GIS in Pharo

PharoOWS & GeoView

ESUG
02/07/2025

/me

- Paul Blottiere
  - https://pblottiere.github.io/
  - Software engineer
  - Open-source contributor to GIS software for 10 years

QGIS

PharoOWS

(disclaimer: not a Pharo developer)

GIS: Geographic Information System

- Collect
- Analyze
- Visualize

“Why did the GIS analyst break up with the map?”

Because it had too many issues with its coordinates.

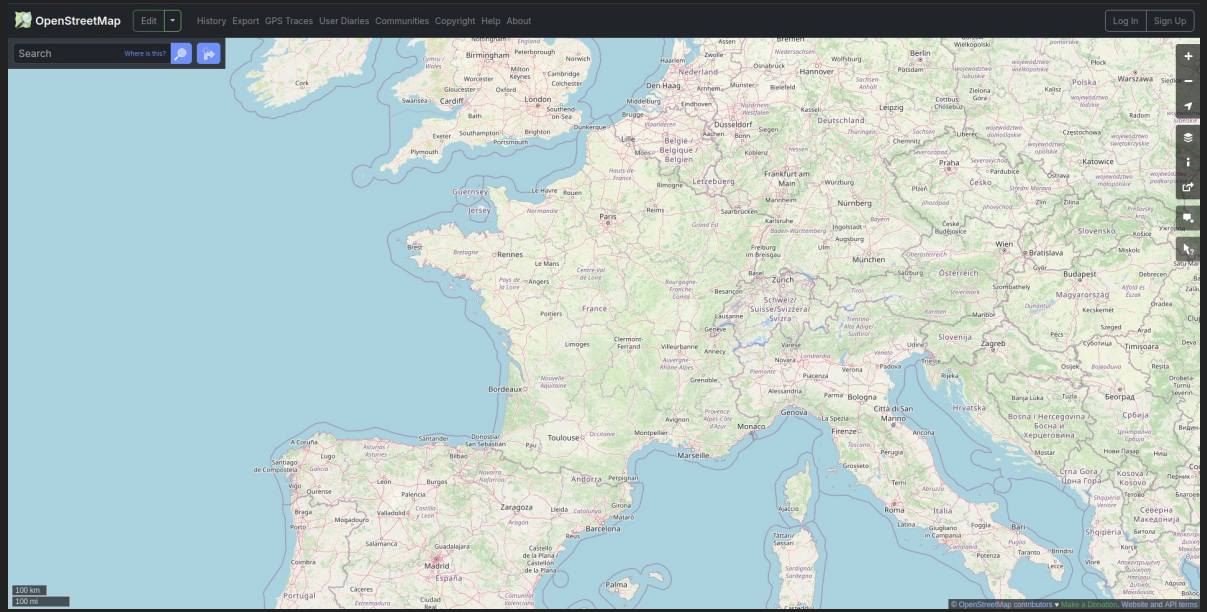
What is PharoOWS?

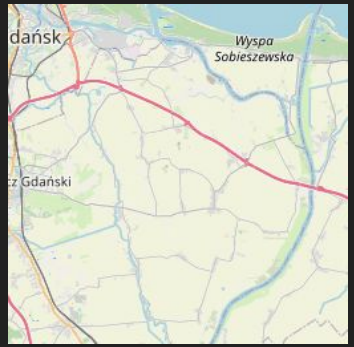
- An open-source GitHub project for Pharo developers
- GIS related

Where do maps come from?

OpenStreetMap : Collaborative geographic database

https://tile.openstreetmap.org/10/565/327.png

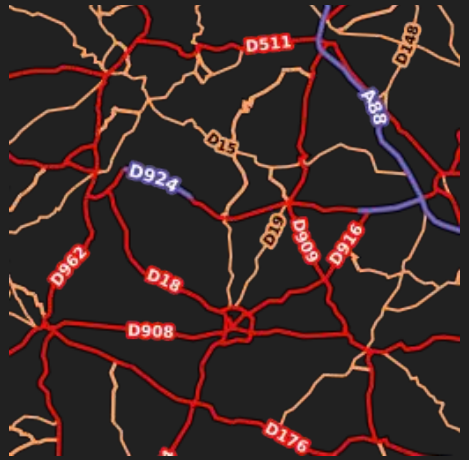




Where do maps come from?

IGN : French National Geographic Institute

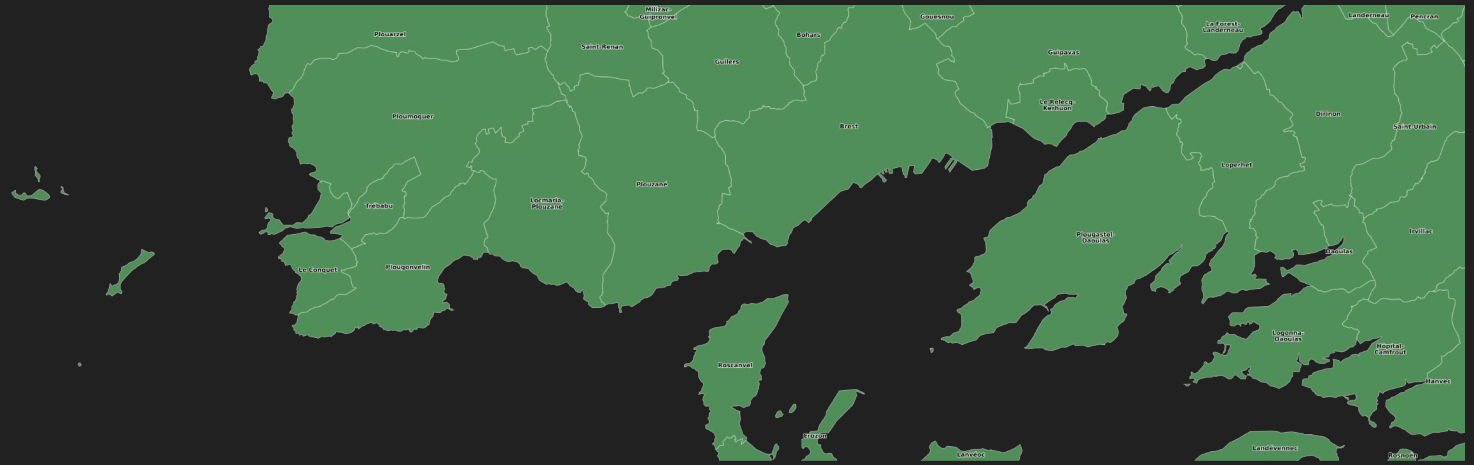
https://data.geopf.fr/wmts?layer=TRANSPORTNETWORKS.ROADS&style=normal&tilematrixset=PM&Service=WMTS&Request=GetTile&Version=1.0.0&Format=image/png&TileMatrix=9&TileCol=255&TileRow=176



Where do maps come from?

IGN : French National Geographic Institute

https://data.geopf.fr/wms-v/ows?SERVICE=WMS&VERSION=1.3.0&REQUEST=GetMap&FORMAT=image%2Fpng&TRANSPARENT=TRUE&LAYERS=CADASTRALPARCELS.GRAPHE&CRS=EPSG%3A3857&STYLES=&WIDTH=2880&HEIGHT=901&BBOX=-546760.6386312215%2C6154901.990418235%2C-467814.24201781943%2C6179600.151303191

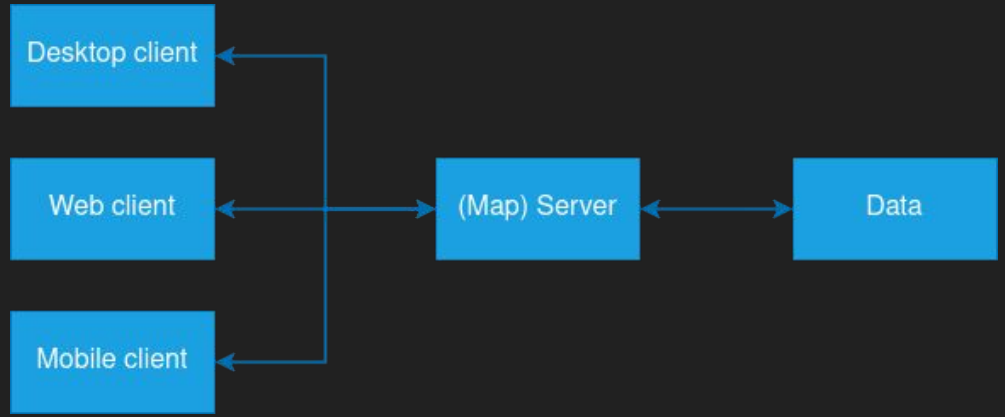


What is PharoOWS?

