# Etat d'avancement du stage – 20/06 – 24/06

**Bérénice Le Mao**

# Sujet : Quelle carte multi-échelle est utile aux gestionnaires de risque après un séisme ?

## I) Tâches réalisées cette semaine :

### A) Ecriture du code Openlayers

Méthodologie :

Je dois écrire deux codes d'Openlayers. Sur le premier, j'importe toutes mes couches pour pouvoir définir pour chacune d'elles l'échelle adaptée. Le choix doit également se réfléchir en lien avec les autres données, et dans la fluidité de l'ensemble. C'est pourquoi le visualiser sur une carte interactive est plus efficace que simplement le visualiser sur QGIS.

Malgré tout, un tel code charge une quantité assez importante de données pour chaque zoom, ce qui entraîne de la latence. C'est pourquoi je dois également écrire un deuxième code avec pour chaque échelle une agrégation de couches préalablement groupées sur geoserver.

pb qui apparaissent

- openlayers (ni geoserver) ne semble pas lire les fichiers de style comprenant des étiquettes et autre information contenue dans la couche attributaire.
- Pour régler le problème d'affichage des noms et des nombres contenus dans la fiche attributaire, je dois alors modifier le code (=TextSymbolizer) du fichier de style dans Openlayers pour l'ajouter car je remarque que l'import sur geoserver n'a pas gardé le nom du champ en majuscule.
- Openlayers ne semble pas non plus lire la variation de taille des points, créée avec l'aide de l'assistant de QGIS. J'ai donc repris les tailles générées automatiquement par l'assistant et je les ai ajoutée manuellement à la symbologie pour faire varier la taille chaque classe de points.

Résultats :

