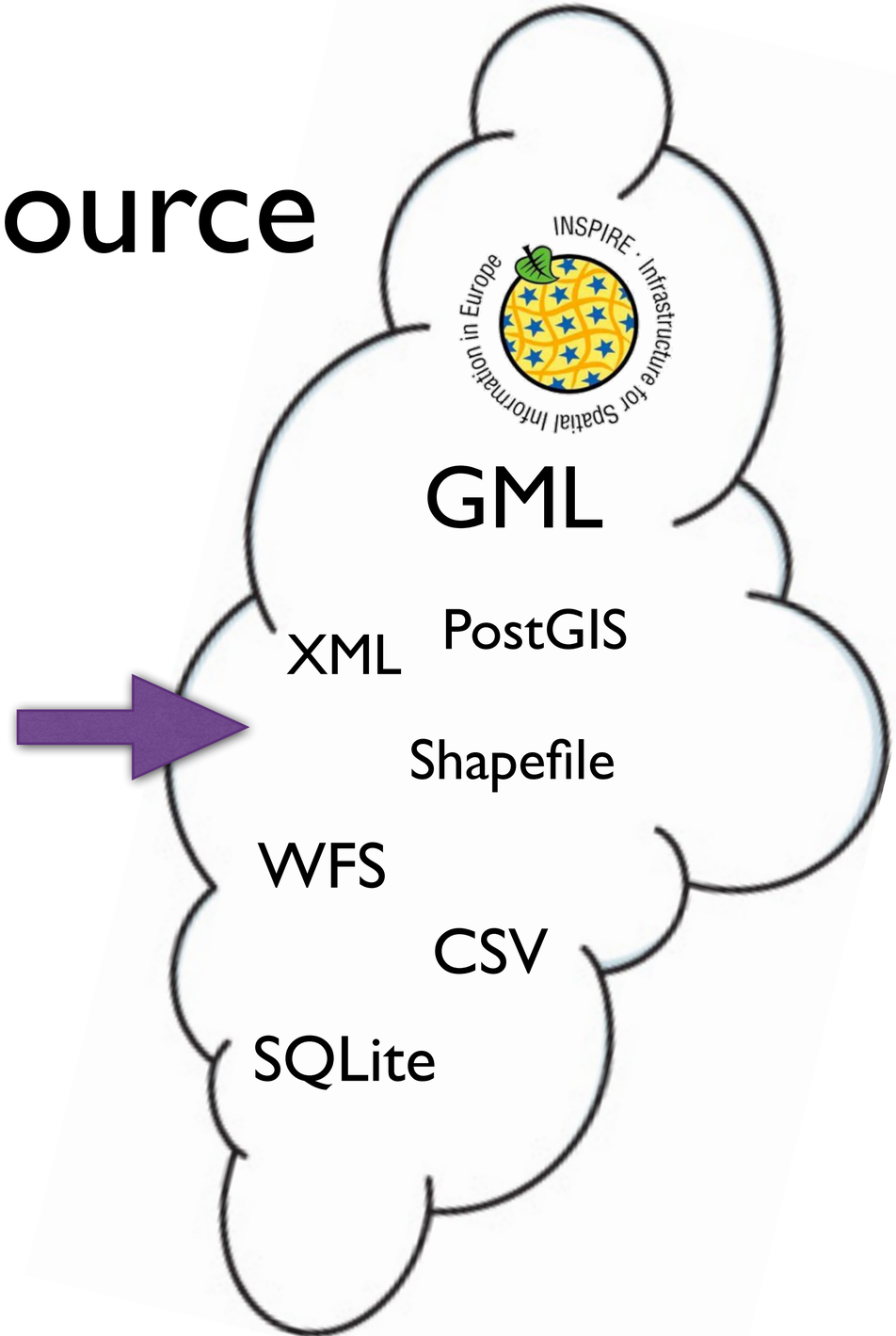

GML
PostGIS
XML
Shapefile
WFS
CSV
SQLite
INSPIRE - Infrastructure for Spatial Information in Europe
GML
XML
PostGIS
Shapefile
WFS
CSV
SQLite

Spatial ETL

Example non-standard source

```xml
<?xml version='1.0' encoding='utf-8'?>
<cities>
    <city>
        <name>Amsterdam</name>
        <lat>52.4</lat>
        <lon>4.9</lon>
    </city>
    <city>
        <name>Bonn</name>
        <lat>50.7</lat>
        <lon>7.1</lon>
    </city>
    <city>
        <name>Rome</name>
        <lat>41.9</lat>
        <lon>12.5</lon>
    </city>
</cities>
```





GDAL/OGR and PROJ.4

http://www.gdal.org

http://proj.osgeo.org

GDAL for Raster Data - The Geospatial Data Abstraction Library

Arc/Info ASCII Grid, **Arc/Info** Binary Grid (.adf), AIRSAR Polarimetric, Microsoft Windows Device Independent Bitmap (.bmp), BSB Nautical Chart Format (.kap), VTP Binary Terrain Format (.bt), CEOS (Spot for instance), First Generation USGS DOQ (.doq), DODS / OPeNDAP, New Labelled USGS DOQ (.doq), Military Elevation Data (.dt0, .dt1), **ERMapper** Compressed Wavelets (.ecw), **ESRI** .hdr Labelled, ENVI .hdr Labelled Raster, Envisat Image Product (.n1), EOSAT FAST Format, FITS (.fits), Graphics Interchange Format (**.gif**), GMT Compatible netCDF, **GRASS Rasters**, Golden Software ASCII Grid, Golden Software Binary Grid, Golden Software **Surfer** 7 Binary Grid, TIFF / **GeoTIFF** (.tif), GXF - Grid eXchange File, Hierarchical Data Format Release 4 (**HDF4**), Hierarchical Data Format Release 5 (HDF5), **Erdas Imagine** (.img), Vexcel MFF2, **Idrisi** Raster, Image Display and Analysis (WinDisp), ILWIS Raster Map (.mpr,.mpl), Japanese DEM (.mem), **JPEG** JFIF (.jpg), **JPEG2000** (JPEG2000, JP2KAK, JP2ECW, JP2MrSID), NOAA Polar Orbiter Level 1b Data Set (**AVHRR**), Erdas 7.x .LAN and .GIS, Daylon Leveller Heightfield, In Memory Raster, Vexcel MFF, Multi-resolution Seamless Image Database, Meteosat Second Generation, NDF, NITF, **NetCDF**, OGDI Bridge, PCI .aux Labelled, PCI Geomatics Database File, Portable Network Graphics (**.png**), PCRaster (.map), Netpbm (.ppm,.pgm), Swedish Grid RIK (.rik), RadarSat2 XML (product.xml), **ArcSDE** Raster, USGS SDTS DEM (*CATD.DDF), Raster Matrix Format (*.rsw, .mtw), SAR CEOS, **SGI** Image Format, USGS ASCII DEM (.dem), OGC Web Coverage Server, X11 Pixmap (.xpm)

