Drift-X WPS: Pesticide atmospheric dispersion Web GIS

Nicolas BOZON[1], Gerald FENOY[2], Venkatesh RAGAHAVAN[3], Bijan MOHAMMADI[4],

[1]3LIZ sarl – Minea Cemagref Montpellier, France, nbozon@3liz.com
[2]GeoLabs sarl – Futur Building Lattes, France, gerald.fenoy@gelobas.fr
[3]Media Center, Osaka City University , Osaka, Japan, raghavan@media.osaka-cu.ac.jp
[4]CERFACS , Toulouse, France, bijan.mohammadi@cerfacs.fr

Abstract

Atmospheric pollution due to agricultural pesticide for viticulture is a major concern today, regarding both public health, sustainable agriculture and ecosystems quality monitoring. Atmospheric dispersion modeling and the use of geographic information systems allow us to spatially quantify the atmospheric pollution on a given area and to proceed to risk analysis. The simulation of air pollution can also be very helpful to determine optimum wether conditions for spraying over a specific terrain. This paper aims to present a multi-disciplinary research based on the use of an atmospheric dispersion model within a Web Processing Service (WPS) architecture, able to predict and map atmospheric pollution dispersion online. Past research and the coupling of the Drift-X model and Open Source GIS are first explained. The linking of Drift-X with ZOO Kernel 1.0 as a Fortran based Web Service is then detailed, with an emphasis on the used multi-language procedure and WPS chaining. The developed platform is then presented with details on the link to MapSerer and the adopted rendering techniques. The WPS Web GIS client is finally illustrated and and some of the user interactions detailed.

1. Introduction

Modern agricultural practices are based on the massive use of phytopharmaceutical products to control crops quality and quantity. This is particularly true about viticulture, which is one of the most pesticide consuming culture after cereals. Advances in vineyards protection and new chemical solutions have contributed to increasing yields and to ensuring regular quality production to agricultural exploitations. Chemical control products have proved to be extremely efficient and allow the pesticide penetration within canopies, but their systematic use impacts soils, water and air. In this paper, we will focus on atmospheric dispersion and the induced deposition of pesticide on the ground.

Atmospheric dispersion of pesticide is a complex process but can be basically explained as follows (Bozon et al, 2009). Phytopharmaceutical products are usually spread over the plots using a sprayer, which is an agricultural machine allowing to spray pesticide over large areas. It is most of the time towed or suspended from a tractor, as shown in figure 1.

Figure 1. Example of pesticide sprayer used in vineyards

Sprayed pesticides then spread in several directions while the sprayer runs through the vineyards. Some of it reaches the vine leaves and grapes, some reaches the ground and run-o on soils and the rest leaves the plot to be transported by the wind. This forms a pesticide cloud, as shown by figure 2, that is prone to dispersion and depositing in the environment (Bozon et al, 2009).

Figure 2. Observed pesticide cloud after early morning treatment

Many agro-meteorological studies are focusing on the spray drift process at different scales, and on its impacts on the environment. Near-field studies tend to become very accurate using Computational Fluid Dynamics (CFD) principles, and allow scientists to characterize pesticide emissions to the air during and after the treatments. This is a major asset to model the long-range transport of pesticide, as the results of validated micro-scaled models can be used as input data in the different atmospheric dispersion models available. In our former research, a long-range transport model was developed with taking some aspects of the GIS data model into account and by using open source tools (Bozon et al, 2007).

2. Coupling atmospheric dispersion modeling and open source GIS

2.1 Atmospheric dispersion modeling

Atmospheric dispersion modeling (ADM) is an essential tool in air quality management because it provides a relationship between source terms locations (i.e where discharges to the air occur) and observed adverse effects on the environment and the neighborhood.

Atmospheric dispersion models refers to the mathematical simulation of air pollutants dispersion in the ambient atmosphere. They are intimately related to numerical simulations as most models are performed with computer programs that solve the mathematical equations and algorithms which simulate the pollutant dispersion. As atmospheric dispersion is complex and because air pollution cannot be measured in every place it occurs, models are used to simplify and simulate the dispersion of air pollutants from emission sources, and to to predict the downwind concentrations or depositions on a given area.