1er code :

```
var defaultZoom = 7.1;

var mapView = new ol.View ({
    center: ol.proj.fromLonLat([7, 43.5]),
    zoom: defaultZoom
});

var map = new ol.Map ({
    target: 'map',
    view: mapView
});

var noneTile = new ol.layer.Tile({
    title: 'None',
    type: 'base',
    visible: false
});

var osmTile = new ol.layer.Tile ({
    title: 'OpenStreetMap',
    visible: true,
    type: 'base',
    source: new ol.source.OSM()
});

// map.addLayer(osmTile);
var baseGroup = new ol.layer.Group({
    title: 'Base Maps',
    fold: true,
    layers: [osmTile, noneTile]
});

/* Déclaration de la source de la couche en format WMS */
var epicentre = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'epicentre', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var epicentre_layer = new ol.layer.Tile({source : epicentre})

/* Déclaration de la source de la couche en format WMS */
var region1 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'region', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var region_layer = new ol.layer.Tile({source : region1})
```

```
/* Déclaration de la source de la couche en format WMS */
var region2 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'region_1', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var region2_layer = new ol.layer.Tile({source : region2})

/* Déclaration de la source de la couche en format WMS */
var intensite = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'intensite seisme', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var intensite_layer = new ol.layer.Tile({source : intensite})

// Prefecture

/* Déclaration de la source de la couche en format WMS */
var prefREG = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'prefecture_reg', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var prefREG_layer = new ol.layer.Tile({source : prefREG})
prefREG_layer.setZIndex(1)

/* Déclaration de la source de la couche en format WMS */
var prefDEP = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'prefecture_dep', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var prefDEP_layer = new ol.layer.Tile({source : prefDEP})
prefDEP_layer.setZIndex(1)

/* Déclaration de la source de la couche en format WMS */
var souspref = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'sous-prefecture', 'TILED': true},
    serverType: 'geoserver',
    visible: true
```

```
});
var souspref_layer = new ol.layer.Tile({source : souspref})
souspref_layer.setZIndex(1)

/* Déclaration de la source de la couche en format WMS */
var clusterREG = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'cluster_reg2', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var clusterREG_layer = new ol.layer.Tile({source : clusterREG})
clusterREG_layer.setZIndex(1)

/* Déclaration de la source de la couche en format WMS */
var clusterDEP = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'cluster_dep3', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var clusterDEP_layer = new ol.layer.Tile({source : clusterDEP})
clusterDEP_layer.setZIndex(1)

/* Déclaration de la source de la couche en format WMS */
var clusterIRIS = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'cluster_iris2', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var clusterIRIS_layer = new ol.layer.Tile({source : clusterIRIS})
clusterIRIS_layer.setZIndex(1)

// Route

var route1 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'route_1_bon', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var route1_layer = new ol.layer.Tile({source : route1})

var route2 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'route_2_bon', 'TILED': true},
```

```
    serverType: 'geoserver',
    visible: true
});
var route2_layer = new ol.layer.Tile({source : route2})

var route3 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'route_3_bon', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var route3_layer = new ol.layer.Tile({source : route3})

var route4 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'route_4_bon', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var route4_layer = new ol.layer.Tile({source : route4})

// Cours d'eau

var eau1 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'reseau_hydro_1', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var eau1_layer = new ol.layer.Tile({source : eau1})

var eau2 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'reseau_hydro_2', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var eau2_layer = new ol.layer.Tile({source : eau2})

var eau3 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'reseau_hydro_3', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var eau3_layer = new ol.layer.Tile({source : eau3})
```

```
var eau4 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'reseau_hydro_4', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var eau4_layer = new ol.layer.Tile({source : eau4})

var eau5 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'reseau_hydro_5', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var eau5_layer = new ol.layer.Tile({source : eau5})

var surfaceO = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'surface_hydro_1', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var surfaceO_layer = new ol.layer.Tile({source : surfaceO})

// Voie ferrée

var train1 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'voie_ferree_1', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var train1_layer = new ol.layer.Tile({source : train1})
train1_layer.setZIndex(1)

var train2 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'voie_ferree_2', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var train2_layer = new ol.layer.Tile({source : train2})

var train3 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
```

```
    params: {'LAYERS': 'voie_ferree_3', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var train3_layer = new ol.layer.Tile({source : train3})

// Base d'enjeu

var caserne = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'caserne', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var caserne_layer = new ol.layer.Tile({source : caserne})

var hopital = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'hopital_2', 'TILED': true},
    serverType: 'geoserver',
    visible: true
});
var hopital_layer = new ol.layer.Tile({source : hopital})

map.addLayer(baseGroup);

var SeismeTile_p = new ol.layer.Tile ({
    title: "dégâts matériels importants",
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_dep_urgent', 'TILED': true},
        serverType: 'geoserver',
        visible: true
    })
});

var SeismeTile_s = new ol.layer.Tile ({
    title: "dégâts matériels non prioritaires",
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_dep_secondaires', 'TILED': true},
        serverType: 'geoserver',
    })
});
SeismeTile_s.setVisible(!SeismeTile_s.getVisible());

/* Déclaration de la couche WMS */
var couche_8 = new ol.layer.Group({
    layers: [SeismeTile_s, SeismeTile_p]
});

var SeismeTile_p2 = new ol.layer.Tile ({
    title: "dégâts matériels importants",
```

```
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_communes_urgent', 'TILED': true},
        serverType: 'geoserver',
        visible: true
    })
});

var SeismeTile_s2 = new ol.layer.Tile ({
    title: "dégâts matériels non prioritaires",
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_communes_secondaires', 'TILED':
true},
        serverType: 'geoserver',
    })
});
SeismeTile_s2.setVisible(!SeismeTile_s2.getVisible());

/* Déclaration de la couche WMS */
var couche_10 = new ol.layer.Group({
    layers: [SeismeTile_s2, SeismeTile_p2]
});

var SeismeTile_p3 = new ol.layer.Tile ({
    title: "dégâts matériels importants",
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_iris_urgent', 'TILED': true},
        serverType: 'geoserver',
        visible: true
    })
});

var SeismeTile_s3 = new ol.layer.Tile ({
    title: "dégâts matériels non prioritaires",
    source: new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'batiment_iris_secondaires', 'TILED': true},
        serverType: 'geoserver',
    })
});
SeismeTile_s3.setVisible(!SeismeTile_s3.getVisible());

/* Déclaration de la couche WMS */
var couche_11 = new ol.layer.Group({
    layers: [SeismeTile_s3, SeismeTile_p3]
});

// map.addLayer(SeismeTile);

// var overlayGroup = new ol.layer.Group({
//     title: 'Overlays',
//     fold: true,
//     layers: [SeismeTile_s3, SeismeTile_p3, SeismeTile_s2,
SeismeTile_p2, SeismeTile_s, SeismeTile_p]
// });
```

```
// map.addLayer(overlayGroup);

map.addLayer(epicentre_layer);
map.addLayer(SeismeTile_p);

var zoomDiv = document.getElementById('zoom');
zoomDiv.innerHTML = 'Zoom : ' + defaultZoom;

map.on("moveend",  function () {
    zoomDiv.innerHTML = 'Zoom : ' + map.getView().getZoom().toFixed(2);
    if(mapView !=parseInt(map.getView().getZoom()) ){
        mapView =parseInt(map.getView().getZoom());

    if(parseFloat(map.getView().getZoom()) <= 8 &&
parseFloat(map.getView().getZoom()) > 7) {
        map.addLayer(prefDEP_layer);
        map.addLayer(prefREG_layer);
        map.addLayer(eau1_layer);
        map.addLayer(eau2_layer);
        map.addLayer(train1_layer);
        map.addLayer(route1_layer);
        map.addLayer(SeismeTile_p);
        map.removeLayer(SeismeTile_p2);
        map.removeLayer(SeismeTile_p3);
        map.removeLayer(intensite_layer);
        map.removeLayer(clusterREG_layer);
        map.removeLayer(region_layer);
        map.removeLayer(souspref_layer);
        map.removeLayer(route2_layer);
        map.removeLayer(eau2_layer);

    }else if(parseFloat(map.getView().getZoom()) <= 6) {
        map.addLayer(region2_layer);
        map.addLayer(intensite_layer);
        map.removeLayer(region_layer);
        map.removeLayer(eau1_layer);
        map.removeLayer(prefREG_layer);
        map.removeLayer(prefDEP_layer);
        map.removeLayer(SeismeTile_p);


    }else if(parseFloat(map.getView().getZoom()) <= 7 &&
parseFloat(map.getView().getZoom()) > 6) {
        map.addLayer(train1_layer);
        map.addLayer(eau1_layer);
        map.addLayer(route1_layer);
        map.addLayer(prefREG_layer);
        map.addLayer(intensite_layer);
        map.addLayer(region_layer);
        map.removeLayer(SeismeTile_p);
        map.removeLayer(SeismeTile_p2);
        map.removeLayer(SeismeTile_p3);
        map.removeLayer(prefDEP_layer);
        map.removeLayer(clusterREG_layer);
        map.removeLayer(region2_layer);
```

```
    }else if(parseFloat(map.getView().getZoom()) <= 9 &&
parseFloat(map.getView().getZoom()) > 8) {
        map.addLayer(eau3_layer);
        map.addLayer(route2_layer);
        map.addLayer(train2_layer);
        map.addLayer(clusterREG_layer);
        map.addLayer(souspref_layer);
        map.addLayer(SeismeTile_p);
        map.removeLayer(SeismeTile_p2);
        map.removeLayer(prefREG_layer);
        map.removeLayer(clusterDEP_layer);
        map.removeLayer(clusterIRIS_layer);

    }else if(parseFloat(map.getView().getZoom()) <= 10 &&
parseFloat(map.getView().getZoom()) > 9) {
        map.addLayer(hopital_layer);
        map.addLayer(SeismeTile_p2);
        map.addLayer(clusterDEP_layer);
        map.removeLayer(clusterREG_layer);
        map.removeLayer(SeismeTile_p);
        map.removeLayer(SeismeTile_p3);
        map.removeLayer(clusterIRIS_layer);
        map.removeLayer(clusterREG_layer);

    }else if(parseFloat(map.getView().getZoom()) <= 11 &&
parseFloat(map.getView().getZoom()) > 10) {
        map.addLayer(train3_layer);
        map.addLayer(route3_layer);
        map.addLayer(eau4_layer);
        map.removeLayer(eau5_layer);
        map.removeLayer(caserne_layer);
        map.removeLayer(SeismeTile_p3);
        map.removeLayer(clusterDEP_layer);

    }else if(parseFloat(map.getView().getZoom()) <= 12 &&
parseFloat(map.getView().getZoom()) > 11) {
        map.addLayer(caserne_layer);
        map.addLayer(SeismeTile_p3);
        map.addLayer(eau5_layer);
        map.addLayer(clusterDEP_layer);
        map.removeLayer(SeismeTile_p);
        map.removeLayer(SeismeTile_p2);
        map.removeLayer(clusterIRIS_layer);
        map.removeLayer(route4_layer);

    }else if(parseFloat(map.getView().getZoom()) > 12) {
        map.addLayer(clusterIRIS_layer);
        map.addLayer(route4_layer);
        map.removeLayer(clusterDEP_layer);
        }
    }
})

var layerSwitcher = new ol.control.LayerSwitcher({
     activationMode: 'click',
     startActive: false,
     groupSelectStyle: 'children'
});
```

```
map.addControl(layerSwitcher);

function toggleLayer(eve) {
    var lyrname = eve.target.value;
    var checkedStatus = eve.target.checked;
    var lyrList = map.getLayers();

    lyrList.forEach(function(element){
        if (lyrname == element.get('title')){
            element.setVisible(checkedStatus);
        }
    })
};

var mousePosition = new ol.control.MousePosition({
    className: 'mousePosition',
    projection: 'EPSG:4326',
    coordinateFormat: function(coordinate){return
ol.coordinate.format(coordinate,'{y} , {x}', 6);}
});

map.addControl(mousePosition);

var scaleControl = new ol.control.ScaleLine({
    bar: true,
    text: true,

});

map.addControl(scaleControl);

// var zoomDiv = document.getElementById('zoom');
// zoomDiv.innerHTML = 'Zoom : ' + defaultZoom;

// map.on('moveend', function() {
//     zoomDiv.innerHTML = 'Zoom : ' +
mapViewmap.getView().getZoom().toFixed(2);
// })
```

2e code (plus court) :

```
var defaultZoom = 7.1;

var mapView = new ol.View ({
    center: ol.proj.fromLonLat([7, 43.5]),
    zoom: defaultZoom
});

var map = new ol.Map ({
    target: 'map',
    view: mapView
});

var noneTile = new ol.layer.Tile({
    title: 'None',
```

```
    type: 'base',
    visible: false
});

var osmTile = new ol.layer.Tile ({
    title: 'OpenStreetMap',
    visible: true,
    type: 'base',
    source: new ol.source.OSM()
});

// map.addLayer(osmTile);
var baseGroup = new ol.layer.Group({
    title: 'Base Maps',
    fold: true,
    layers: [osmTile, noneTile]
});

map.addLayer(baseGroup);

var sourceWMS_6 = new ol.source.TileWMS({
        url: 'http://localhost:8080/geoserver/Nice/wms',
        params: {'LAYERS':'zoom6', 'TILED': true},
        serverType: 'geoserver',
});
/* Déclaration de la couche WMS */
var couche_6 = new ol.layer.Tile({ source: sourceWMS_6 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_7 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom7', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_7 = new ol.layer.Tile({ source: sourceWMS_7 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_8 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom8', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_8 = new ol.layer.Tile({ source: sourceWMS_8 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_9 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom9', 'TILED': false},
    serverType: 'geoserver'
```