- An open-source GitHub project for Pharo developers
- GIS related
- Provides a simple API for querying map servers and parsing the data

How do a (map) server and client work together?

- (Map) Server
  - pre-calculated map: Static tile server
  - computed on the fly: Map Server
- Interoperability
  - formal protocols and de facto standards
  - needs an open standards organization for geospatial services



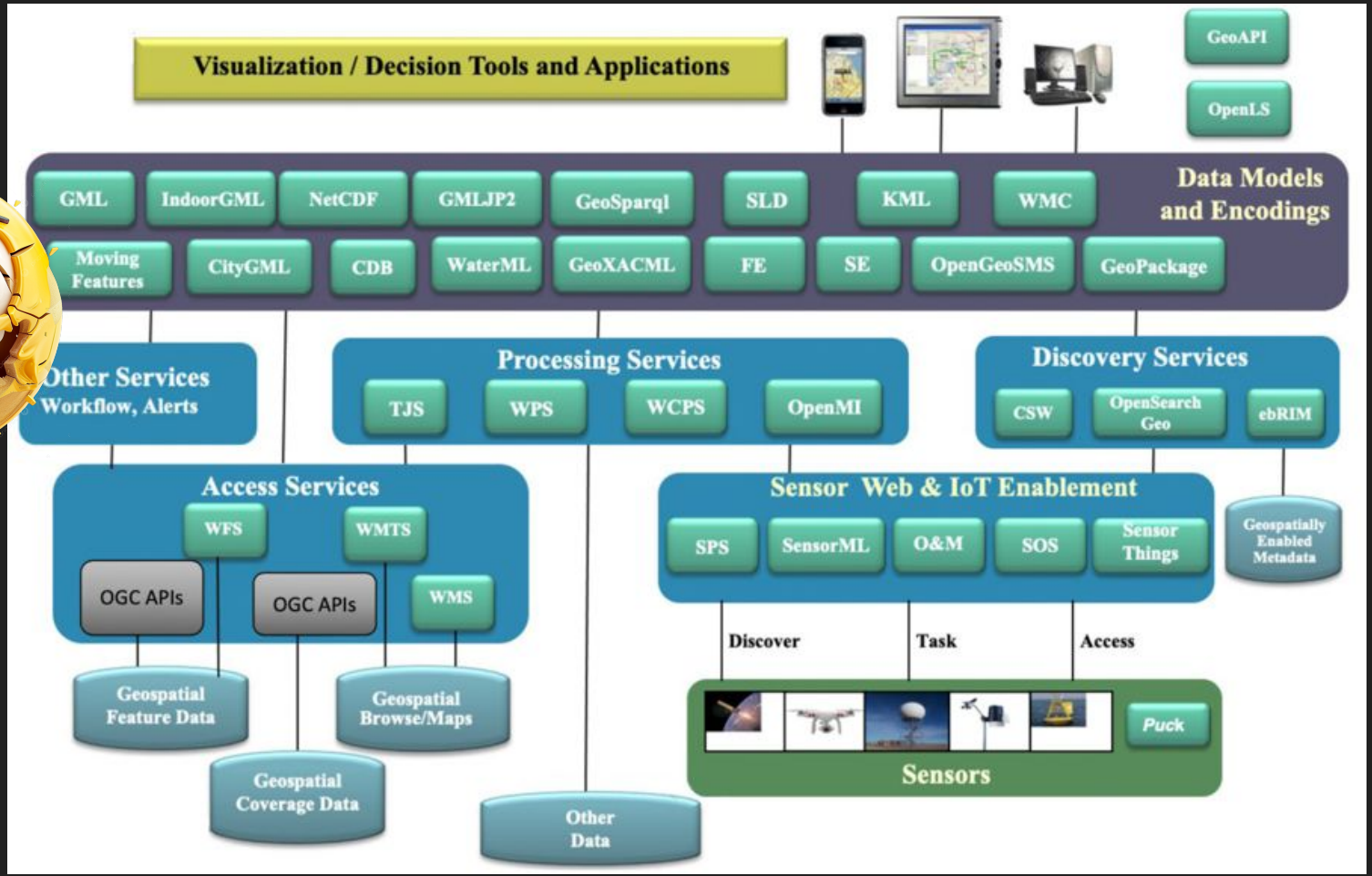
OGC

- Open Geospatial Consortium
  - Defines open geospatial standards
  - Enables data and service interoperability
  - OGC-compliant map servers
    - GeoServer (Java)
    - Map Server (C)
    - QGIS Server (C++)
    - …

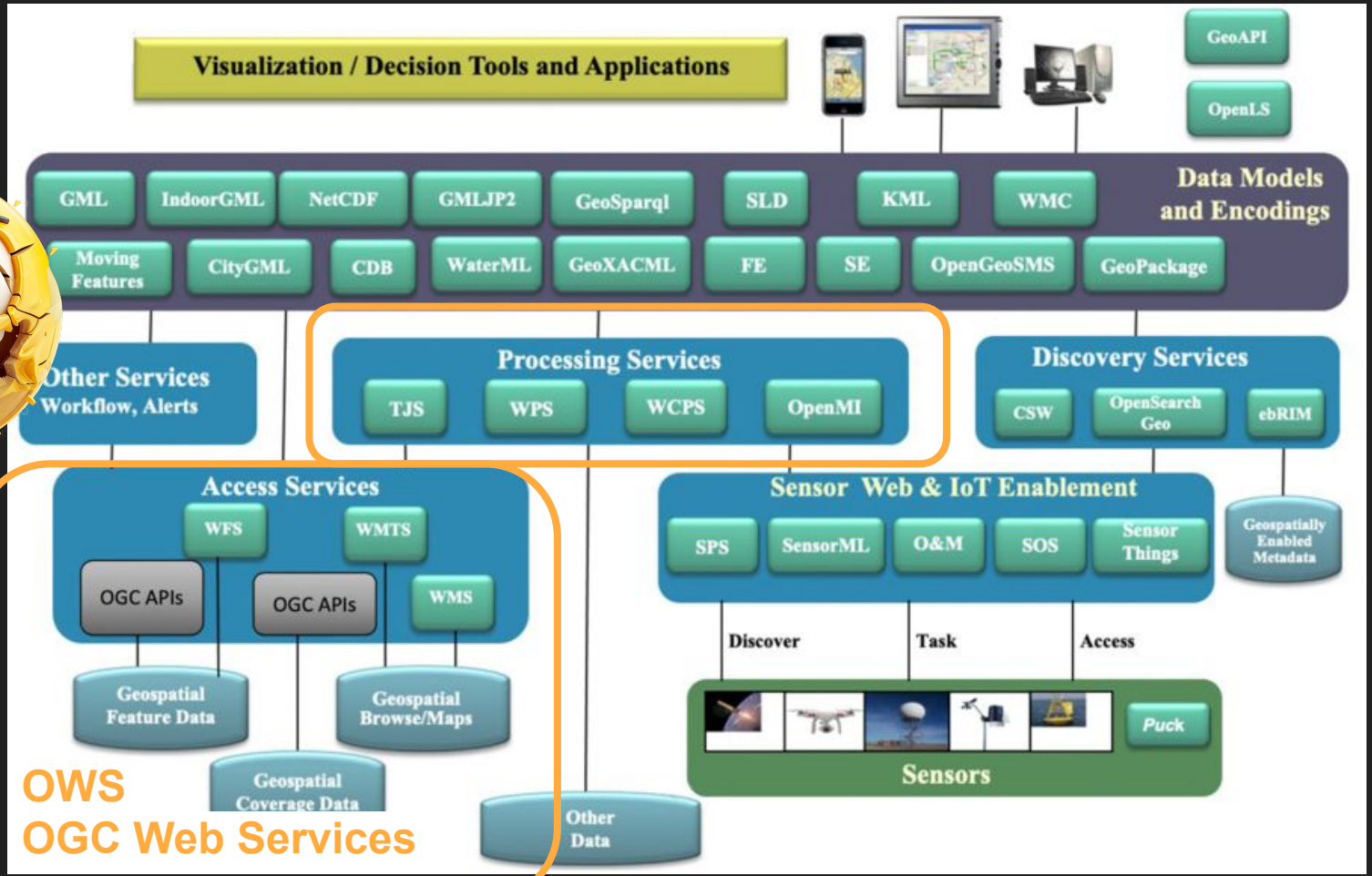
Open Geospatial Consortium®

“The nice thing about standards is that you have so many to choose from.”
— Andrew S. Tanenbaum