Despite the fact that many models prove to be efficient on small domains (i.e a few square meters), only a few are adapted and validated for larger areas (i.e a few square kilometers) and simulations of atmospheric spray drift are seldom performed at the watershed scale. The decision of building a GIS-based dispersion model was thus taken, focussing on cartographic projections, scales changes and DEM layers in the original model 's code (Bozon et al, 2009).

2.2 The Drift-X model

Accurate wind data-sets are difficult to acquire over large areas and long time-series but required by most CFD models. Some of the eolian processes included in pesticide atmospheric dispersion are also to complex to be solved by numerical simulations at that stage, due to their uncertainty and variability. Furthermore, CFD tools appear to be quite long to use for simulations, as both input data and domain geometry have to be pre-processed through the use of several softwares. Their use necessitate quite expensive hardware and software and most of all involves very long calculation costs.

Given that wind data are quite poor for our concern and that a large set of assumptions has to be done to model the whole dispersion process at different spatial and temporal scales, the use of CFD tools to solve our problem was ill-advised. In relation to the necessary accuracy and because calculation costs have to stay low in order to perform fast simulations, an original alternative to the use of CFD is therefore proposed (Bozon et al, 2008).

Drift-X model is a probabilistic simplified Gaussian atmospheric dispersion model able to forecast pesticide spray drift after the treatment, from the plot to the watershed scales. The model operates within a domain of a several square kilometers, corresponding to a typical southern French small wine-growing area. Drift-X is based on a reduced-order modeling approach to flow field reconstruction with a small number of measurements, as well as to Gaussian plume transport over realistic topographies

and unsteady wind flows (Bozon et al, 2009).

The main goal of Drift-X is to provide the mean trajectory of a pesticide cloud after spraying applications, by forecasting the wind field and the pesticide concentrations for a permanent state. The wind flow is calculated according to the parameters and the DEM layer provided by the user. A user-defined quantity of pesticide is then transported and deposited according to the wind flow. Here is an example Drift-X simulation based on the following parameters and displayed as common vector and raster formats in figure 3 and 4.

- The domain for calculation is 8km2.
- The cartographic projection is extended Lambert 2 (EPSG:27572)
- The number of points for the output grid is 900.
- The used input DEM layer is SRTM 90m resolution.
- The source plot is 1 ha with 33 rows to treat.
- The sprayer treats 3 rows at the same time at the average speed of 1 m/s.
- The spraying nozzle output velocity is 7 m/s with an output ow of 0.001 kg/s.
- Two wind points are used to calculate the flow field (N 60 - 5 m/s wind , N 30 -4 m/s wind.)

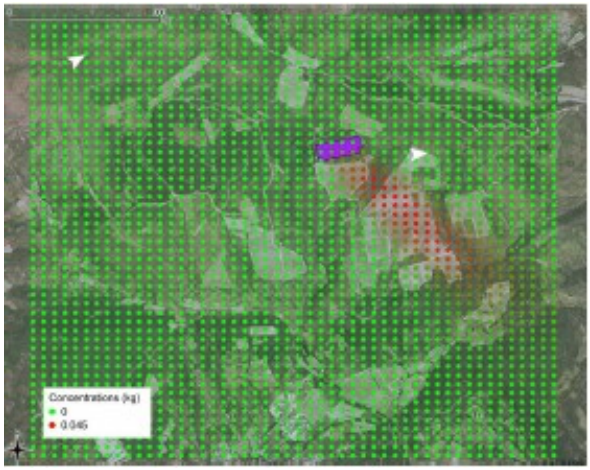

Figure 3. Resulting pesticide cloud rendered as ESRI shapefile (.shp)