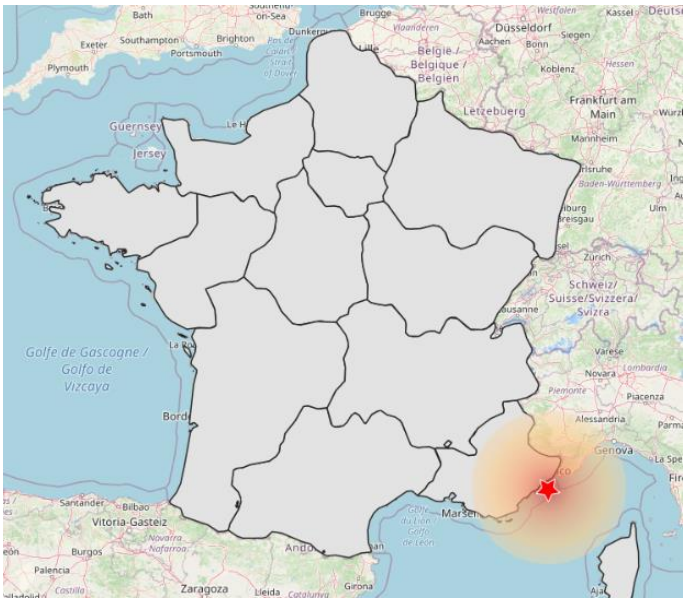
```
})
/* Déclaration de la couche WMS */
var couche_9 = new ol.layer.Tile({ source: sourceWMS_9 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_10 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom10', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_10 = new ol.layer.Tile({ source: sourceWMS_10 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_11 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom11', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_11 = new ol.layer.Tile({ source: sourceWMS_11 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_12 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom12', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_12 = new ol.layer.Tile({ source: sourceWMS_12 });

/* Déclaration de la source de la couche en format WMS */
var sourceWMS_13 = new ol.source.TileWMS({
    /* Chargement du lien WMS */
    url: 'http://localhost:8080/geoserver/Nice/wms',
    /* Chargement de l'espace de travail : couche */
    params: {'LAYERS': 'zoom13', 'TILED': false},
    serverType: 'geoserver'
})
/* Déclaration de la couche WMS */
var couche_13 = new ol.layer.Tile({ source: sourceWMS_13 });

// map.addLayer(SeismeTile);

map.addLayer(couche_7);

var zoomDiv = document.getElementById('zoom');
zoomDiv.innerHTML = 'Zoom : ' + defaultZoom;

map.on("moveend",  function () {
    zoomDiv.innerHTML = 'Zoom : ' + map.getView().getZoom().toFixed(2);
```

```
    if(mapView !=parseInt(map.getView().getZoom()) ){
        mapView =parseInt(map.getView().getZoom());

    if(parseFloat(map.getView().getZoom()) <= 8 &&
parseFloat(map.getView().getZoom()) > 7) {
        map.addLayer(couche_8);
        map.removeLayer(couche_7);
        map.removeLayer(couche_9);

    }else if(parseFloat(map.getView().getZoom()) <= 6) {
        map.addLayer(couche_6);
        map.removeLayer(couche_7);

    }else if(parseFloat(map.getView().getZoom()) <= 7 &&
parseFloat(map.getView().getZoom()) > 6) {
        map.addLayer(couche_7);
        map.removeLayer(couche_6);
        map.removeLayer(couche_8);

    }else if(parseFloat(map.getView().getZoom()) <= 9 &&
parseFloat(map.getView().getZoom()) > 8) {
        map.addLayer(couche_9);
        map.removeLayer(couche_10);
        map.removeLayer(couche_8);

    }else if(parseFloat(map.getView().getZoom()) <= 10 &&
parseFloat(map.getView().getZoom()) > 9) {
        map.addLayer(couche_10);
        map.removeLayer(couche_9);
        map.removeLayer(couche_11);

    }else if(parseFloat(map.getView().getZoom()) <= 11 &&
parseFloat(map.getView().getZoom()) > 10) {
        map.addLayer(couche_11);
        map.removeLayer(couche_12);
        map.removeLayer(couche_10);

    }else if(parseFloat(map.getView().getZoom()) <= 12 &&
parseFloat(map.getView().getZoom()) > 11) {
        map.addLayer(couche_12);
        map.removeLayer(couche_11);
        map.removeLayer(couche_13);

    }else if(parseFloat(map.getView().getZoom()) > 12) {
        map.addLayer(couche_13);
        map.removeLayer(couche_12);
        }
    }
})

var overlayGroup = new ol.layer.Group({
    title: 'Overlays',
    fold: true,
    layers: []
});

map.addLayer(overlayGroup);
```

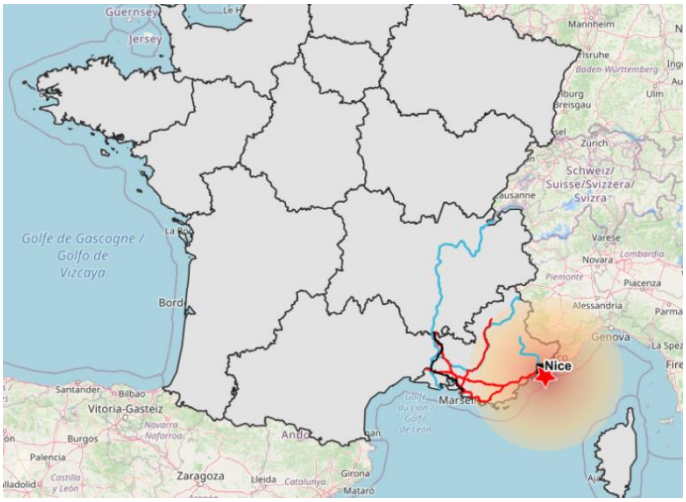
```
var layerSwitcher = new ol.control.LayerSwitcher({
     activationMode: 'click',
     startActive: false,
     groupSelectStyle: 'children'
});

map.addControl(layerSwitcher);

function toggleLayer(eve) {
    var lyrname = eve.target.value;
    var checkedStatus = eve.target.checked;
    var lyrList = map.getLayers();

    lyrList.forEach(function(element){
        if (lyrname == element.get('title')){
            element.setVisible(checkedStatus);
        }
    })
};

var mousePosition = new ol.control.MousePosition({
    className: 'mousePosition',
    projection: 'EPSG:4326',
    coordinateFormat: function(coordinate){return
ol.coordinate.format(coordinate,'{y} , {x}', 6);}
});

map.addControl(mousePosition);

var scaleControl = new ol.control.ScaleLine({
    bar: true,
    text: true,

});

map.addControl(scaleControl);

var zoomDiv = document.getElementById('zoom');
zoomDiv.innerHTML = 'Zoom : ' + defaultZoom;

map.on('moveend', function(e) {
    zoomDiv.innerHTML = 'Zoom : ' + mapView.getZoom().toFixed(2);
});
```

Résultat visuel :