OGC


Visualization / Decision Tools and Applications
GeoAPI
OpenLS
GML
IndoorGML
NetCDF
GMLJP2
GeoSparql
SLD
KML
WMC
Data Models and Encodings
Moving Features
CityGML
CDB
WaterML
GeoXACML
FE
SE
OpenGeoSMS
GeoPackage
Other Services
Workflow, Alerts
Processing Services
TJS
WPS
WCPS
OpenMI
Discovery Services
CSW
OpenSearch Geo
ebRIM
Access Services
WFS
WMTS
OGC APIs
OGC APIs
WMS
Sensor Web & IoT Enablement
SPS
SensorML
O&M
SOS
Sensor Things
Geospatially Enabled Metadata
Discover
Task
Access
Geospatial Feature Data
Geospatial Browse/Maps
Geospatial Coverage Data
Other Data
Puck
Sensors

OGC


Visualization / Decision Tools and Applications
GeoAPI
OpenLS
GML
IndoorGML
NetCDF
GMLJP2
GeoSparql
SLD
KML
WMC
Data Models and Encodings
Moving Features
CityGML
CDB
WaterML
GeoXACML
FE
SE
OpenGeoSMS
GeoPackage
Other Services
Workflow, Alerts
Processing Services
TJS
WPS
WCPS
OpenMI
Discovery Services
CSW
OpenSearch Geo
ebRIM
Access Services
WFS
WMTS
OGC APIs
OGC APIs
WMS
Sensor Web & IoT Enablement
SPS
SensorML
O&M
SOS
Sensor Things
Geospatially Enabled Metadata
Discover
Task
Access
Geospatial Feature Data
Geospatial Browse/Maps
Geospatial Coverage Data
Puck
Sensors
Other Data
OWS
OGC Web Services

What is PharoOWS?

- An open-source GitHub project for Pharo developers
- GIS related
- Provides a simple API for querying map servers and parsing the data
- Supports several OGC protocols and de-facto standards

OGC Protocols and de-facto standards

- OGC protocols
  - Traditional Services
    - WMS : Web Map Service
    - WMS-T : Web Map Service with Time dimension
    - WMTS : Web Map Tile Service
    - WFS : Web Feature Service
    - WPS : Web Processing Service
    - …
  - Modern services (REST API)
    - OGC API Features
    - …

- De-facto standards
  - Widely adopted in web mapping
  - XYZ / TMS (Tile Map Service)
  - OpenStreetMap, Google (official API), …

Focus on the OGC WMS protocol

- OGC protocols :
  - Multiple requests per service
  - Numerous parameters per request
  - Vendor-specific parameters and protocols in map server implementations

- WMS
  - Requests:
    - GetCapabilities, GetMap, GetLegendGraphics, GetStyles, GetFeatureInfo, …
    - GetMap
      - VERSION, LAYERS, STYLES, BBOX, WIDTH, HEIGHT, FORMAT, …

Strengths and Limitations

	TMS / XYZ	OGC Protocols
Simplicity	Very easy to implement	More complex setup
Performance	Fast	Slower for on-the-fly rendering protocols (e.g., WMS)
Introspection	None	Rich due to metadata availability
Flexibility	Fixed tiles, limited styling	Supports projections, filters, styling, …
Adoption	Widely used in web mapping	Common in GIS and public infrastructure

What is PharoOWS?

- An open-source GitHub project for Pharo developers
- GIS related
- Provides a simple API for querying map servers and parsing the data
- Supports several OGC protocols and de-facto standards
- Supports
    - TMS / XYZ
    - WMS: GetCapabilities & GetMap
    - WMTS: GetCapabilities & GetTile
    - Coming soon: more protocols, requests and parameters
- https://github.com/ThalesGroup/PharoOWS

Example : create a WMS client

- Create a WMS client targeting the IGN WMS server

```
wms := OWSServiceWMS new.
wms url: 'https://data.geopf.fr/wms-r'.
```

Example : introspect server capabilities

- List available requests

```
operations := wms operations.

# 'operations' is a XMLOrderedList(
#   a OWSServiceWMSOperation <GetCapabilities>
#   a OWSServiceWMSOperation <GetMap>
#   a OWSServiceWMSOperation <GetFeatureInfo>
# )
```

Example : introspect server capabilities

- List available layers

```
layers := wms layers.

# 'layers' is a XMLOrderedList(
#   a OWSServiceWMSLayer(ADMINEXPRESS-COG-CARTO.LATEST)
#   a OWSServiceWMSLayer(ADMINEXPRESS-COG.2017)
#   [...]
# )
```

Example : introspect server capabilities

- List available image formats for GetMap

```
getmap := operations detect: [ :request | request name = 'GetMap' ].
getmap formats.

# 'formats' is an OrderedCollection(
#   image/jpeg
#   image/png
#   image/tiff
#   image/geotiff
#   image/x-bil;bits=32
# )
```

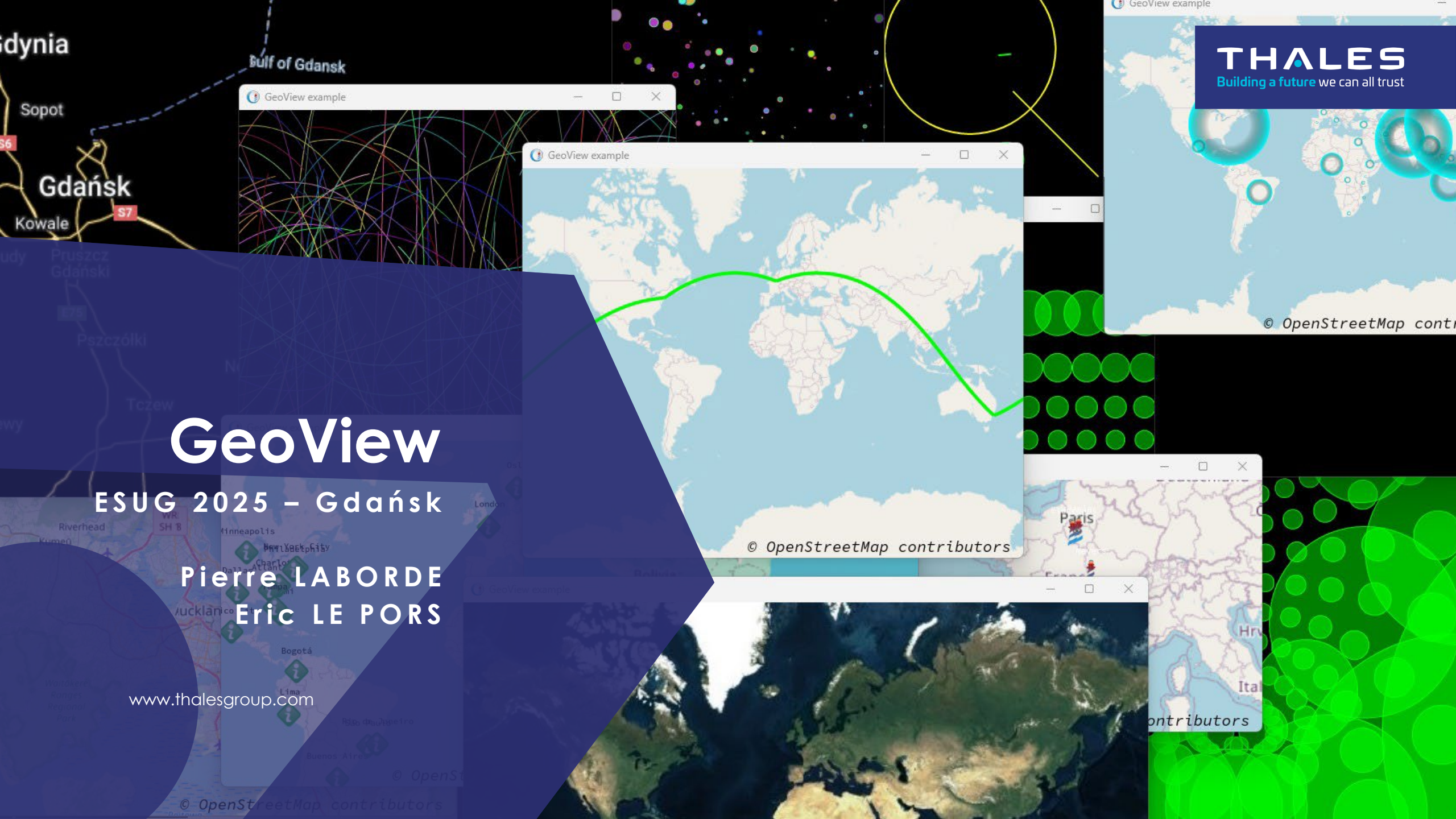
Example : on-the fly rendering

- Download a raster map with two layers for a bounding box defined by EPSG:3857 coordinates

```
map := wms
  map: { 'EL.GridCoverage'. 'FORETS.PUBLIQUES' }
  bbox: (-546079 @ 6126282 corner: -398839 @ 6212047)
  size: 800 @ 600
  epsg: '3857'
  format: 'image/png'.
```

What is PharoOWS?