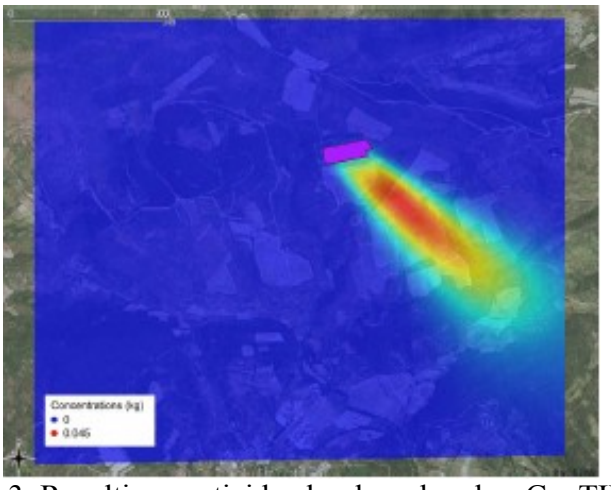

Figure 3. Resulting pesticide cloud rendered as GeoTIFF (.tiff)

2.3 Integration of Drift-X model in Quantum GIS

The reduced order modeling approach used for the modeling made the programing aspects easier. The equations composing the model have been transcribed in Fortran language, including a local spray drift model, the wind flow calculation aand a travel-time based transport model as routines. The choice of Fortran was made because it is one of the languages that is best suited to compute complex mathematical expressions ((Bozon et al, 2009).

The fastness of the Fortran compiler and the mesh free approach allows us to compute the solution in only a few seconds depending on the size of the domain and on the elevation data resolution. Indeed, the DEM values are extracted for the domain and then sent to Drift-X, which computes the solution using the x, y, z triplets as base topography. The results are then written to output files which contain point-based information for the whole domain.

The program has not been transformed into a independent GIS class yet as the integrated approach would suggest, but is used as a standalone and fast executable program. Both input and output datasets are communicating with Quantum GIS. Despite the fact that a more integrated GIS oriented ADM class will greatly enhance the coupling, there is a major advantage in the resulting coupling which is that the Fortran program stay independent of the GIS software, which will make any modification in the model easier as we will only need to re-compile the Fortran program.

Using Quantum GIS which is based on a robust C++ API that presents plenty of spatial algorithms and native GIS functions, we could easily integrate the Fortran executable (Bozon et al, 2008). QGIS has been designed according to an extensible plugin architecture. This allows new features and user-oriented functions to be easily added to the application and that's why QGIS offers advanced programming possibilities. Plugins can be created using C++ or the related Python bindings, which allow a simpler programming environment for developing specic plugins that directly interact with the C++ source code. The Drift-X plugin was thus developed using the QGIS Python bindings, (Bozon et al, 2008). and its interface is presented in figure 5.

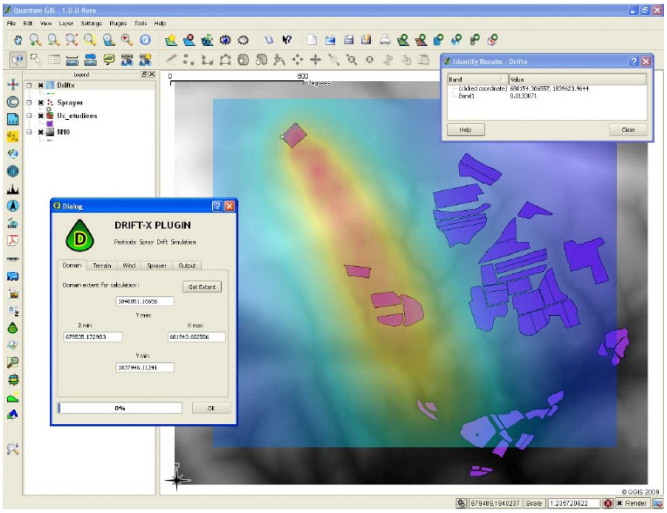

Figure 5. Drift-X plugin interface within the QGIS interface.

3. Adapting Drift-X to WPS by using ZOO 1.0

3.1 Research context and goals

Once the Drift-X model was integrated into QGIS, we could use it directly into a geo-referenced framework and run many simulations simply, according to various wind parameters and DEM resolution layers. Other analysis tools provided by QGIS could also been used directly to work on the resulting pesticide clouds layers, and to intersect them with other relevant geodata to tend to risk assessment (i.e intersecting the simulated pesticide cloud with population density layers for example).