> zoom5

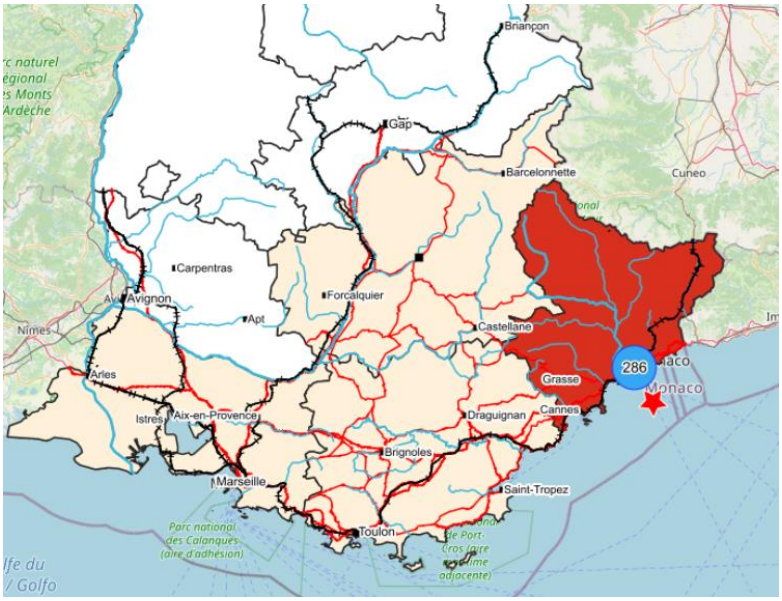



> zoom6




> zoom7

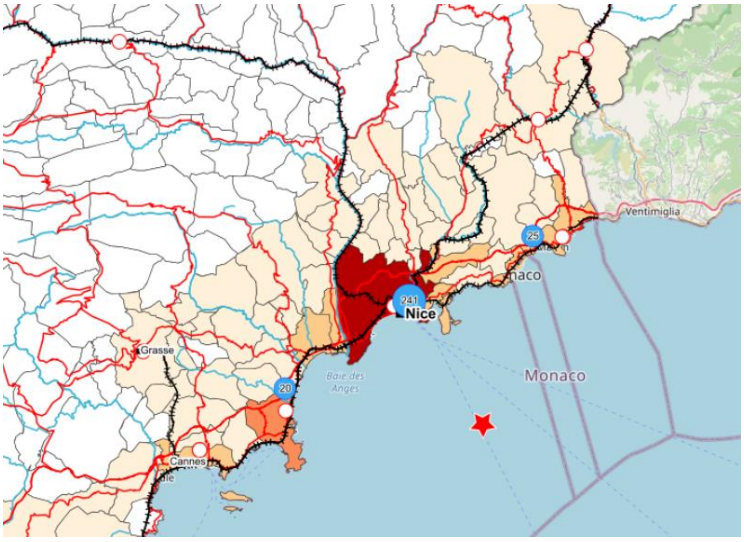
> zoom8


Briançon
Gap
Barcelonnette
Cuneo
Carpentras
Avignon
Forcalquier
Apt
Nîmes
Castellane
Arles
Grasse
286
Monaco
Istres
Aix-en-Provence
Draguignan
Cannes
Brignoles
Marseille
Saint-Tropez
Toulon
Parc national des Calanques (aire d'adhésion)

> zoom9

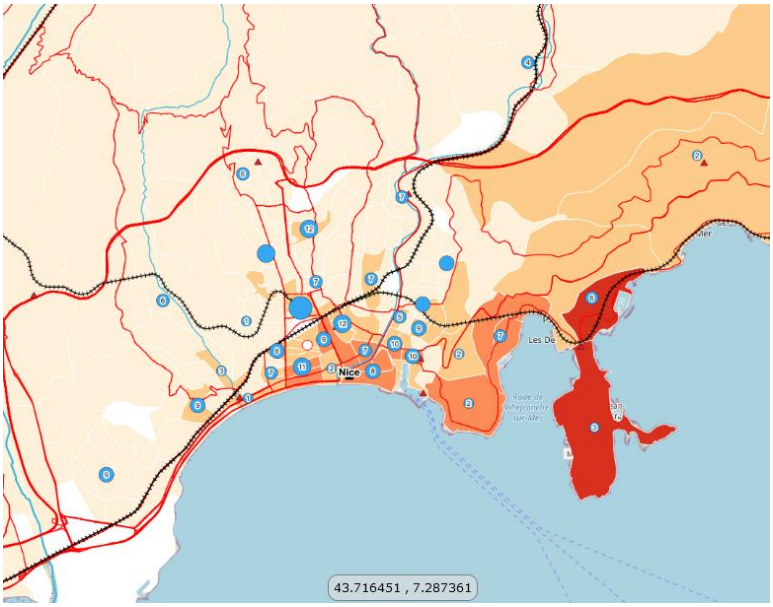

Parco Naturale delle Alpi Marittime
Castellane
Sanremo
Ventimiglia
25
241
Nice
Grasse
Monaco
20
Cannes
Draguignan
Saint-Tropez

> zoom10

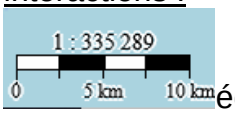



> zoom11


> zoom12



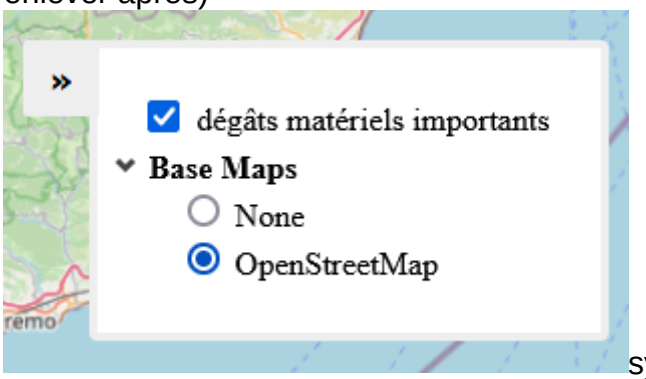
interactions :



échelle

coordonnées de la souris

niveau de zoom (que j'utilise pour mes recherches mais que je vais enlever après)



système de cochage et décochage

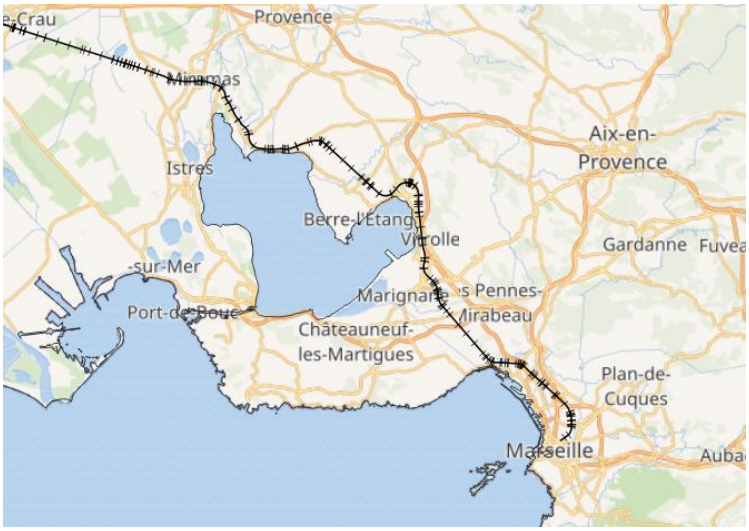
## B) Continuer la généralisation (simplification et déplacement)

outils que je peux utiliser :

- douglas peucker : outil de filtrage de ligne mais jamais tout seul dans le processus de généralisation
- outils GRASS : v.generalize