- An open-source GitHub project for Pharo developers
- GIS related
- Provides a simple API for querying map servers and parsing the data
- Supports several OGC protocols and de-facto standards
- Features
    - TMS / XYZ
    - WMS: GetCapabilities & GetMap
    - WMTS: GetCapabilities & GetTile
    - Coming soon: more protocols, requests and parameters
- https://github.com/ThalesGroup/PharoOWS
- Used by GeoView



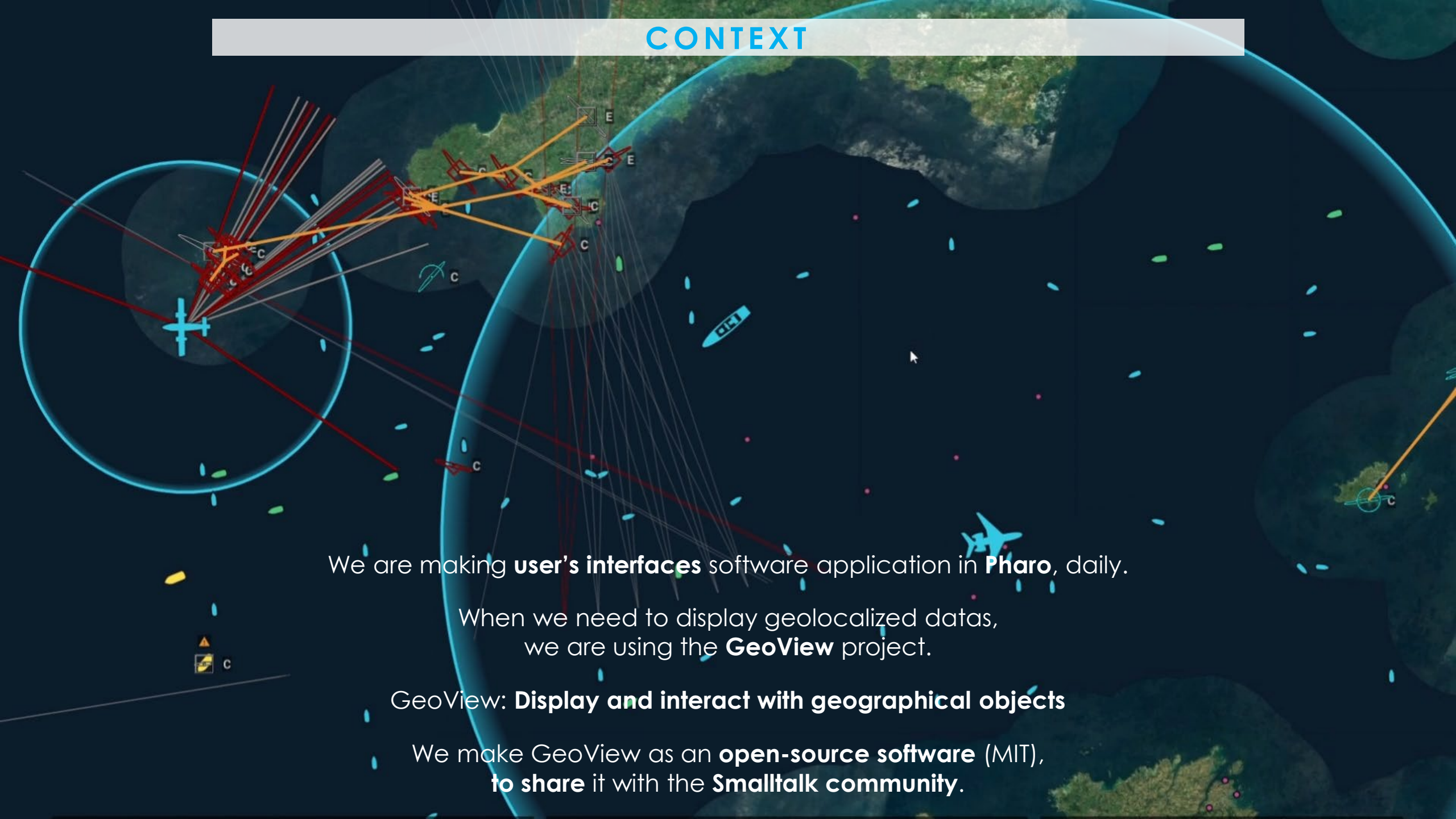
GeoView

ESUG 2025 – Gdańsk

Pierre LABORDE
Eric LE PORS

www.thalesgroup.com

CONTEXT



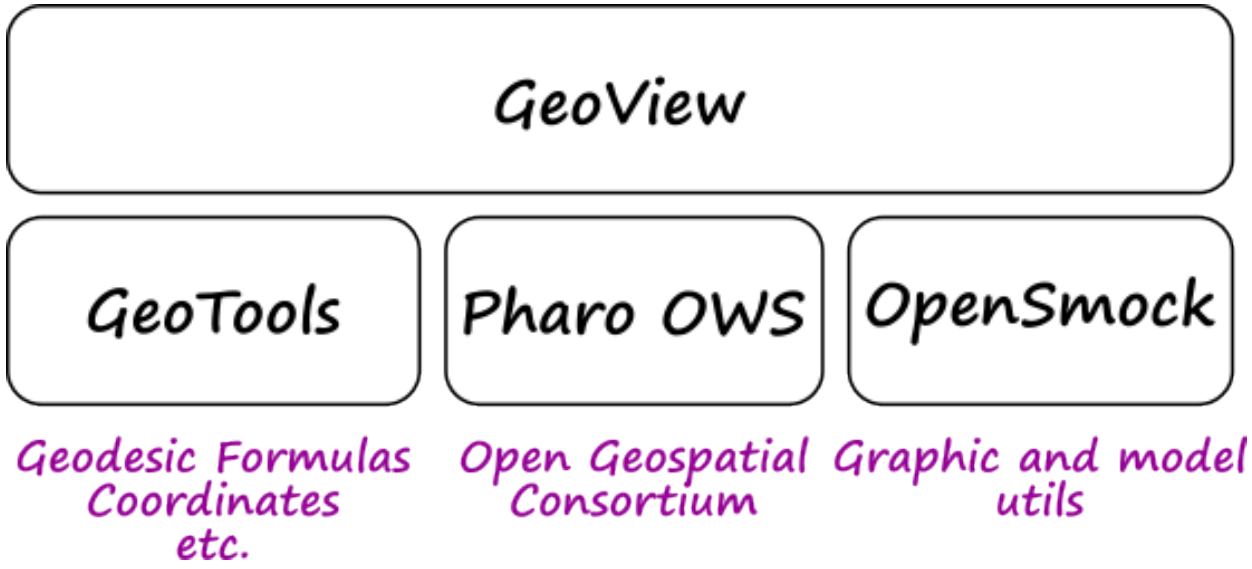
We are making **user's interfaces** software application in **Pharo**, daily.

When we need to display geolocalized datas,
we are using the **GeoView** project.