OGR for Vector Data - Simple Feature Library

Arc/Info Binary Coverage, Comma Separated Value (**.csv**), DODS/OPeNDAP, DWG, **DXF**, ESRI Personal GeoDatabase, ESRI ArcSDE, **ESRI Shapefile**, FMEObjects Gateway, GML, **GMT Mapping**, **GRASS Vectors**, INTERLIS, **Google Earth KML**, Mapinfo File, Microstation DGN, Spatial **MySQL**, OGDI Vectors, ODBC generic database access layer, **Oracle** Spatial, PostgreSQL **PostGIS**, S-57 (ENC), SDTS, SQLite, UK .NTF, U.S. Census TIGER/Line, VRT - Virtual Datasource, Informix DataBlade

From: https://live.osgeo.org/en/overview/gdal_overview.html

Kersten
@Fernerkundung

Follow

When people describe #GDAL als the "swiss army knife of GIS data" this is what I think of.

RETWEETS 20 LIKES 21

12:49 PM - 25 Jan 2016

20 21

Plenty of Tools…

ogr2ogr

Each tool is powerful by itself but cannot do the entire ETL

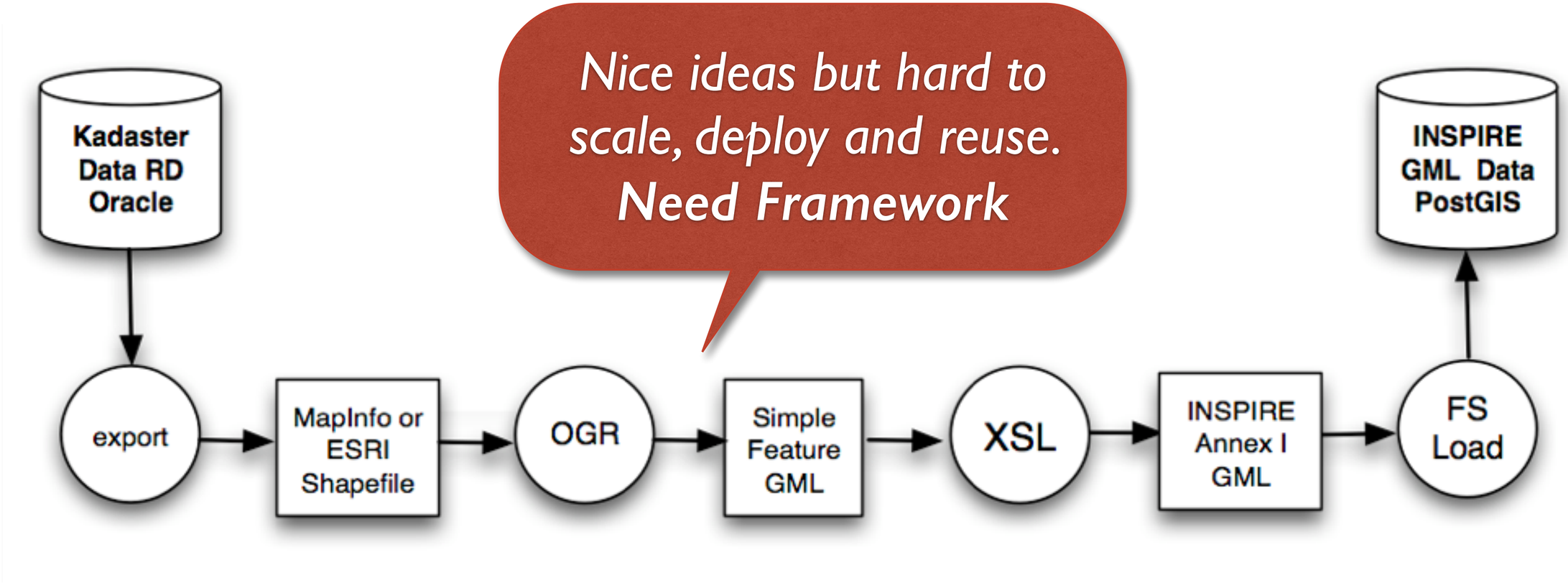
FOSS ETL - How to Combine Components?

+

+

+ .. = ?

Example - 2011 INSPIRE-FOSS



http://inspire.kademo.nl/doc/design-etl.html

Solution: Add Python to the Equation

(

+

+

+ ..) = ?