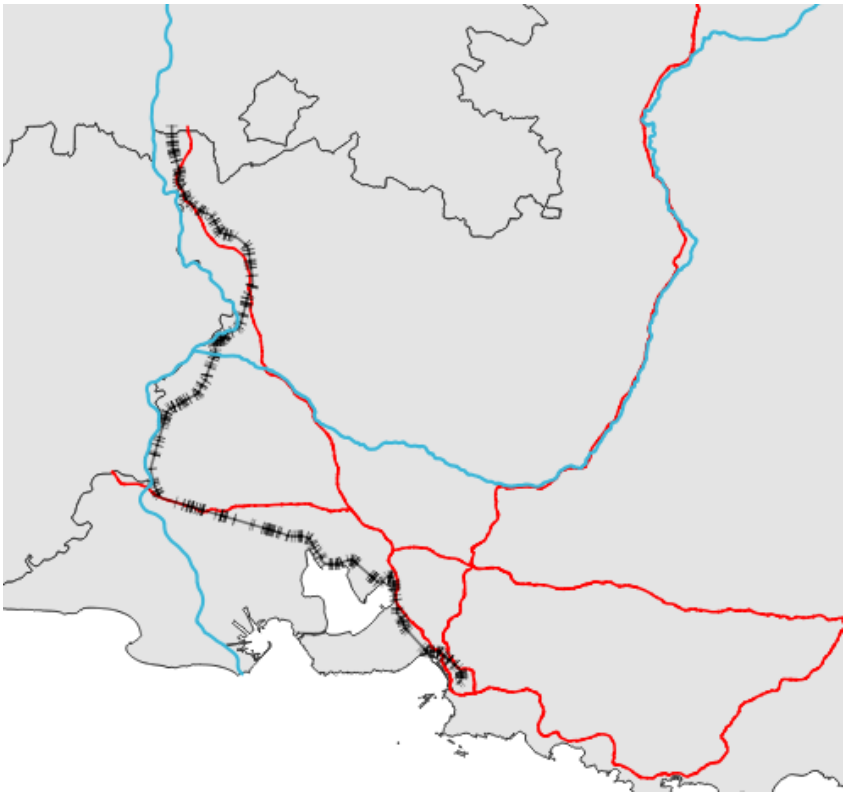
### a. Généralisation zoom 7

Malgré mes essais, j'ai l'impression que ces outils automatiques permettent de généraliser des entités à assez grande échelle. Or je dois pouvoir généraliser des entités à l'échelle de régions, ce qui m'oblige à procéder manuellement. L'avantage notable de cette méthode repose sur la possibilité de regrouper la simplification et le déplacement de l'entité. En effet, il est possible d'adapter la généralisation du trait en fonction des éléments spatiaux comme un étang ou une autre donnée superposée.




Dans le premier cas (voir ci-contre), il est très probable qu'un outil automatique sur QGIS simplifie la voie ferrée jusqu'à la faire passer au-dessus de l'étang. Manuellement, ce problème est facilement contournable. Dans le deuxième cas, il est possible de faire des déplacements en tenant compte des interactions avec les autres données. Par exemple, si l'on regarde

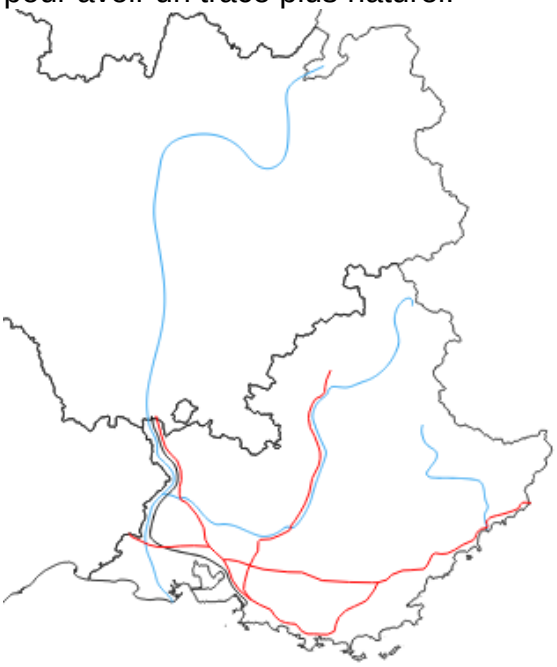
l’image dessous, on peut voir plusieurs superpositions avec au centre une superposition de routes et de courts d’eau et à l’ouest, une superposition générale entre les voies ferrées, les routes, les cours d’eau ainsi que les limites départementales. Ainsi, je dois d’abord avoir une idée précise de la hiérarchie des données et une image du placement de chaque entité pour éviter que les lignes ne se coupent.

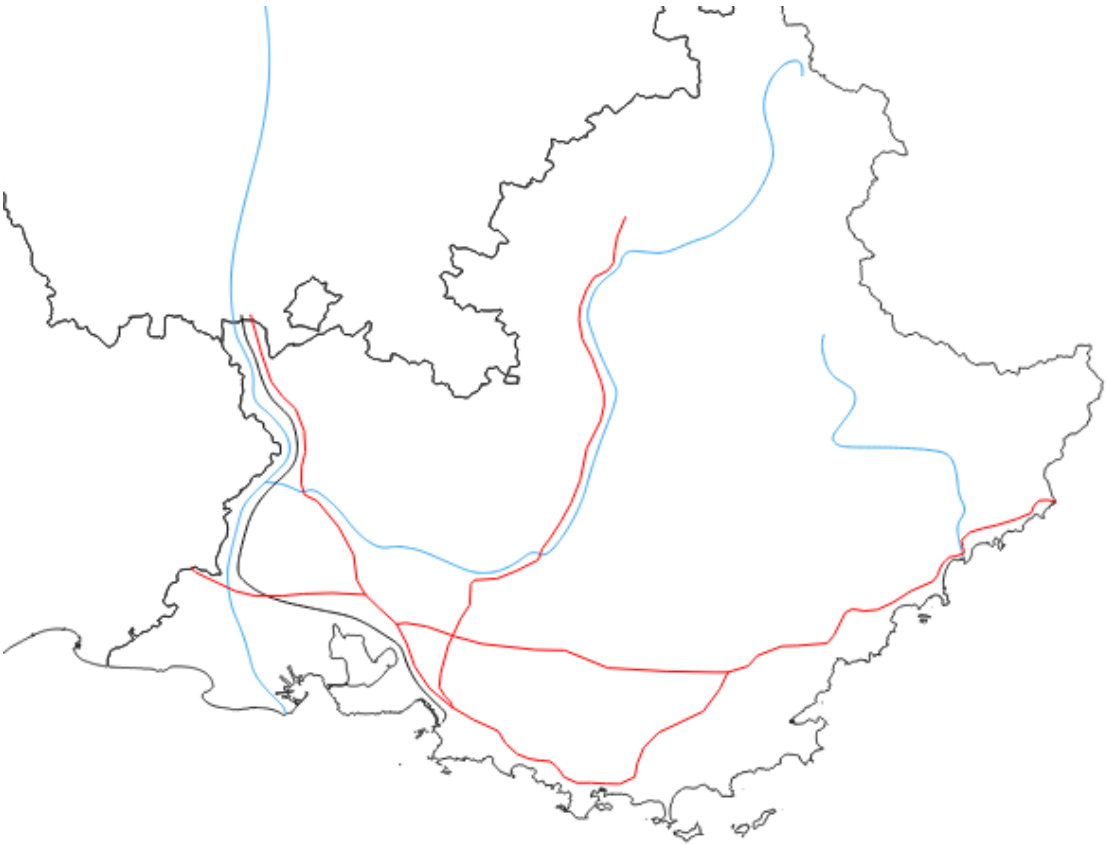



Sur cette partie de la carte, plusieurs superpositions apparaissent. En regardant d’où partent les lignes et où elles vont, des coupures sont inévitables. L’objectif est de les placer au bon endroit et de séparer les entités le reste du temps afin de les différencier à la bonne échelle. Je mets également en évidence les coupures car elles correspondent à des endroits stratégiques comme les ponts, lorsqu’une route croise un cours d’eau.

A cette échelle, j'ai décidé de garder une distance de 2000 à 3000 mètres entre les différentes données car c'est suffisant pour pouvoir les visualiser séparément.

Seulement ensuite, je peux commencer à tracer une ligne qui me semble juste en prenant en compte tous ces paramètres et je la lisse avec l'outil "lissage" de QGIS pour avoir un tracé plus naturel.