Although it gave satisfying results, the scientists and the drift-X users soon needed a more generic and user-friendly platform, as it was intended to be used also by non GIS experts. Indeed, the use of the QGIS plugin requires the basic knowledge of the software, and the plugin is also subject to instability if some major changes occur in the QGIS API. The decision to adapt Drift-X to the Web was thus taken, as it appeared to be the simplest way to provide a stable, cross-platform and easy-to-use simulation platform.

As the development work done into QGIS was based on the chaining of various processing tasks, the idea of using WPS to reproduce the procedure on the server-side was taken rather early (Fenoy et al, 2009). This was also the best way to proceed to such processing in a OGC standardized way. The use of ZOO1.0 for adapting Drift-X to WPS is presented in the next section, and the whole WPS chaining is then detailed.

3.2 Linking Drift-X with ZOO Kernel 1.0

ZOO Kernel is the heart of ZOO 1.0. It is a server-side C Kernel which makes it possible to create, manage and chain WPS 1.0.0 compliant Web Services, by loading dynamic libraries and handling them on-demand. Thus, it can easily connect to geospatial libraries and scientific models, but also with the common cartographic engines and spatial databases.

ZOO Kernel is written in C language, but Web Services can be programmed in C, Python, Java, Fortran, PHP and JavaScript. This multi-language support is convenient for developers and allows above all to use existing code to create new Web Services. Open source GIS libraries or specific code can so be ported server-side with very little modifications (Fenoy et al, 2009).

Using ZOO was thus very convenient for our concern, as it supports Fortran code, but also includes server-side adaptation of the GDAL and OGR libraries by default, so it was helpful for us as we also needed some GDAL commands in the chaining. ZOO 1.0 was thus installed on a Linux server with the Fortran support activated. This implied to install the Fortran compiler (G77) on the machine, before starting to create the needed Web Services. MapServer 5.6 was also installed on the server, as it is used as the default cartographic renderer in the intended Web GIS. Its use is explained in the next section.

3.3 ZOO Services developed for Drift-X

Once the server configuration was done, we could start by defining our chaining and develop the different targeted Web Services step-by-step. For each new Service, a ZOO configuration file (.zcfg) was added in the ZOO installation. The different steps are detailed bellow.

The first needed operation is to get the input parameters defined by the users, in order to generate the Drift-X configuration file needed for the server-side computation. A simple OpenLayers map and a set of input forms were set up, allowing the user to draw a calculation extent, create wind points, selecting a vineyards plot and fill other vineyards specific parameters. Once those values available, they are sent to a simple WPS service written in Python, that writes an ASCII file containing the well formated input data needed by Drift-X. An extract of the Python code is presented bellow:

```
def driftX_writeParams(xmin,max,ymin,ymax,windx,windy,windspeed...)
    f=open('driftx.data', 'w')
    f.write(xmin+xmax+ymin+ymax)
    f.write(windx+windy)
    f.write(windspeed)
    …
    f.close()
```

The second step is then to add a DEM layer in the interface and to clip it according to the user-selected extent. An extract ot the SRTM 90m DEM is thus displayed in the platform using a WMS configured MapServer installation. Then, the extent written by the first Service is passed to the available ZOO GDAL Service, so the SRTM can be dynamically clipped server-side using the following simple GDAL Translate command.

```
gdal_translate -ot Float32 -projwin xmin ymax xmax ymin
                input_dem .tif output_dem .tif
```

These two first step thus provide the Drift-X configuration file and the input DEM layer, so that everything is ready to launch the Drift-X Fortran code. The next step is then to create a ZOO Service using the Drift-X Fortran source code. This is done using the ZOO Fortran support, which converts Fortran code into dynamic C library. Only a few modifications in the original code were needed to make it work, mainly to adapt it to the ZOO Kernel data structure. After several test, we concluded that the outputs were exactly the same as the local Drift-X, and that the calculation costs were not that much affected, depending on the client bandwidth of course (Fenoy et al, 2009).