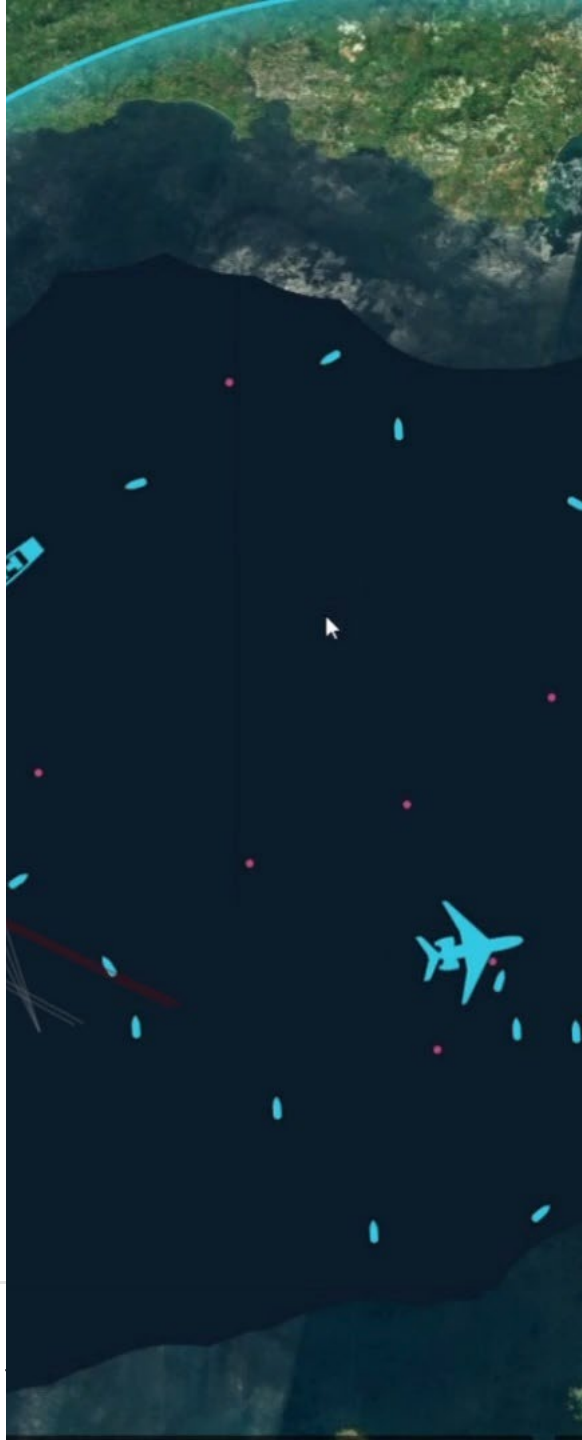
GeoView: **Display and interact with geographical objects**

We make GeoView as an **open-source software** (MIT),
to share it with the **Smalltalk community**.

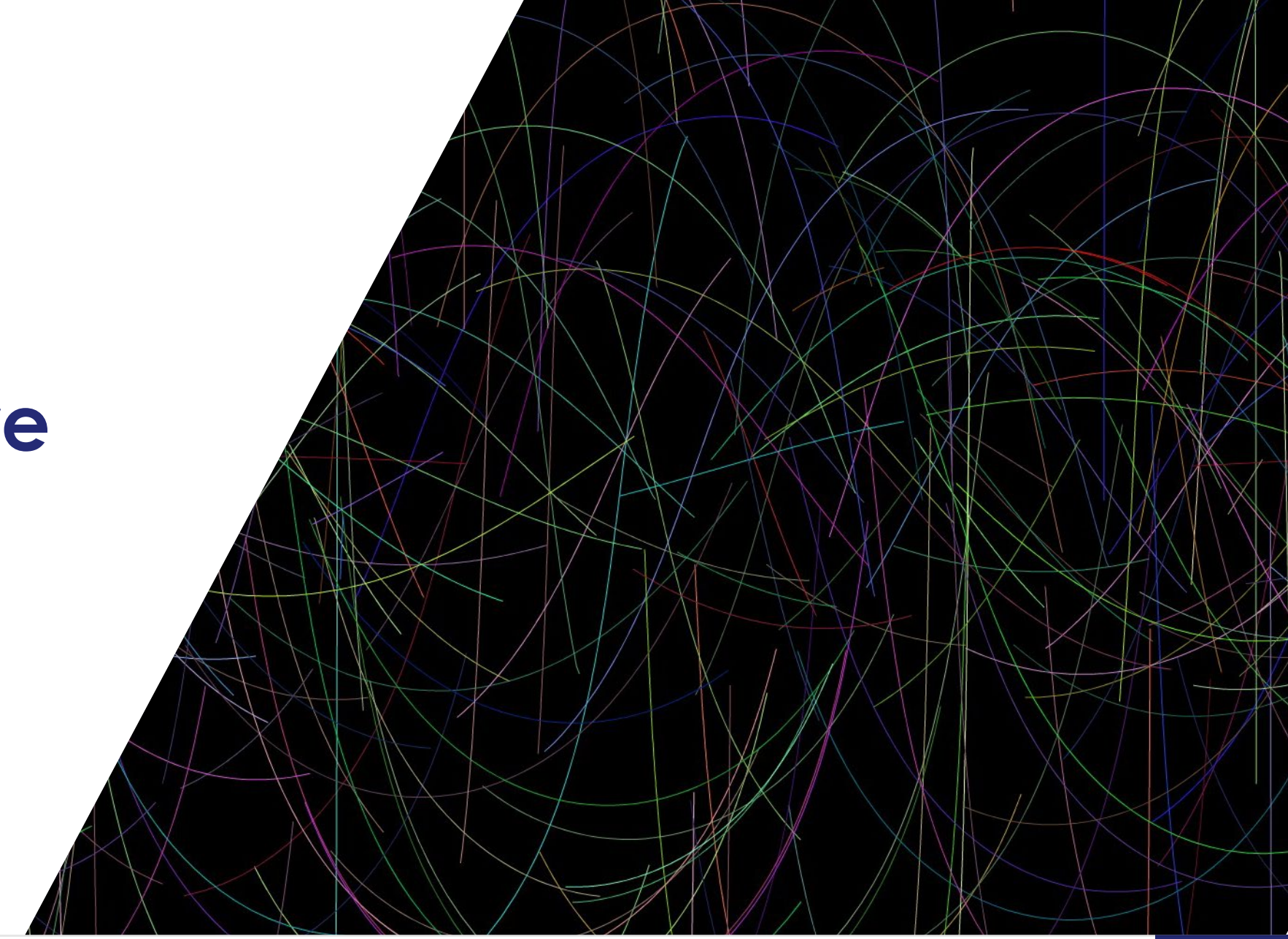
Project composition



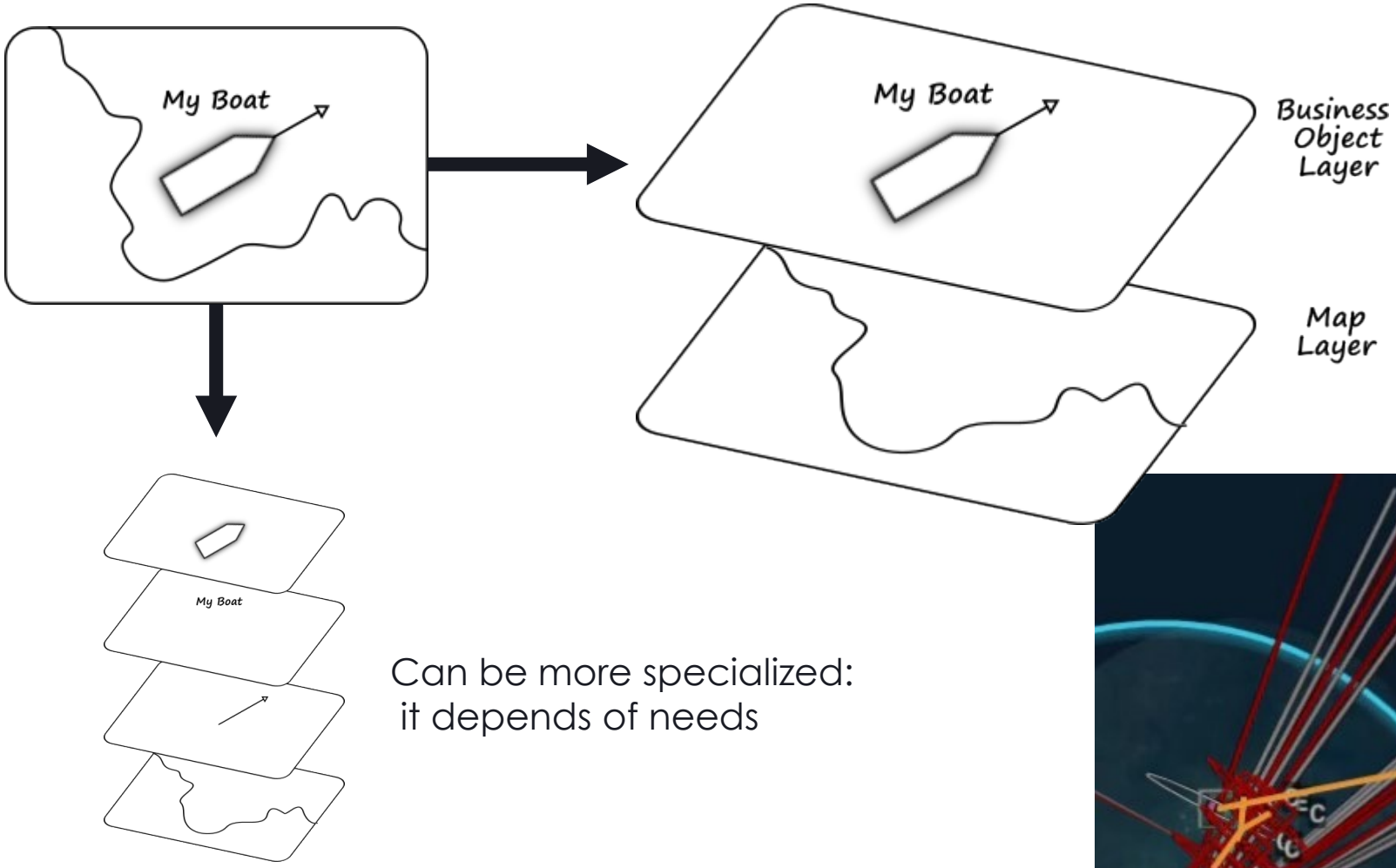
- Pharo and Bloc+Alexandrie implementation
- « Moleculization » available (Not mandatory)