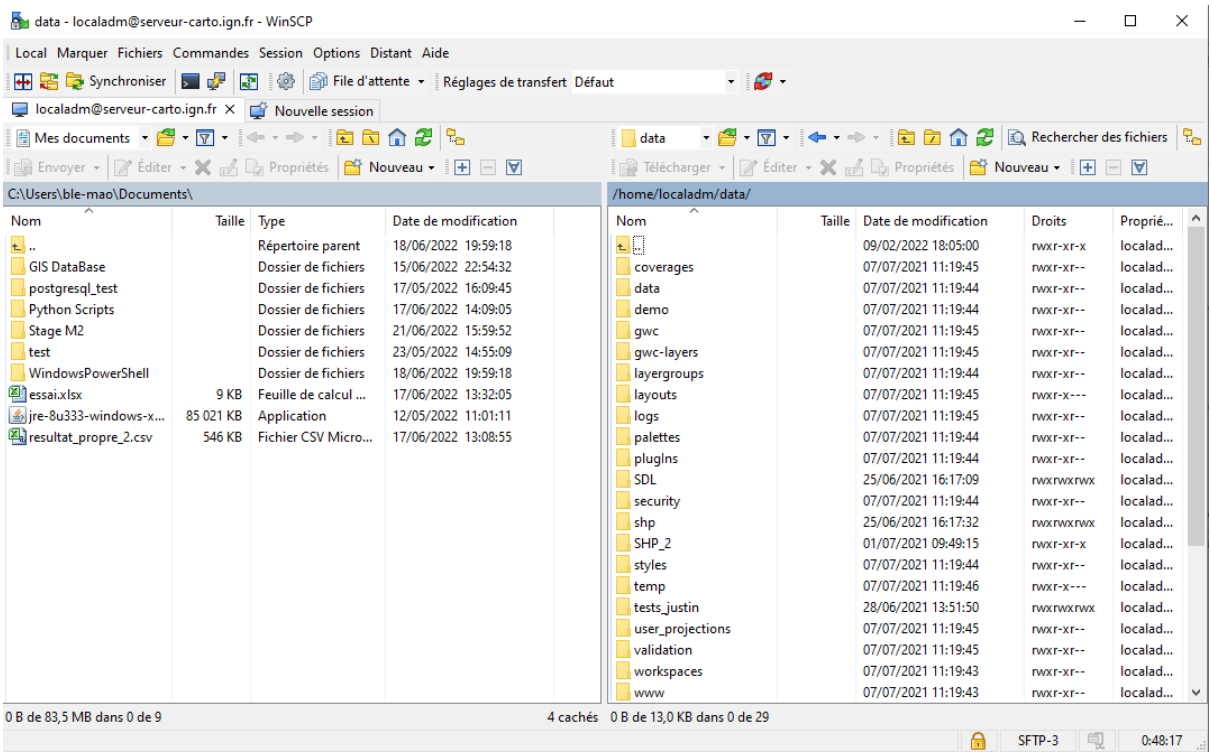


Ceci fait, je dois compléter la table attributaire, maintenant vide, des entités, pour pouvoir afficher le nom des cours d'eau par exemple.

C) Me préparer à l'atelier du 1er juillet

Trouver un geoserver fonctionnel

Face au problème constant du geoserver d'AWS, et donc à la possibilité de ne pas pouvoir présenter ma carte le jour de ma présentation face aux sapeurs-pompiers, je me dirige vers le geoserver de l'IGN. Je ne m'étais pas tournée vers celui-là au début, car le proxy empêche la visualisation des données hors du réseau IGN. Cependant, Laura (l'ingénieure qui m'accompagne lors de l'atelier) possède une machine avec un VPN capable de se connecter sur le réseau IGN depuis Gardanne, au sud d'Aix-en-Provence. Ainsi, je dois transférer toutes mes données sur le nouveau geoserver. N'ayant pas les accès pour me connecter à la base de données postgres de l'IGN, je décide d'importer mes couches directement sur le geoserver en me connectant à l'aide de WinSCP.




D) Passer mes premiers entretiens

J'ai passé mon premier entretien, d'une durée de 2h. J'ai suivi la méthodologie que j'avais mise en place au début de mon stage. L'entretien a été enregistré et j'ai pris des notes. La vidéo sera transcrite plus tard.

## E) Objectifs de la semaine qui suit :

1) Finir la simplification des données

- Zoom 8, 9, 10, 11, 12, 13

2) Implémenter les besoins des utilisateurs

- Dans le zoom 7, le reste de la France n'a plus de fond, et je ne sais pas si cette réduction soudaine de l'espace peut perturber les utilisateurs. Je vais donc garder le fond gris de la région du zoom précédent jusque dans ce zoom pour garder la continuité de la carte. De plus, je vais supprimer les trois départements au-dessus de la région car les valeurs de dégâts étaient trop faibles pour être gardées et leur présence sur la carte interroge, en plus de perturber les représentations classiques de la France (par régions, par départements…).
- Dans le zoom 9, les limites communales sont trop foncées et je n'arrive pas bien à voir les routes et cours d'eau. Je suis bien embêtée car je vois bien qu'il y a du clutter visuel, mais pourtant, c'est une donnée plus importante que les deux autres. J'avais essayé de mettre un tracé moins foncé pour qu'on puisse quand même voir les limites de communes tout en diminuant l'amas trop important d'informations mais ce type de symbologie ne semble pas être importé dans geoserver.
- Sur le zoom 11 et 12, certaines informations n'apparaissent pas comme le nombre de blessés au cœur de Nice (l'information la plus capitale). Sur geoserver, la couche n'a aucun problème donc le soucis vient d'Openlayers et de la représentation sur une carte. C'est peut-être dû à une trop grande quantité d'informations à cet endroit (comme un hôpital caché dessous). Je compte régler le problème en séparant les clusters des données agrégées et mettre le niveau le plus supérieur d'affichage avec la ligne de code : nomducluster_layer.setZIndex(1).
- dans le zoom9 encore, les hopitaux apparaissent, mais ne sont pas bien visibles lorsqu'ils sont placés en dehors des zones affectées par le séisme, car représentés en blanc avec un contour rouge. Pourtant, si cette représentation permet de bien situer les hôpitaux dans les zones à risque, les sapeurs-pompiers sont plus intéressés par les hôpitaux de campagne, soit ceux qui ont le moins de chance d'être touchés voire détruits par le séisme. Je dois donc trouver une symbologie qui permette même de mieux voir les hôpitaux de campagne. Je comptais importer la représentation commune des hôpitaux depuis QGIS mais le symbole ne semble pas s'afficher correctement. Dans tous les cas, je dois trouver autre chose car on pourrait les prendre comme des ronds-points si les hôpitaux sont placés non loin d'une route.
- Au zoom 9 toujours, certaines communes ne s'affichent pas comme au niveau de Marseille. Je vais vérifier si la jointure s'est bien faite. Sinon, je vais regarder s' il y a des dégâts sur la zone et s'il y en n'a pas, je complèterai les trous avec des polygones correspondants blancs.
- Au zoom 9, rajouter la limite des départements car il y a une discontinuité sur ce côté-là.

- J'ai certes commencé le processus de généralisation mais elle n'est pas finie. Je dois simplifier les données selon les échelles. De plus, certaines couches comme les cours d'eau et les routes se superposent donc je dois les déplacer aussi.