Solution: Add Python to the Equation

(+ + + ..) =

Stetl

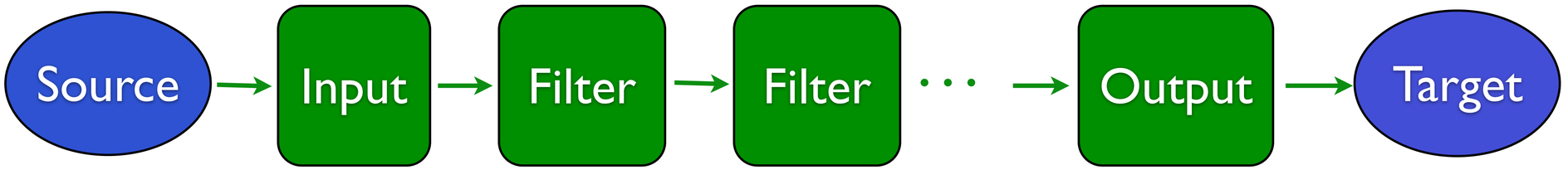
Stetl

=

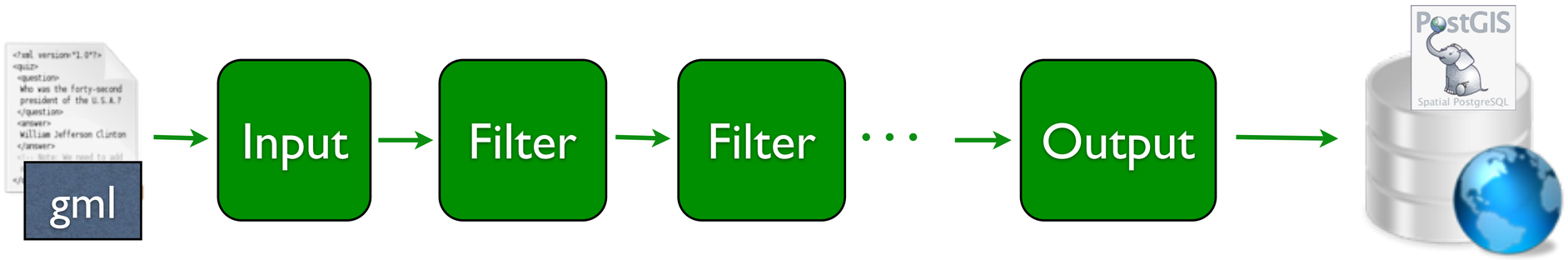
Simple
Streaming
Spatial
Speedy
ETL

Stetl
Concepts

Process Chain

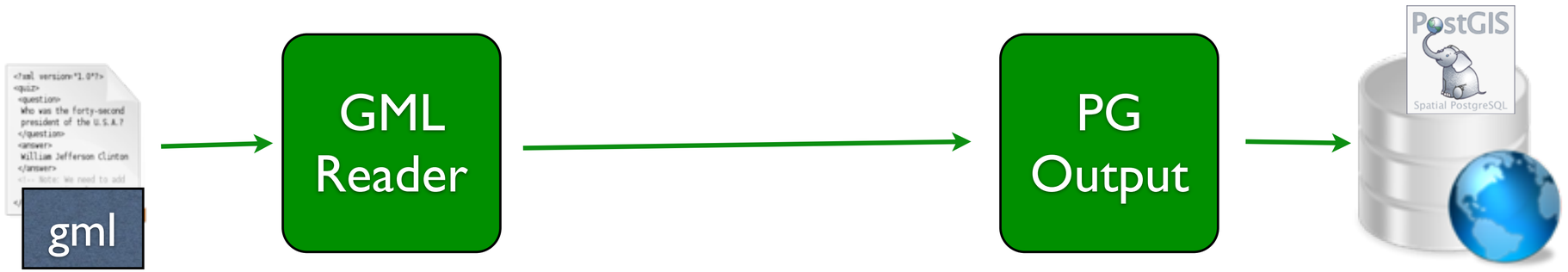