After a few seconds, the ZOO Drift-X Service thus generates an output .CSV file containing the x,y,z triplets from the DEM layer and the corresponding deposited pesticide concentrations calculated

by Drift-X. The ZOO GDAL Service is called once again to be able to convert this file into an new GeoTIFF, using the following GDAL Grid command.

```
gdal_grid -of GTiff -ot Float64 -l driftx driftx.vrt output.tif
```

Some other GDAL options or commands can be used for advanced rendering of the raster pesticide cloud, such as different kind of spatial interpolations. In our case, the Inverse Distance Weighted (IDW) interpolation command was called like this:

```
gdal_grid -a invdist : power =1.0: smoothing =50.0
          -txe xmin xmax -tye ymin ymax
-of GTiff -ot Float64 -l driftx driftx.vrt output.tif
```

The last step is to create a ZOO Service able to get the resulting tiff and to write the corresponding mapfile, so that MapServer can dynamically display the resulting raster file as WMS, each time that a simulation is achieved. This was done using Python and a simplified extract is given bellow:

```
def driftX_writeMapFile(xmin,max,ymin,ymax,...)
   f=open('driftx.map', 'w')
   f.write("MAP")
   f.write("NAME" +""+ "Driftx map")
   f.write("SIZE" +""+ size)
   f.write("EXTENT" +""+ xmin + ymin + xmax +ymax)
   …
   f.close()
```

Five inter-dependent WPS compliant Services were thus developed using different languages, namely Python for input configuration file writing, C for clipping and converting the input DEM layer, Fortran to execute Drift-X model, C again for converting and processing the Drift-X results, and finally Python for writing MapServer mapfiles on the fly.

Every Service are executed by ZOO Kernel thanks to their .zcfg files that provide the five Services metadata, in other words some information on their respective Provider, Input and Output. In order to add logic in the chaining and to be able to automate it, the ZOO API was also used at the end of the development steps. Indeed, ZOO 1.0 comes with ZOO API which is a Mozilla-based server-side Javascript API that can simplify the chaining and make it more generic. The chaining is thus driven by a simple JavaScript file that orders and chains *ZOO.Process* and *ZOO.Request* functions to call the five Services that were developed.

4. Building an OGC compliant simulation Web GIS

The last step of our research is to make the working WPS chaining communicate with a webmapping client application. This is the simulation platform front-end that is also developed by using the available open source tools.

An OpenLayers map is first set up to a suitable extent for Drift-X, and the input SRTM DEM Layer displayed as WMS by MapServer. For the user needs, a Corine Land Cover vineyards layer was also added as WMS, thus allowing to identify the suitable areas on which running simulations.

Then, OpenLayers is linked to a Jquery toolbar widget in order to provide advanced navigation to the user. The default OpenLayers navigation controls appear in the toolbar, and some Drift-X specific ones are also developed on top of the OpenLayers API. For example, an extent selection box control is created from the native OpenLayers Zoom Selection Box control, and allows the user to define the extent on which to simulate. Some other specific controls are developed for wind points and starting point creation, by using and adapting the OpenLayers vector editing functions.

A set of Jquery forms which are linked to some of the specific controls are then placed in the interface. Longitude and latitude values of extent, wind points and stating point are automatically filled by the controls. Wind directions and speeds values are driven by user-friendly sliders. Some other parameters can be provided manually by the user. A button is then added to the forms in order to send the user-defined values to the first ZOO Service and to launch the chaining defined in the previous section.

Finally, as soon as the chaining is finished (i.e when every ZOO service returned a “Service Succeeded” statement, the resulting interpolated raster pesticide cloud is served by MapServer as a WMS layer and added to the OpenLayers map. A screenshot of the Drift-X WPS user interface is shown bellow:

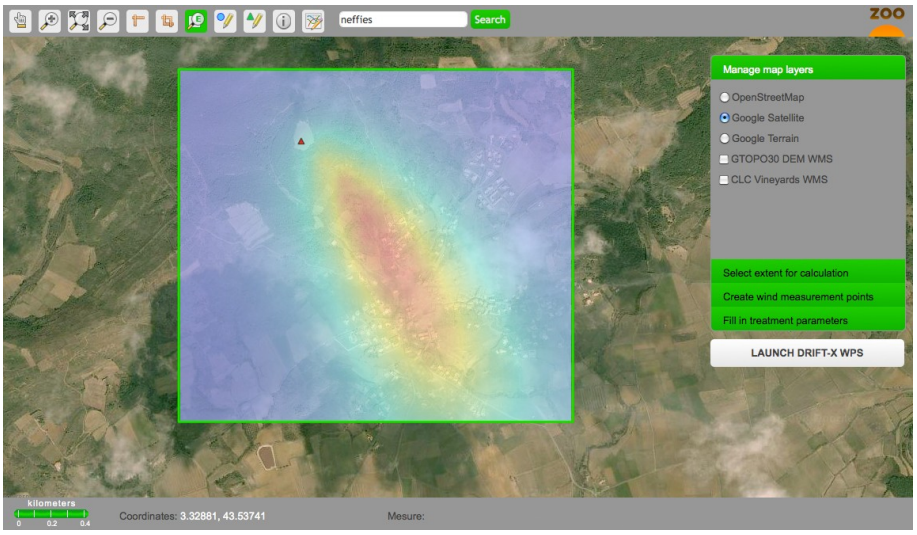

Figure 5. Drift-X WPS interface with an example WMS pesticide cloud.

5. Conclusion

A GIS based atmospheric dispersion model was presented and its integration in Quantum GIS was first explained. The same procedure development procedure was proposed and adapted in order to use Drift-X model within a WPS architecture based on ZOO 1.0.

A multi-languages WPS chaining was set up to run the different processes composing the model and according to the user-defined simulation parameters. The use of the original Drift-X code was possible thanks to the ZOO Fortran support, and this proves that other scientific models can be easily integrated as ZOO Services. Furthermore, the whole chaining is driven by JavaSrript thanks to the ZOO API.

Finally, a Web GIS client application was developed by using open source tools and let the user define the parameters, launch the simulation and visualize the resulting pesticide cloud layer. Both server-side and client-side are thus OGC compliant thanks to the use of WPS for processing and WMS for publishing the input/output.

References

Bozon N., 2009, *Coupling Atmospheric dispersion modeling and GIS: application to pesticide spray drift*, PhD thesis, University of Montpellier 2.

Bozon, N., Mohammadi, B. and Sinfort, C, 2007. '*Similitude and non symmetric geometry for dispersion modelling*'. Proceeding of STIC and Environment 2007. 5th edition. e-sta Vol.5, number 2,.

Bozon, N. and Mohammadi B., 2008. "Geo-Grenelle Award". Institut Géographique National. Geo-Evenement 20th edition.

Bozon, N. and Mohammadi, B., 2009 , '*GIS based atmopsheric dispersion modelling Forecasting the pesticide atmospheric spray drift from a vineyard plot to a watershed*'. Applied Geomatics, Springer, Vol.2.

Bozon, N., Mohammadi, B. and Sinfort, C., 2010. 'A GIS-based atmospheric dispersion model '. Proceeding of STIC and Environment 2009. JESA – Hermès.

Fenoy G., Bozon N. and Raghavan V., '*ZOO Project: The Open WPS Platform*'. Free and Open Source Software for Geospatial, FOSS4G 2010 – Barcelona, Spain, http://2010.foss4g.org/presentations_show.php?id=3621

Fenoy G., Bozon N. and Raghavan V., '*ZOO Project: The Open WPS Platform*'. OSGIS 2010 Conference – Nottingham, United Kindom, http://cgs.nottingham.ac.uk/~osgis10/os_call2010.html

Fenoy G., Bozon N. and Raghavan V., '*ZOO Project: The Open WPS Platform'*. WPS and Scientific computing for climate change Workshop – Trento, Italy, http://mpba.fbk.eu/sites/mpba.fbk.eu/files/02_ZOO-Project-WPSDay_2010.pdf

Fenoy G., Bozon N. and Raghavan V., 'ZOO Project: The Open WPS Platform'. Free and Open Source Software for Geospatial, FOSS4G 2009 – http://2009.foss4g.org/presentations/#presentation_93

Bozon N. and Mohammadi B., "GIS-based atmospheric dispersion modelling ". Free and Open Source Software for Geospatial, FOSS4G 2008 - http://conference.osgeo.org/index.php/foss4g/2008/paper/view/312/104