Les améliorations selon les besoins des utilisateurs (encore en cours de récolte)

*idées générales*
- outil facile d'utilisation → ce ne sont pas des professionnels
- outil complexe quand il le faut → complet mais pas trop complexe
- outil servant à la remontée d'informations
- carte utilisée sur ordi ou tablette (pas téléphone)

*idées pratiques*
- on doit pouvoir écrire dessus (à échelle très locale)
- possibilité de partager les informations → excel + imprimante
- besoin d'afficher la légende
- implémenter une boîte de dialogue pour faire afficher de l'information avec un clic (mettre superficie, nb de victimes)
- afficher nom des cours d'eau, des communes, dep…
- mettre la base de données d'enjeu comme données à cocher et décocher (barrage, pont, sites SEVESO, sites économiques, hub comme les aéroports)
- mieux sélectionner les casernes, notamment selon le nb de lits qu'ils peuvent accueillir (impacte le processus de sélection et à relier avec l'infobulle)
- mieux visualiser les limites communales (va de paire avec le pb de superposition des données)
- avoir la possibilité de pouvoir afficher chaque donnée séparées, proposées par le BRGM. C'est très important car dans mon travail, j'ai supposé qu'ils n'avaient besoin que des dégâts très importants et des blessés graves, ce qui m'avait fait regrouper plusieurs données, alors que ce n'est pas forcément ce qu'ils demandent. Chaque donnée à son importance !
- Les noms des villes et préfectures ainsi que les autoroutes sont de forts marqueurs spatiaux, ce qui exige peut-être de les démarquer et de mieux compléter la base de données.
- séparer les dégâts humains et matériels comme couches à cocher et décocher

ajouter des fonds de carte utiles : google earth, google map, street view, vue aérienne

## 3) Représentation des données

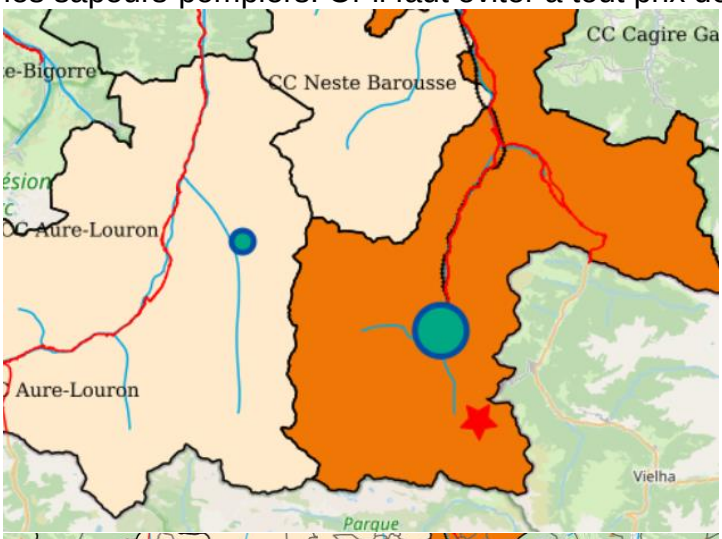
### a. Représenter les pertes humaines

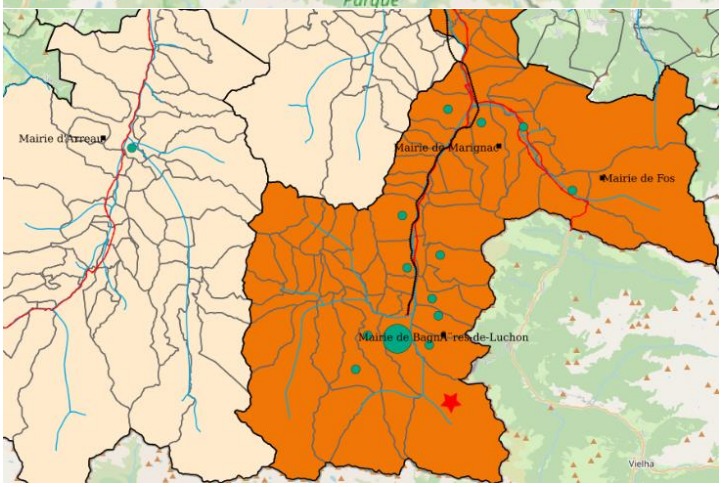
> plusieurs enjeux : représentation (infobulle) et position

- infobulle → code
- position → déplacements

renvoie au gestalt. Comparer avec la carte de thileli. La généralisation autorise à déplacer les entités, encore faut-il que ça soit fait intelligemment. si on déplace un cluster pour le mettre entre les deux entités qu'il comprend au zoom suivant, l'utilisateur comprend que la première entité regroupe les deux. Elle fait un lien logique. SI néanmoins une même entité à un zoom est déplacée pour diverses raisons (position trop proche des frontières…) et qu'au zoom suivant, la même entité est plus loin de l'emplacement de la première, le lien logique se fait moins facilement et on est plus apte à se poser tout un tas de questions comme : "est-ce la même entité ou la désagrégation d'un cluster ?" "Pourquoi a-t-elle été déplacée ?" Si c'est effectivement la désagrégation d'un cluster, le gestalt ne s'opère pas toujours bien et il est plus facile de faire un lien d'identité entre les deux entités au lieu d'un lien d'agglomération. Je pense que ce gestalt a un périmètre plus réduit et qui dépend aussi des autres entités. Il faut le prendre en compte en réfléchissant à une carte.

Par exemple, ci-dessous, je reprends la carte développée par Thileli, l'ancienne stagiaire, qui utilise le déplacement pour les deux cas. Le premier cas (ci-dessous), montre le déplacement d'un même point à l'ouest. Ce déplacement peut faire l'objet d'interrogations. Peut-être est-ce dû à un processus de clusterisation mal adapté ou le résultat d'un choix personnel. Dans tous les cas, un tel déplacement signifie dans mon cas "désagrégation", ce qui me pousse à regarder autour pour voir d'autres points. Ici, le lien d'identité ne s'est pas tout de suite créé pour ma part et je fais l'hypothèse que si c'est le cas pour moi, c'est possiblement aussi un problème pour les sapeurs-pompiers. Or il faut éviter à tout prix de leur faire perdre de leur temps.



C’est pourquoi dans ma méthodologie, je vais uniquement utiliser le principe de déplacement par agrégation car je considère que si un déplacement d’une même entité a dû être fait à plus grande échelle, c’est qu’il manque une échelle plus fine de clusters.

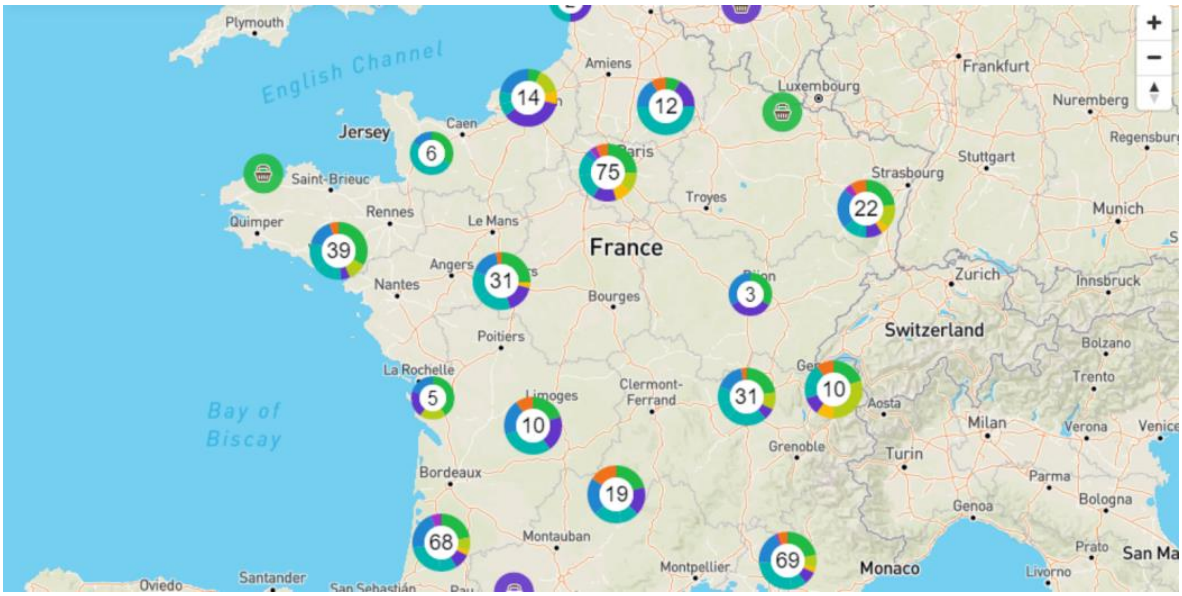
> Apparence des clusters

la forme :