Process Chain



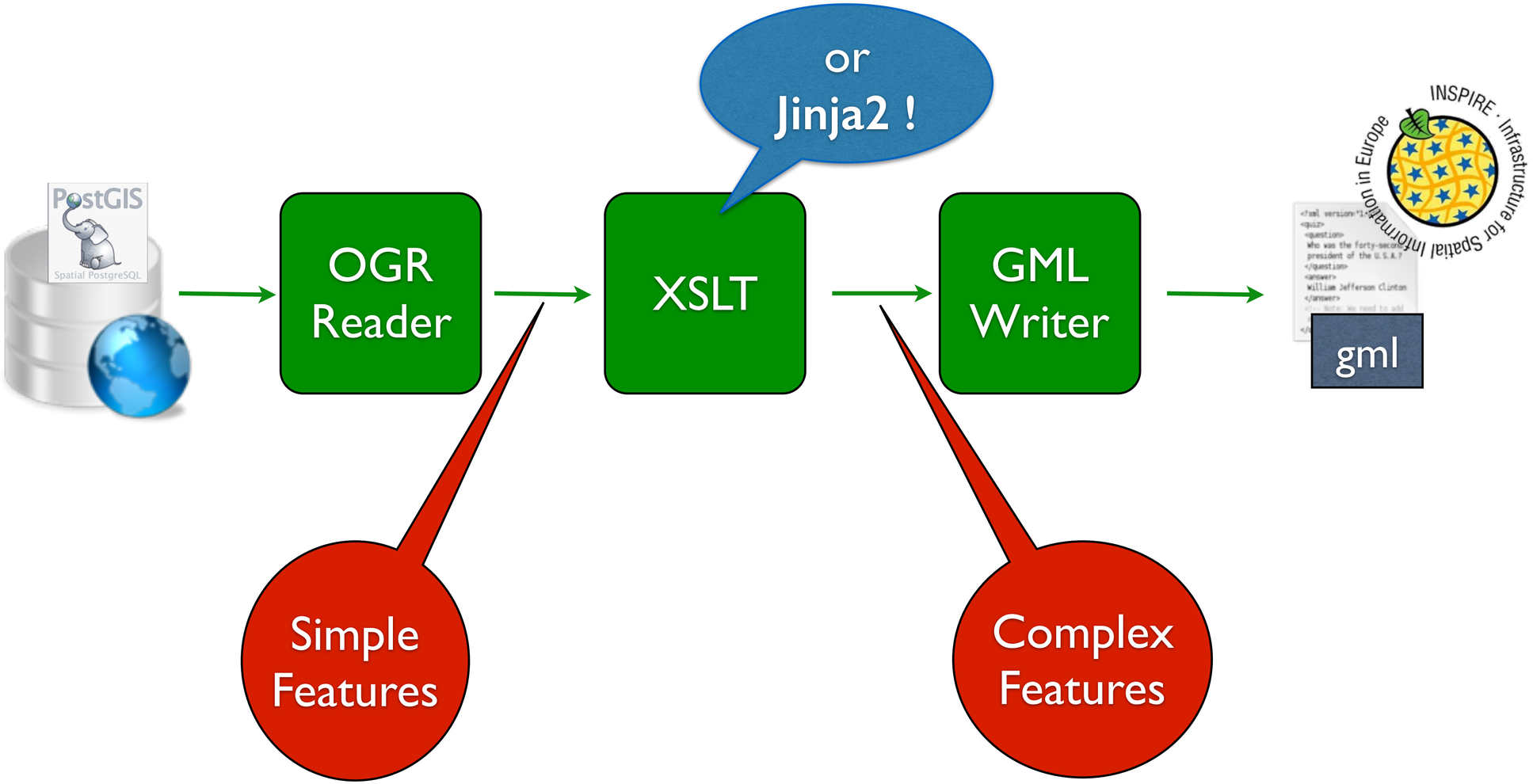
Stetl concepts

Example: GML to PostGIS

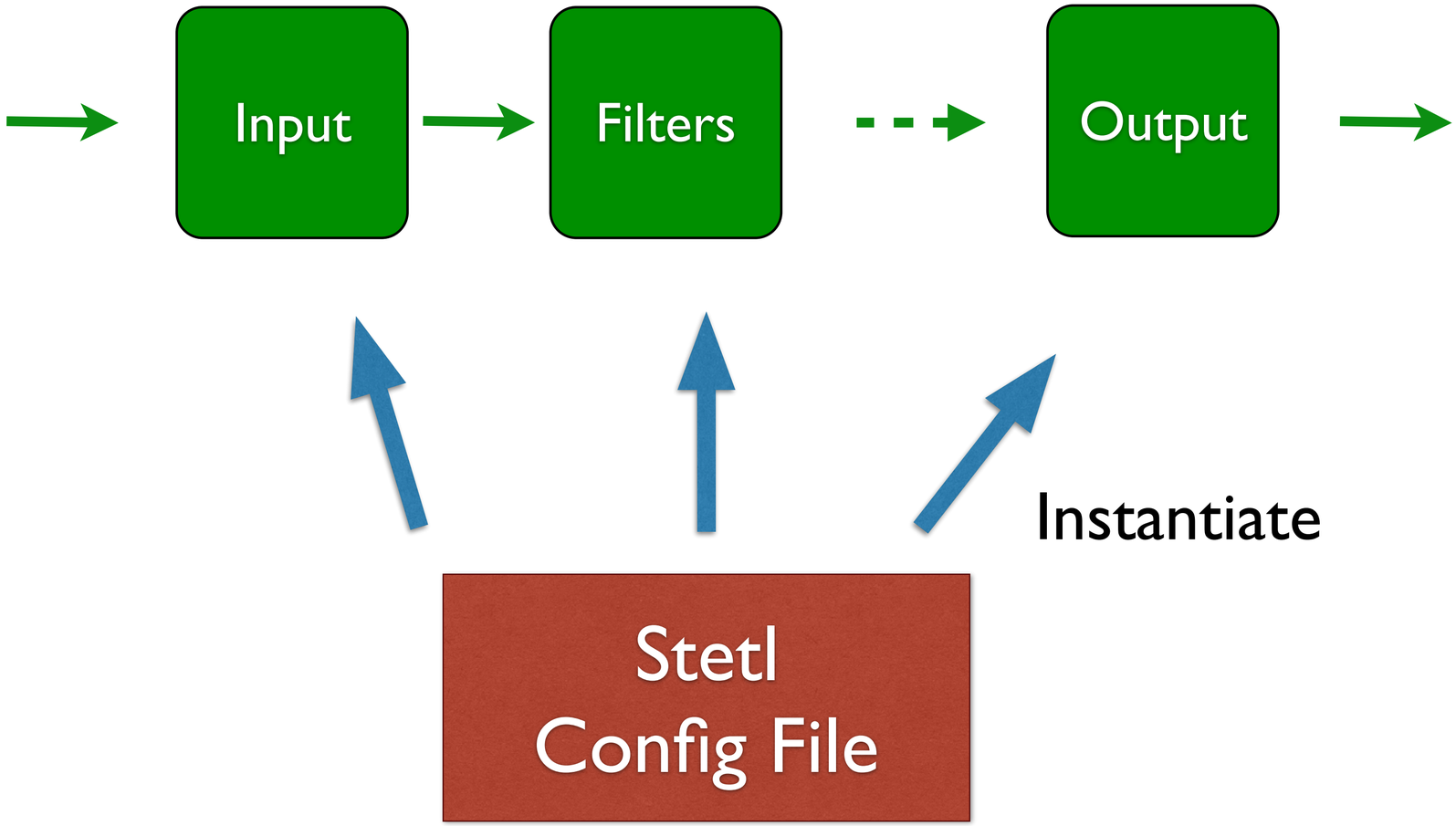


Stetl concepts

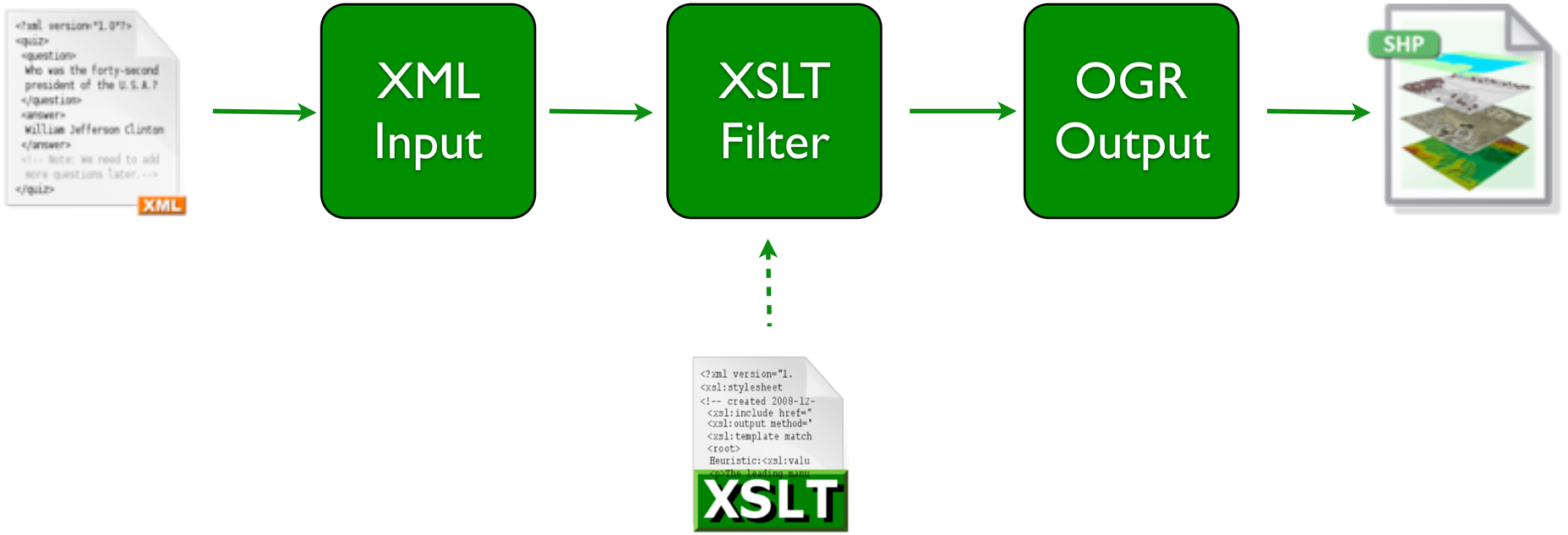
Example: Data Model Transform



Process Chain - How?