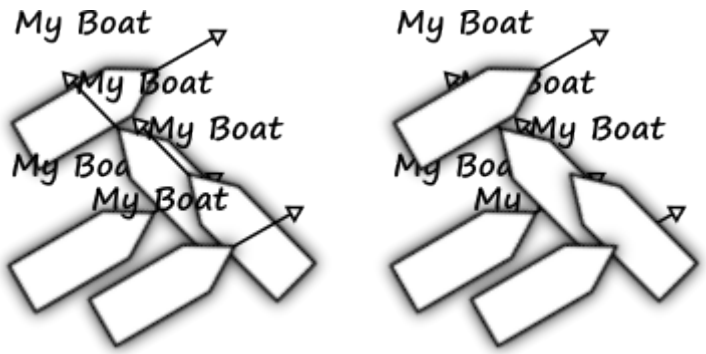
Architecture Overview



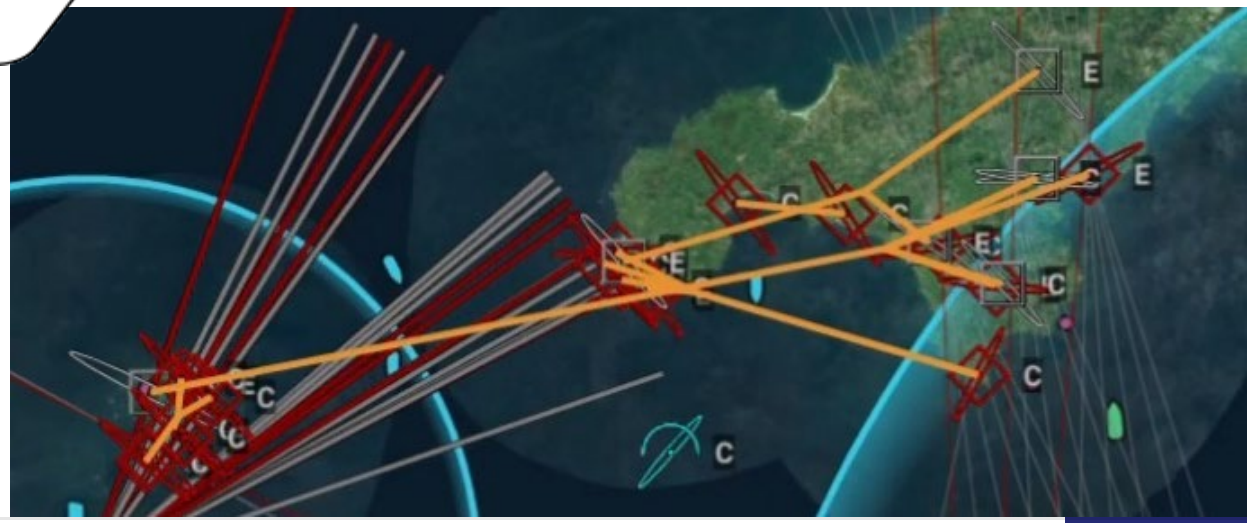
Composing view: the layer stack



In each layer, objects can be ordered (z-order)



Can be more specialized:
it depends of needs

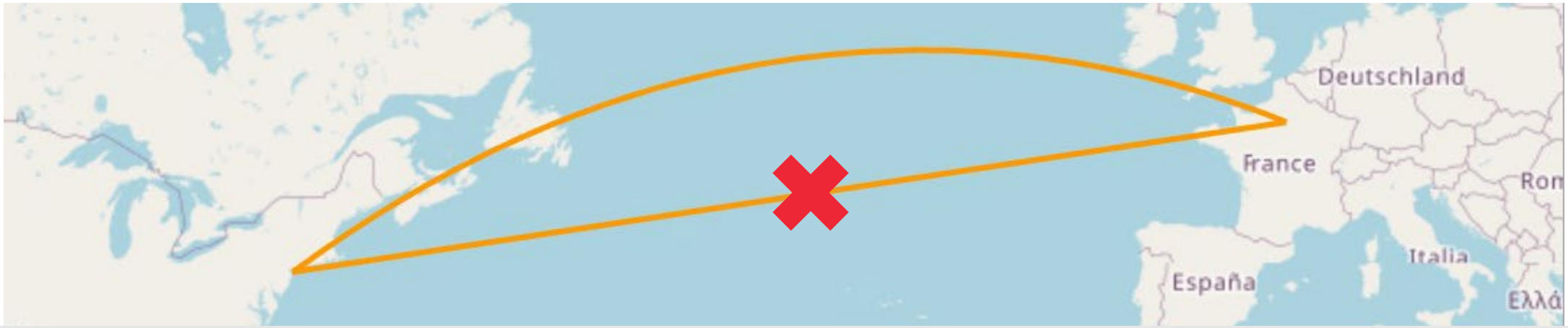


OUR CONSTRAINTS:

SPOILER ALERT !

EARTH IS NOT FLAT !

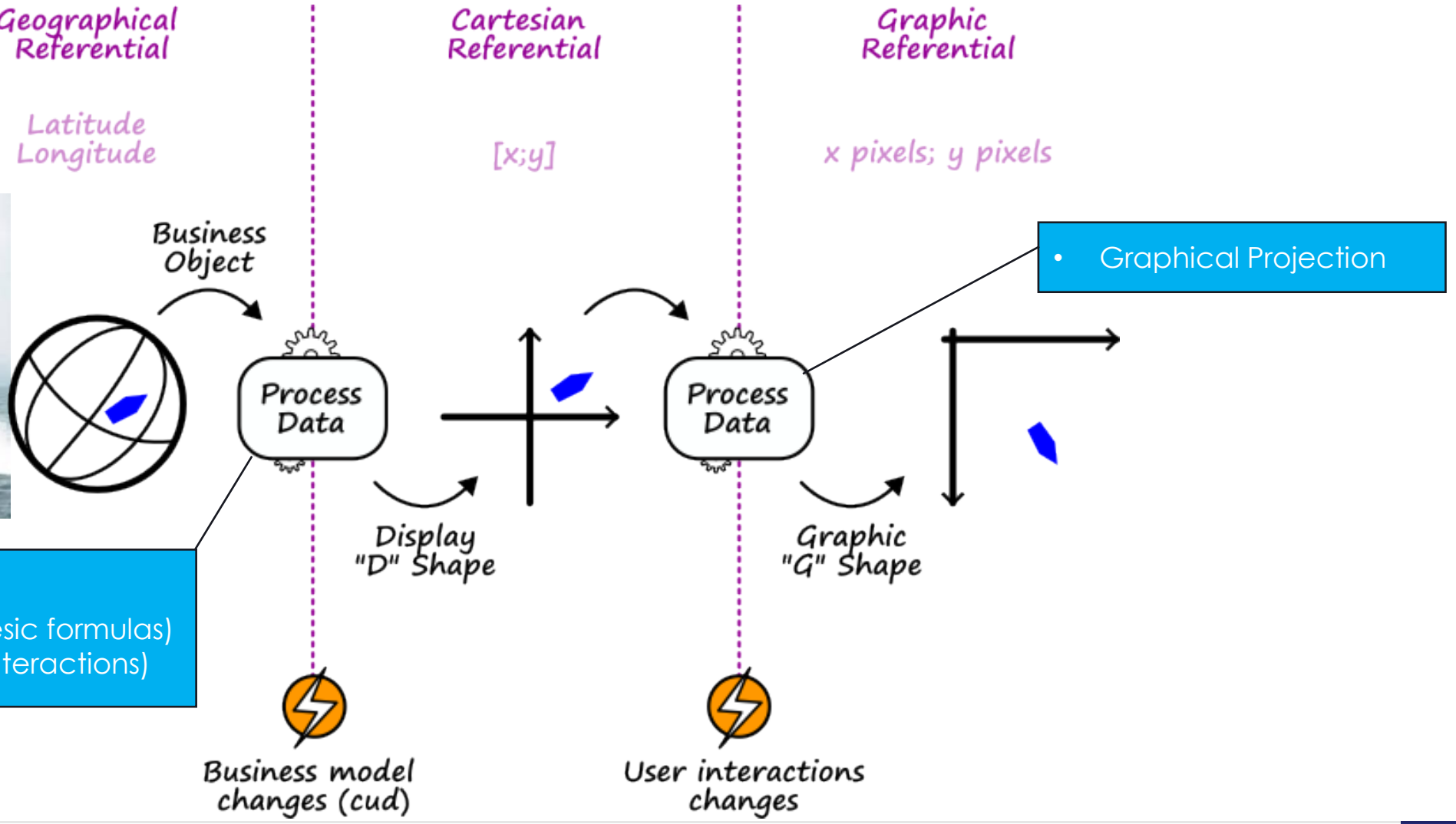
We need to manage cartographic projections to transform a geographical model (latitude, longitude) into a graphic model (pixels).