Les sapeurs-pompiers proposaient d’utiliser des camemberts et par ce choix, cela traduit bien une volonté d’avoir un aperçu global de l’ensemble des enjeux. Pourtant, le camembert et partout critiqué par leur manque de visibilité. En effet, l’œil a du mal à traduire les angles en proportions, et donc la valeur de la catégorie. Cette difficulté à faire des comparaisons entre données s’intensifie au fur et à mesure que le nombre de valeurs affichées augmente, car la taille de chaque segment diminue. Il est dit d’éviter plus de 3 catégories par pie chart.

Pourtant, cette règle s’applique aux cartes papier, mais la carte numérique (multi-échelle qui plus est) peut s’affranchir de cette contrainte car la vision “générale” qu’offre le camembert suffit pour un premier aperçu. Pour avoir d'avantages d’informations, il ne suffit que de cliquer dessus pour voir s’ouvrir une fenêtre avec le détail des dégâts. Il faut bien garder à l’esprit qu’une carte utile aux pompiers n’est pas forcément une carte thématique avec la possibilité de regrouper des ensembles de dégâts. Ils travaillent par secteur, ce qui nous oblige à laisser les catégories, aussi nombreuses soient-elles, pour convenir aux différentes missions du sapeur-pompier.

Malgré tout, même dans une carte numérique, le camembert présente l’inconvénient de prendre plus de place que leurs alternatives, comme un graphique à barres par exemple. C’est pourquoi j’ai décidé de m’inspirer de plusieurs types de cartes comme celui ci-dessous car ce n’est plus l’angle qui aide à déterminer les valeurs mais les distances proches du périmètre. Cette représentation a également l’avantage de garder une valeur au centre, ce qui permet, en une échelle, d’avoir deux niveaux de détails sur les dégâts humains. Trois en comptant la fenêtre en pop-up.



les couleurs :

Une tranche de couleur plus foncée apparaît plus grande qu'une tranche de même taille d'une couleur plus claire donc je vais m'appuyer sur cette connaissance pour en faire un atout et non un inconvénient. Ne pouvant pas faire de hiérarchisation dans le processus de sélection de l'information à afficher, c'est un bon moyen pour apporter une première hiérarchisation en fonction de l'urgence des réponses à apporter. Certes, si les données sont affichées, c'est qu'elles sont considérées toutes comme urgentes mais certaines nécessitent une intervention plus rapide que d'autres. Ainsi, je reprends la classification effectuée par le BRGM (P1 à 4) et je mets un dégradé de couleurs afin que le foncé (donc ce qui se voit le mieux) corresponde aux dégâts les plus urgents.

C. Implémenter les besoins des utilisateurs

> hiérarchiser l'urgence au cas où je n'aie pas le temps de tout finir pour l'atelier du 1er juillet

### a. Développer le système d'infobulle

> adapter la formule :

```
const name = feature.get('name');

const info = $('#info');
info.tooltip({
  animation: false,
  trigger: 'manual',
});

const displayFeatureInfo = function (pixel) {
  info.css({
    left: pixel[0] + 'px',
    top: pixel[1] - 15 + 'px',
  });
  const feature = map.forEachFeatureAtPixel(pixel, function (feature) {
    return feature;
  });
  if (feature) {
    info.attr('data-original-title', feature.get('name')).tooltip('show');
  } else {
    info.tooltip('hide');
  }
};

map.on('pointermove', function (evt) {
  if (evt.dragging) {
    info.tooltip('hide');
```

```
    return;
  }
  displayFeatureInfo(map.getEventPixel(evt.originalEvent));
});

map.on('click', function (evt) {
  displayFeatureInfo(evt.pixel);
});
```

ou

```
const element = document.getElementById('popup');

const popup = new Overlay({
  element: element,
  positioning: 'bottom-center',
  stopEvent: false,
  offset: [0, -10],
});
map.addOverlay(popup);

const info = document.getElementById('info');
map.on('moveend', function () {
  const view = map.getView();
  const center = view.getCenter();
  info.innerHTML = formatCoordinate(center);
});

map.on('click', function (event) {
  $(element).popover('dispose');

map.on('click', function (event) {
  $(element).popover('dispose');

  const feature = map.getFeaturesAtPixel(event.pixel)[0];
  if (feature) {
    const coordinate = feature.getGeometry().getCoordinates();
    popup.setPosition([
      coordinate[0] + Math.round(event.coordinate[0] / 360) * 360,
      coordinate[1],
    ]);
    $(element).popover({
      container: element.parentElement,
      html: true,
      sanitize: false,
      content: formatCoordinate(coordinate),
      placement: 'top',
    });
    $(element).popover('show');
  }
```

```
});

map.on('pointermove', function (event) {
  if (map.hasFeatureAtPixel(event.pixel)) {
    map.getViewport().style.cursor = 'pointer';
  } else {
    map.getViewport().style.cursor = 'inherit';
  }
});
```

**b. Afficher un symbole personnalisé des hôpitaux (depuis Openlayers)**

> adapter la formule :

```
import Polygon from 'ol/geom/Polygon';

const symbol = [
  [0, 0],
  [4, 2],
  [6, 0],
  [10, 5],
  [6, 3],
  [4, 5],
  [0, 0],
];

const vector = new VectorLayer({
  source: new VectorSource({
    url: 'data/kml/2012_Earthquakes_Mag5.kml',
    format: new KML({
      extractStyles: false,
    }),
  }),
  style: styleFunction,
});
```

**c. Exporter la carte en pdf**

```
import WKT from 'ol/format/WKT';

const format = new WKT();

const dims = {
  a0: [1189, 841],
  a1: [841, 594],
  a2: [594, 420],
  a3: [420, 297],
  a4: [297, 210],
  a5: [210, 148],
```

```
};

const exportButton = document.getElementById('export-pdf');

exportButton.addEventListener(
  'click',
  function () {
    exportButton.disabled = true;
    document.body.style.cursor = 'progress';

    const format = document.getElementById('format').value;
    const resolution = document.getElementById('resolution').value;
    const dim = dims[format];
    const width = Math.round((dim[0] * resolution) / 25.4);
    const height = Math.round((dim[1] * resolution) / 25.4);
    const size = map.getSize();
    const viewResolution = map.getView().getResolution();

mapContext.globalAlpha = 1;
      const pdf = new jspdf.jsPDF('landscape', undefined, format);
      pdf.addImage(
        mapCanvas.toDataURL('image/jpeg'),
        'JPEG',
        0,
        0,
        dim[0],
        dim[1]
      );
      pdf.save('map.pdf');
      // Reset original map size
      map.setSize(size);
      map.getView().setResolution(viewResolution);
      exportButton.disabled = false;
      document.body.style.cursor = 'auto';
    });

    // Set print size
    const printSize = [width, height];
    map.setSize(printSize);
    const scaling = Math.min(width / size[0], height / size[1]);
    map.getView().setResolution(viewResolution / scaling);
  },
  false
);
```