Example: XML to Shape

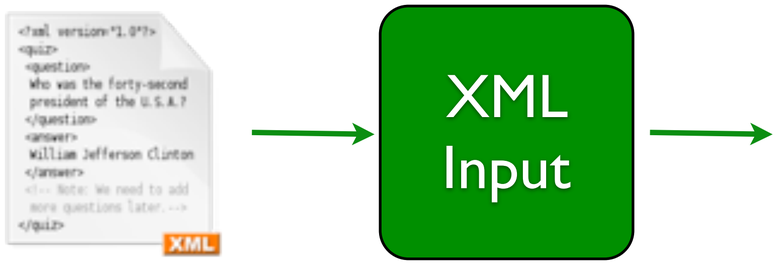


Example: XML to Shape

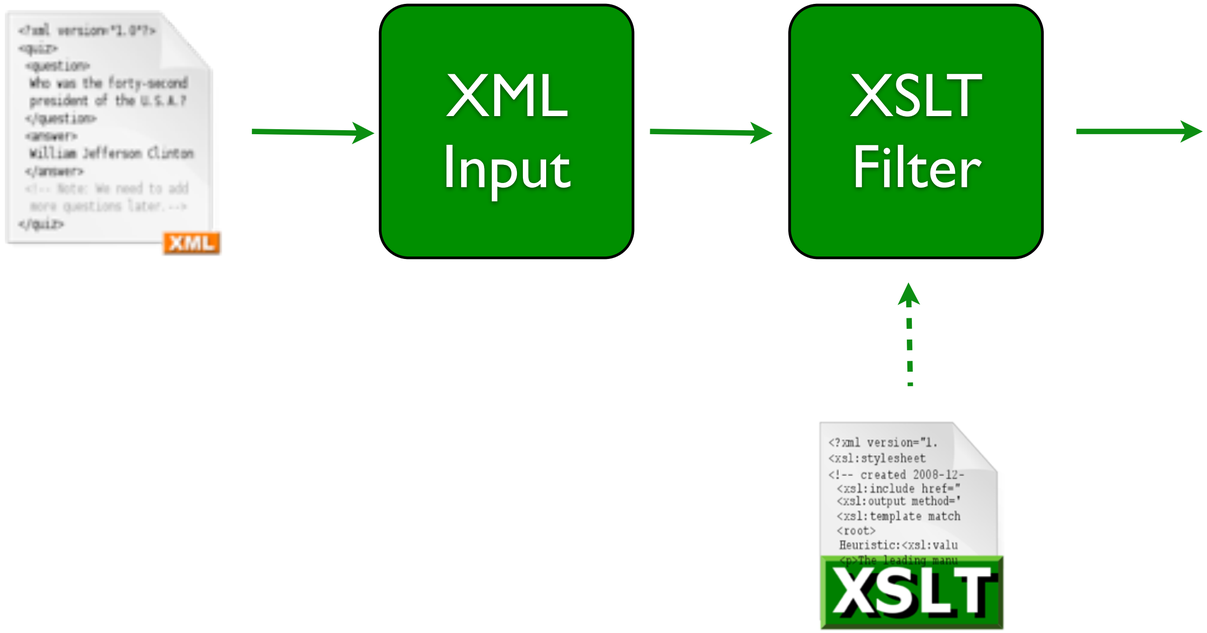
```xml
<?xml version='1.0' encoding='utf-8'?>
<cities>
    <city>
        <name>Amsterdam</name>
        <lat>52.4</lat>
        <lon>4.9</lon>
    </city>
    <city>
        <name>Bonn</name>
        <lat>50.7</lat>
        <lon>7.1</lon>
    </city>
    <city>
        <name>Rome</name>
        <lat>41.9</lat>
        <lon>12.5</lon>
    </city>
</cities>
```

The Source File

Example: XML to Shape



Example: XML to Shape



Example: XML to Shape

```
    <xsl:template match="/">
        <ogr:FeatureCollection
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:ogr="http://ogr.maptools.org/"
                xmlns:gml="http://www.opengis.net/gml"
                xsi:schemaLocation="http://ogr.maptools.org/ ../gmlcities.xsd  http://www.opengis.net/gml http://schemas
                >
            <gml:boundedBy>
              <gml:Box>
                <gml:coord><gml:X>-180.0</gml:X><gml:Y>-90.0</gml:Y></gml:coord>
                <gml:coord><gml:X>180.0</gml:X><gml:Y>90.0</gml:Y></gml:coord>
              </gml:Box>
            </gml:boundedBy>
             <!-- Loop through all cities. -->
            <xsl:apply-templates/>
        </ogr:FeatureCollection>
    </xsl:template>

    <!-- Make each city an ogr:featureMember. -->
    <xsl:template match="city">
        <gml:featureMember>
            <ogr:City>
                <ogr:name>
                    <xsl:value-of select="name"/>
                </ogr:name>
                <ogr:geometry>
                    <gml:Point srsName="urn:ogc:def:crs:EPSG:4326">
                        <gml:coordinates><xsl:value-of select="lat"/>,<xsl:value-of select="lon"/></gml:coordinates>
                     </gml:Point>
                </ogr:geometry>
            </ogr:City>
        </gml:featureMember>
    </xsl:template>
</xsl:stylesheet>
```