OPEN

02/07/2025 Thales DMS France / Modèle : 87211168-DOC-GRP-EN-006

Working with 3 types of referentials



- Graphical Projection

- Appearence and styling
- Map Projections (Geodesic formulas)
- States mecanism (user interactions)

02/07/2025 Thales DMS France / Modèle : 87211168-DOC-GRP-EN-006

WHAT ABOUT USER EXPERIENCE?

Use cases depends on the context

- datas and domains,
- work allocation time,
- level of representation,
- etc.

There are several user profiles, with or without geodesic notions, etc.

3 levels of user's API

> Easy : Quick and ready to use

- « GeoObjects » classes hierarchy: Circles, Rectangles, Polylines, Positions, etc. and ready-to-use dedicated layers
- Standard map layers: Open-Street-Map, Google Maps, etc.
- Picking: ask what is it at a pixel point (i.e. a mouse click)? Geo-position, domain objects, etc.
- User events and interactions
- Change view position, scale (=zoom), bounds, map projection, etc.
- Conversions between Geo referential ⇔ Graphic referential

> Standard: Personalize and build your needs

- [Our team works here daily]
- Build your own process-datas to define the apparence of domain objects (Geo -> DShape)
- « DShape » classes hierarchy

> Advance: Customize and extend capacities

- Manage « GShape » behavior
- All process-datas level access
- Customized layers and rendering, etc.

Focus on DShapes (no exhaustive properties list)

> Draw mode

- User: Geographic referential (1 pt = 1 meter)
- Device: Graphic referential (1 pt = 1 pixel)

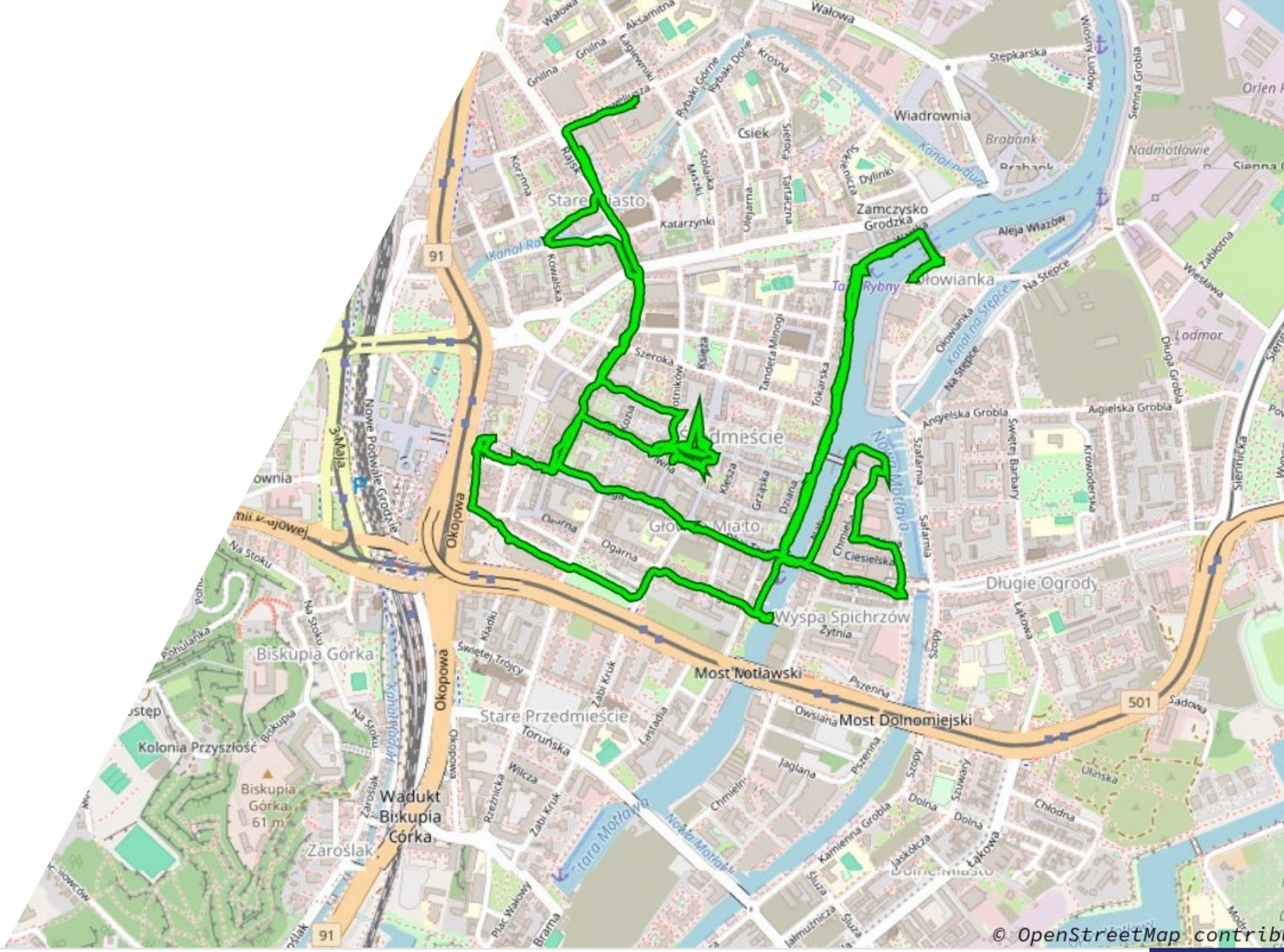
> Style

- Fill, stroke, etc.: dedicated classes (i.e. Gradient, Font, etc.) and system classes when it makes sense (i.e. Color, etc.)
- Of course it depends of shape's type

> And more…

- Image (Form)
- Tree hierarchy model (with children)
- Transformations (i.e. rotation)
- Etc.