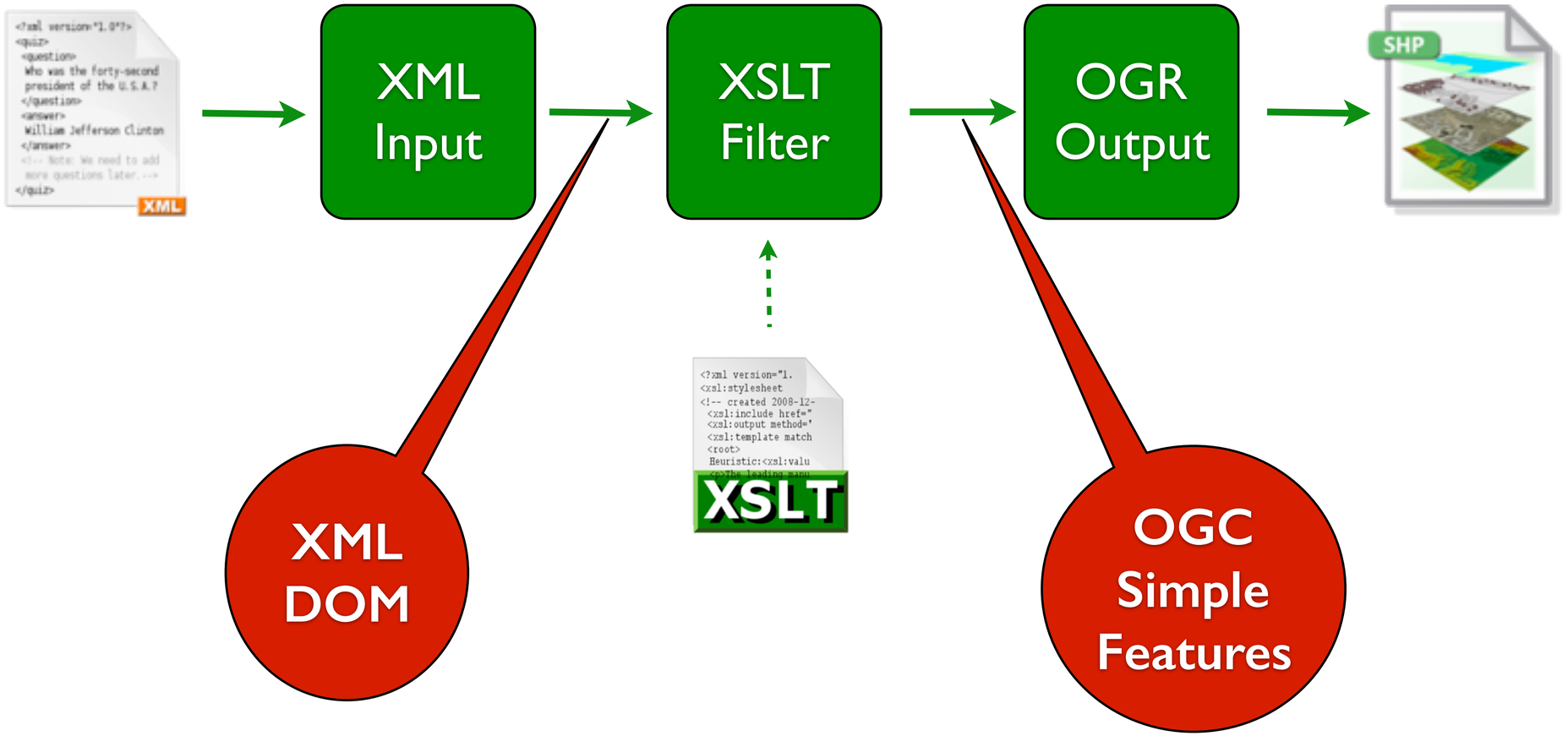
Prepare XSLT Script

Example: XML to Shape

```
<?xml version='1.0' encoding='utf-8'?>
<ogr:FeatureCollection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ogr="http://
  <gml:boundedBy>
    <gml:Box>
      <gml:coord>
        <gml:X>-180.0</gml:X>
        <gml:Y>-90.0</gml:Y>
      </gml:coord>
      <gml:coord>
        <gml:X>180.0</gml:X>
        <gml:Y>90.0</gml:Y>
      </gml:coord>
    </gml:Box>
  </gml:boundedBy>
  <gml:featureMember>
    <ogr:City>
      <ogr:name>Amsterdam</ogr:name>
      <ogr:geometry>
        <gml:Point srsName="urn:ogc:def:crs:EPSG:4326">
          <gml:coordinates>52.4,4.9</gml:coordinates>
        </gml:Point>
      </ogr:geometry>
    </ogr:City>
  </gml:featureMember>
  <gml:featureMember>
    <ogr:City>
      <ogr:name>Bonn</ogr:name>
      <ogr:geometry>
        <gml:Point srsName="urn:ogc:def:crs:EPSG:4326">
          <gml:coordinates>50.7,7.1</gml:coordinates>
        </gml:Point>
      </ogr:geometry>
    </ogr:City>
  </gml:featureMember>
```

XSLT GML Output

Example: XML to Shape



Example: XML to Shape

```
# Transform input xml to valid GML file and then to Shape.

[etl]
chains = input_xml_file | transformer_xslt | output_ogr_shape

[input_xml_file]
class = inputs.fileinput.XmlFileInput
file_path = input/cities.xml

[transformer_xslt]
class = filters.xsltfilter.XsltFilter
script = cities2gml.xsl

# The ogr2ogr command-line.
[output_ogr_shape]
class = outputs.ogroutput.Ogr2OgrOutput
temp_file = temp/gmlcities.gml
ogr2ogr_cmd = ogr2ogr \
    -overwrite \
    -f "ESRI Shapefile" \
    -a_srs epsg:4326 \
   output/gmlcities.shp \
   temp/gmlcities.gml
```

Process Chain

XML Input

XSLT Filter

ogr2ogr Output

The Stetl Config File

Running Stetl

```
stetl -c etl.cfg
```

```
stetl -c etl.cfg [-a <properties>]
```

Installing Stetl - PyPi

```
sudo pip install stetl
```

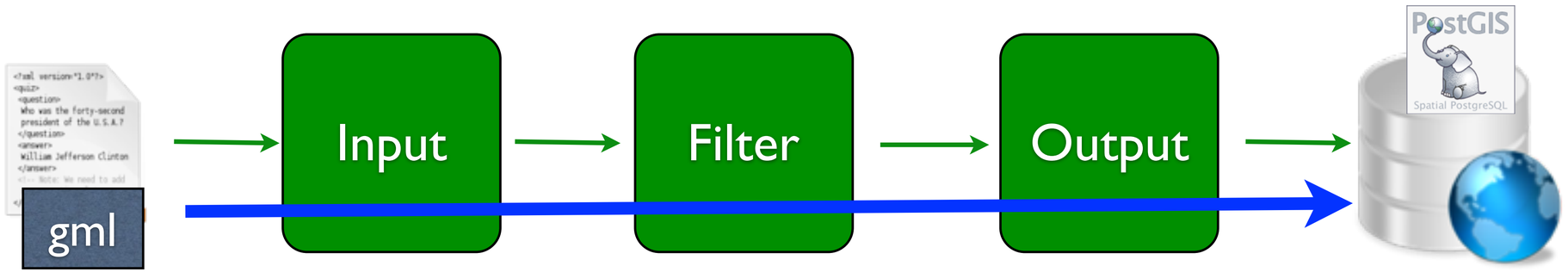
Deps

- GDAL+Python bindings
- lxml (xml proc)
- psycopg2 (Postgres)

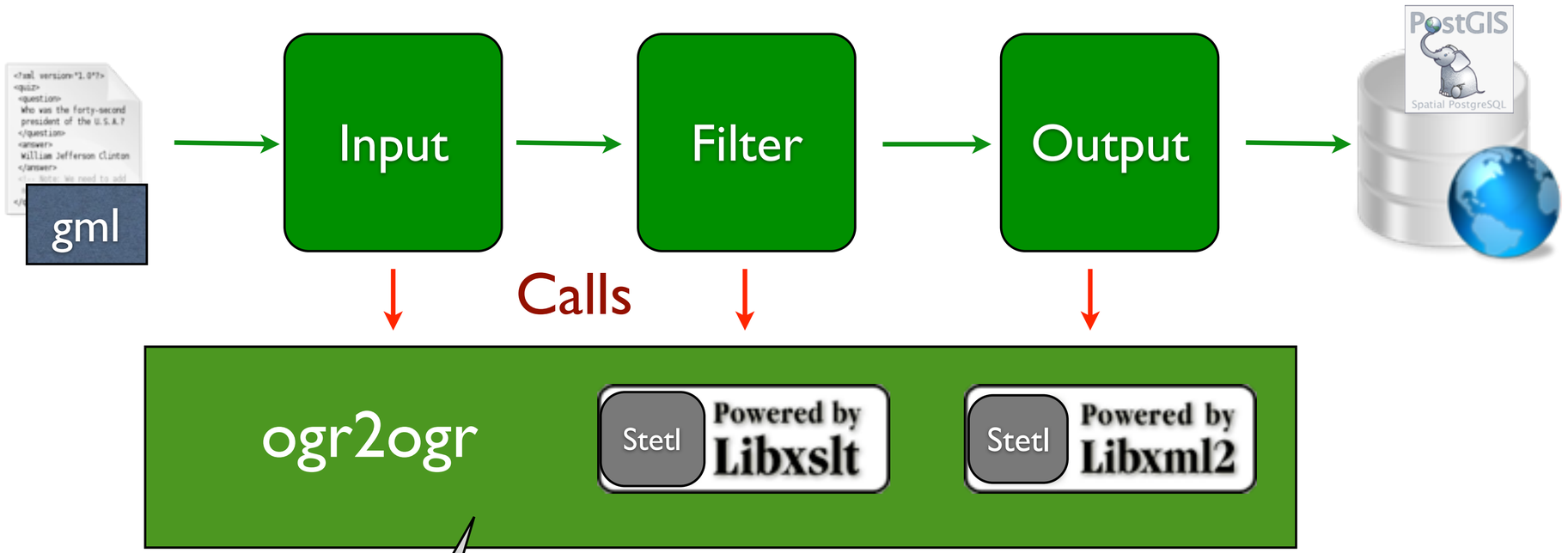
Installing Stetl - new: Docker

https://hub.docker.com/r/justb4/stetl

Speed: Streaming



Speed: Going Native



Native C Libs/Progs

Example Components

XMLFile	XSLT	GML
ogr2ogr	XMLAssembler	GDAL/OGR
LineStream	XMLValidator	WFS-T
Postgres/PostGIS	Jinja2	Postgres/PostGIS
deegree*	FeatureExtractor	deegree*
YourInput	*YourFilter*	*YourOutput*

Example: XsltFilter Python

```
from util import Util, etree
from filter import Filter
from packet import FORMAT

log = Util.get_log("xsltfilter")


class XsltFilter(Filter):
    # Constructor
    def __init__(self, configdict, section):
        Filter.__init__(self, configdict, section, consumes=FORMAT.etree_doc,
produces=FORMAT.etree_doc)

        self.xslt_file_path = self.cfg.get('script')
        self.xslt_file = open(self.xslt_file_path, 'r')
        # Parse XSLT file only once
        self.xslt_doc = etree.parse(self.xslt_file)
        self.xslt_obj = etree.XSLT(self.xslt_doc)
        self.xslt_file.close()

    def invoke(self, packet):
        if packet.data is None:
            return packet
        return self.transform(packet)

    def transform(self, packet):
        packet.data = self.xslt_obj(packet.data)
        log.info("XSLT Transform OK")
        return packet
```

Your Own Components

```
class MyFilter(Filter):
    # Constructor
    def __init__(self, configdict, section):
        Filter.__init__(self, configdict, section, consumes=FORMAT.etree_doc,
                                                   produces=FORMAT.etree_doc)

    def invoke(self, packet):
        log.info("CALLING MyFilter OK!!!!")
        return packet
```

Step 1- Define Class

```
[etl]
chains = input_xml_file|my_filter|output_std

[input_xml_file]
class = inputs.fileinput.XmlFileInput
file_path = input/cities.xml

# My custom component
[my_filter]
class = my.myfilter.MyFilter

[output_std]
class = outputs.standardoutput.StandardXmlOutput
```