Software Integration

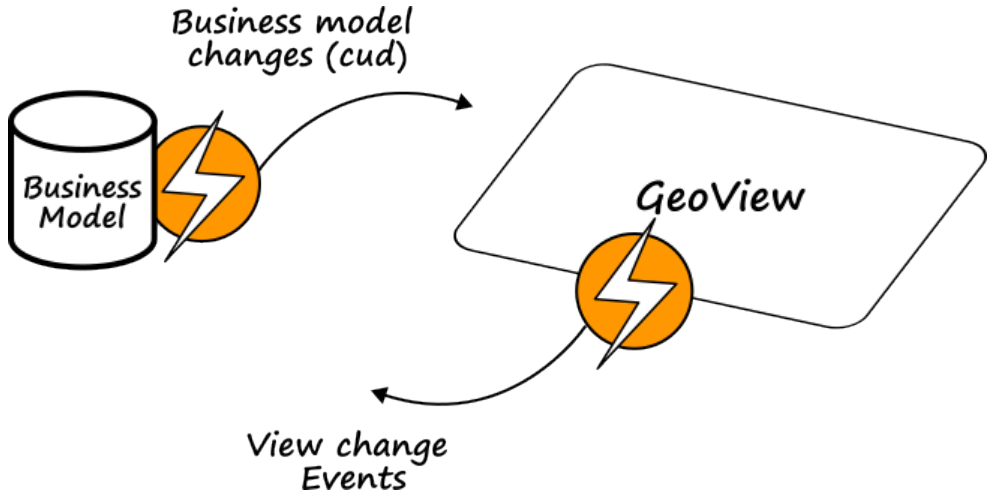


YOU MANAGE YOUR OWN DATAS

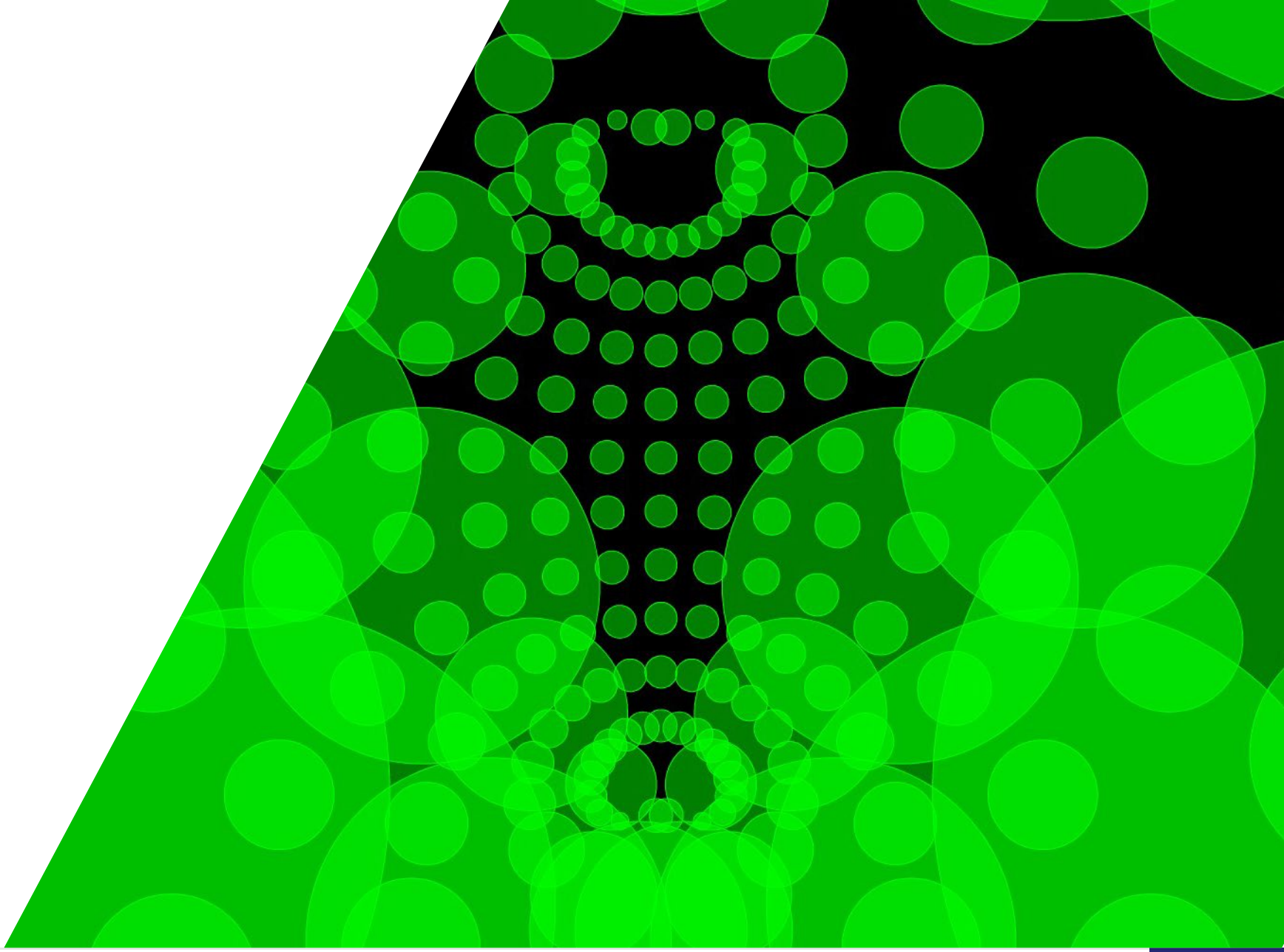
GeoView don’t store the business model.

Need to have an update mecanism to inform when data are changed:
created, updated and deleted.
Don’t forget the update frequency!

GeoView send events
and provide services to read states.



Futur projections



Roadmap

> Actually in beta version, we stabilize our API at each iteration

> Rewrite Map layers, in particularly tiles management (« POC » state)

- Enrich with Pharo OWS to enable other map projections for tiles

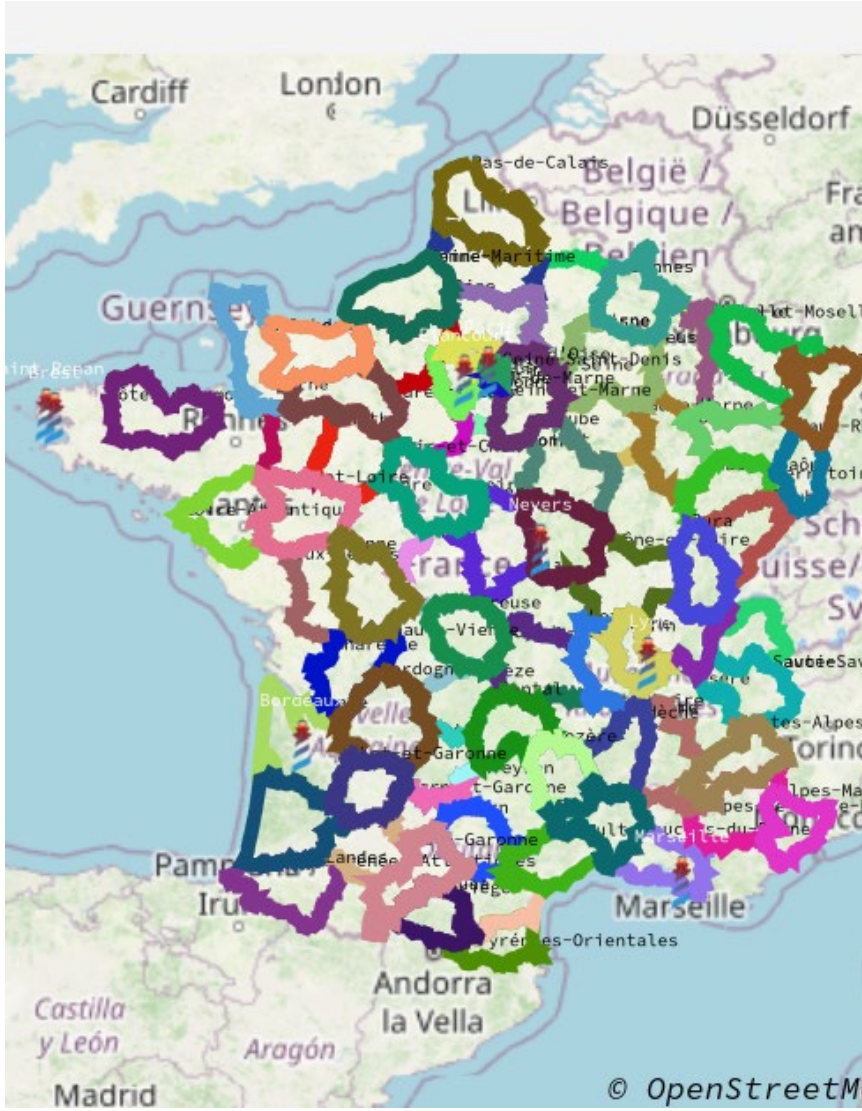
> Add others cartographic background support

- Coast line, Vmap, KML, Shapefile
- Good experimentations using https://github.com/zweidenker/GeoJSON and https://github.com/pharo-GIS/Shapes

> Support more complex projected shapes

- Surfaces and polygons geometric operations: intersection, union, centroid, simplification, etc. (OGC-Pharo?)

> Enrich the « Easy level » user API



Reflexions

> High density layers (> 1000 objects)

- Clustering, heatmap, etc.
- Server/Local processing, cache, etc.
- Multi-threading

> Smooth map transitions

> Graphical tool to select map server layer

> Animated objects

- GIF
- Object selection rendering

> 3D (Woden?)

You're welcome to participate!

THALES
Building a future we can all trust

Thanks

github.com/ThalesGroup/GeoView