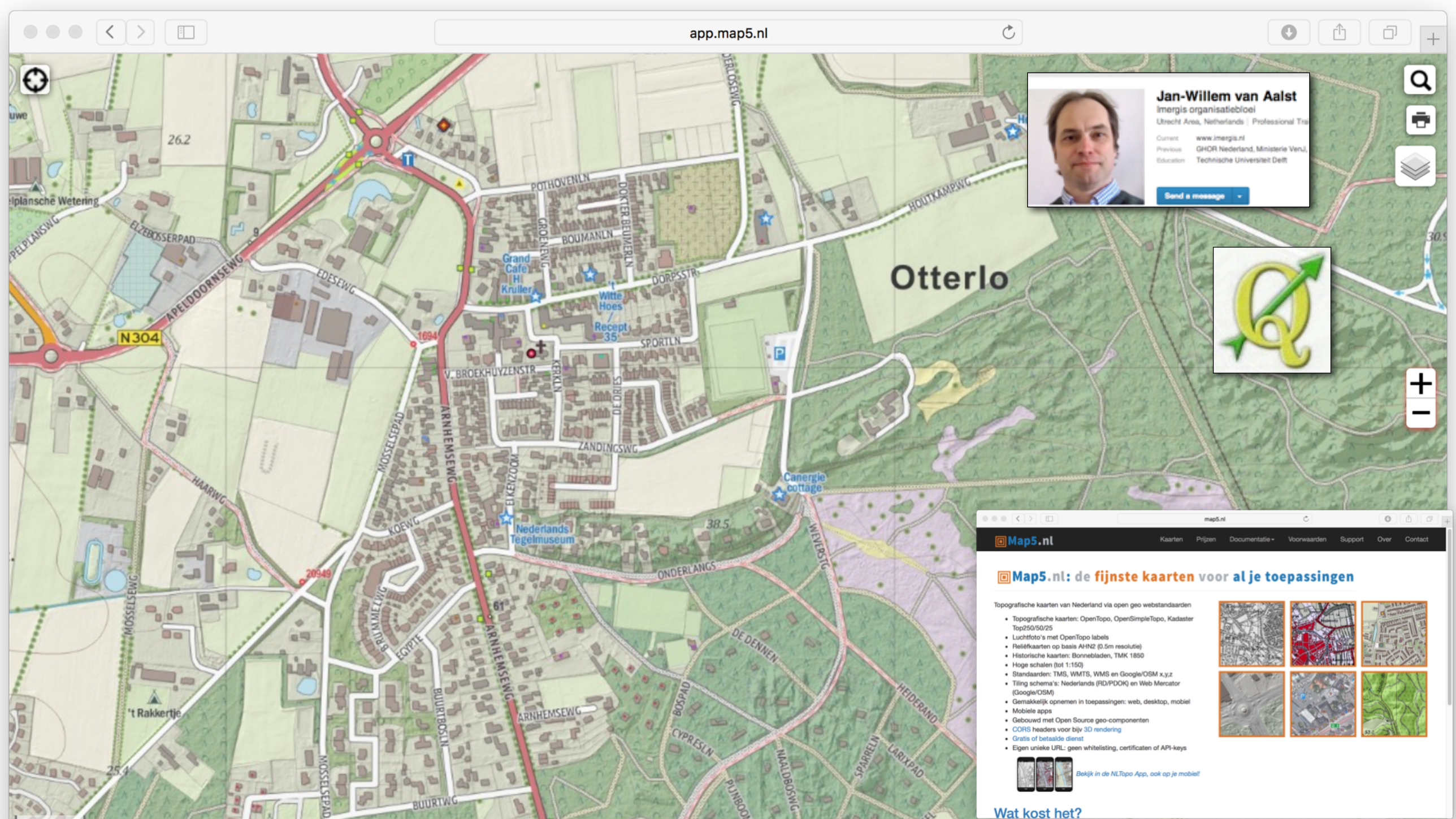
Step 2- Configure Class

Data Structures

- Components exchange *Packets*
- *Packet* contains data and status
- Data formats, e.g. :

  xml_line_stream
  etree_doc
  etree_element (feature)
  etree_element_array
  string
  any
  .
  .

Cases


app.map5.nl
Jan-Willem van Aalst
Imergis organisatiebloei
Send a message
Otterlo
Grand Cafe 't Kruller
't Witte Hoes / Recept 35
Nederlands Tegelmuseum
Canergie cottage
't Rakkertje
N304
APELDOORNSEWG
ARNHEMSEWG
ONDERLANGS
DE DENNEN
Map5.nl
Kaarten
Prijzen
Documentatie
Voorwaarden
Support
Over
Contact
Map5.nl: de fijnste kaarten voor al je toepassingen
Topografische kaarten van Nederland via open geo webstandaarden
Topografische kaarten: OpenTopo, OpenSimpleTopo, Kadaster Top250/50/25
Luchtfoto's met OpenTopo labels
Reliëfkaarten op basis AHN2 (0.5m resolutie)
Historische kaarten: Bonnebladen, TMK 1850
Hoge schalen (tot 1:150)
Standaarden: TMS, WMTS, WMS en Google/OSM x,y,z
Tiling schema's: Nederlands (RD/PDOK) en Web Mercator (Google/OSM)
Gemakkelijk opnemen in toepassingen: web, desktop, mobiel
Mobiele apps
Gebouwd met Open Source geo-componenten
CORS headers voor bijv 3D rendering
Gratis of betaalde dienst
Eigen unieke URL: geen whitelisting, certificaten of API-keys
Bekijk in de NLTopo App, ook op je mobiel!
Wat kost het?

Cases - GML to PostGIS

- National Dutch Open Datasets (GML)

  http://nlextract.nl

  - ✳ Topography: Top10NL, BGT
  - ✳ Cadastral Parcels (BRK)

- Ordnance Survey UK
  - ✳ PoC Topography (OS Mastermap)

Cases - IoT/SensorWeb

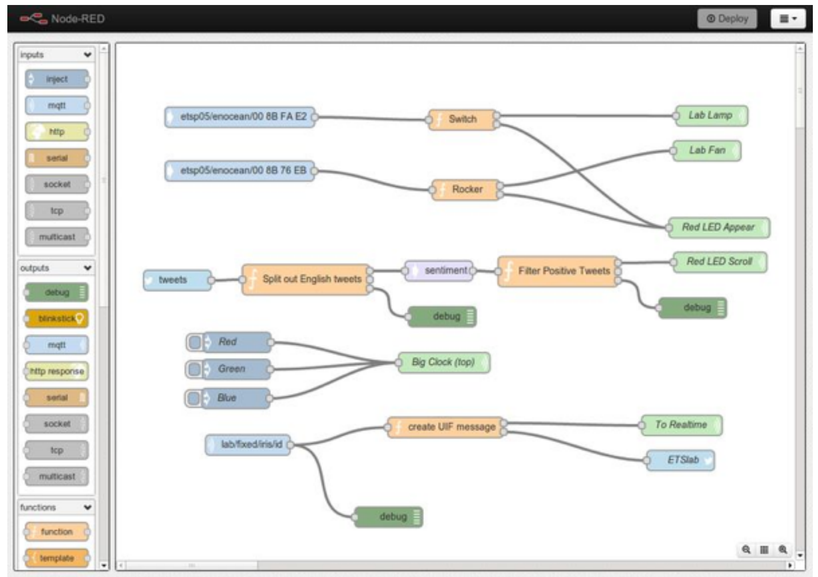
- SOSPilot
  - ✳ Dutch Air Quality Data
  - ✳ publish to PostGIS+SOS (SOS-T)
  - ✳ EU Reporting (Jinja2 Filter)
  - ✳ sospilot.geonovum.nl
- Smart Emission
  - ✳ AQ Sensors hosted by citizens
  - ✳ Calibration/Aggregation
  - ✳ publication to SOS and OGC SensorThings
  - ✳ data.smartemission.nl

Project Status - June 24, 2016

- v1.0.9 installable via PyPi or Docker
- Documentation on www.stetl.org
- Real world transforms done

Project - Planned & in progress

- PyWPS integration
- GUI
  - Flask+Celery+Redis?
  - Node Red?
  - Jupiter Notebook?
- More on GitHub
  github.com/justb4/stetl



Thank You !

www.stetl.org

github.com/justb4/stetl