

PORTEAU: AN OBJECT-ORIENTED PROGRAMMING HYDRAULIC TOOLKIT FOR WATER DISTRIBUTION SYSTEMS ANALYSIS

Olivier Piller, Denis Gilbert, Karim Haddane and Sandrine Sabatie

Cemagref, UR REBX, Cestas, France
olivier.piller@cemagref.fr

Abstract

Several computer tools exist for Water Distribution Systems Analysis. The most well known of which – Epanet – will not be maintained in the near future. To remedy this, open source development projects have recently been proposed. Cemagref have developed the Porteau software, with several tools. They have decided to make their software open and freely available. In this paper, we present our experience to design a hydraulic toolkit for Water Distribution Analysis which could benefit the community in the context of an open-source project.

Firstly, technology choices are explained and highlighted in the current context. Some key elements were the objectives of a multiplatform product, the graph library availability and computation performance. Then, the main results in terms of packages and classes are presented. Graph elements consist of nodes, links and hydraulic devices. A collection of hydraulic devices/equipments may be placed at different locations on a pipe. Different classes with distinct attributes and methods have been created for the nodal and link demands as well as for the local and background leakages. Finally, the calculation targets are described with an example of extension ease with pressure-driven modelling and a reliability measure.

Keywords

Object-oriented programming, computer-aided software engineering tool, open-source software, hydraulic tool

1. INTRODUCTION

Several computer tools exist for Water Distribution Systems Analysis. One of them, Epanet [1], is widely used by the academic community because its C code is freely available and other programs can call it by means of a DLL library. Nevertheless, in the short term, Epanet will not be maintained but replaced by an open-source project [2]. This entails redesigning and rewriting the code in order to be easily understood and modified and to be conducive to collaborative development [3]. The existing Epanet tool has been written in a sequential language that does not facilitate making modifications and adding new components.

By contrast, Object-Oriented Programming (OOP) is specially designed to allow software evolution and extensions throughout life cycles. In recent years, some OOP hydraulic toolkits were diffused that implement an object-oriented layer over the Epanet solver (e.g. OOTEN [4]). More recently, an additional step was made that consists of designing a hydraulic solver in C++: In the CWSNet toolkit [5], two other layers are added that have benefits for equation assembly and implementation. Several scales of customisation are then possible. The project is only at its start and the GUI and all the Epanet calculation modules are not completely implemented.

With the recent progress in Computer Science, the application of Computer-Aided Software Engineering (CASE) results in high-quality and maintainable software products with a good independence from the targeted coding language. Objecteering (now Modelio) is a CASE tool that has been used for seven years at Cemagref to design their own software PortEau [6]. OOP and Unified Modelling Language (UML) are used to facilitate future developments and to certify the software use and the outputs. The coding is in Java for the Graphical User Interface (GUI) and in C++ for the calculation engines. This year the PortEau software becomes free to use and it will be the root of an open source project. The data file format is open and XML-based to save the data organization. In this paper, we present our experience in using UML and a CASE tool to design a hydraulic toolkit for Water Distribution Analysis. Firstly, the technologic choices are described.

2. TECHNOLOGIC CHOICES

Porteau and its three computational targets were rebuilt from scratch from 2004 to 2008. Previous code in 2003 was written in C++ but the GUI, the network model and the solvers were interconnected and not independent. It was advised to redesign and rethink everything. An incremental lifecycle model of development was adopted.

That is, the product is designed, implemented, integrated and tested as a series of incremental builds. Design was realized with the help of the Objecteering CASE tool. Below, the selected technological choices are explained.

2.1 Target languages

A distinction was made between:

- The source code that is automatically produced by the CASE tool that defines the network model layer. Model driven engineering mode was selected so that the code always corresponds to the model. In fact, it is only possible to modify declarations by modifying the model itself.
- The code that is human-made for the GUI interface. It is in an independent layer that is not described in the CASE database;
- The source codes that produce the different solvers in an independent layer.

C++, C# and Java were compared with regards to their facility of use, the graphic library availability and the possibility to be interoperated on multiple computer platforms. C++ was removed from the list for the two first layer items. The main inconvenience concerned the libraries for GUI implementation that were at this time OS dependant. Now, it has improved and several multiplatform libraries are available [7]. At the time of the decision, in 2004, Java was preferred because of its maturity compared to C#. Because C++ language is more efficient for pure calculation than Java, it was decided to program the mathematical modules in C++. Moreover, template library exists that allow a high level of generality to be obtained. As a consequence, the same routines can be used for dense, sparse and distributed matrices.

2.2 Graph and chart solution

The IBM Ilog Jviews Enterprise library was chosen for its capacity of representing different views of a graph and its facility of development. This is proprietary solution software that has to be paid for. Cemagref are currently coding their proper chart and diagram solution to get a complete independent and open source solution. The main window is composed of four views that are shown in Figure 1:

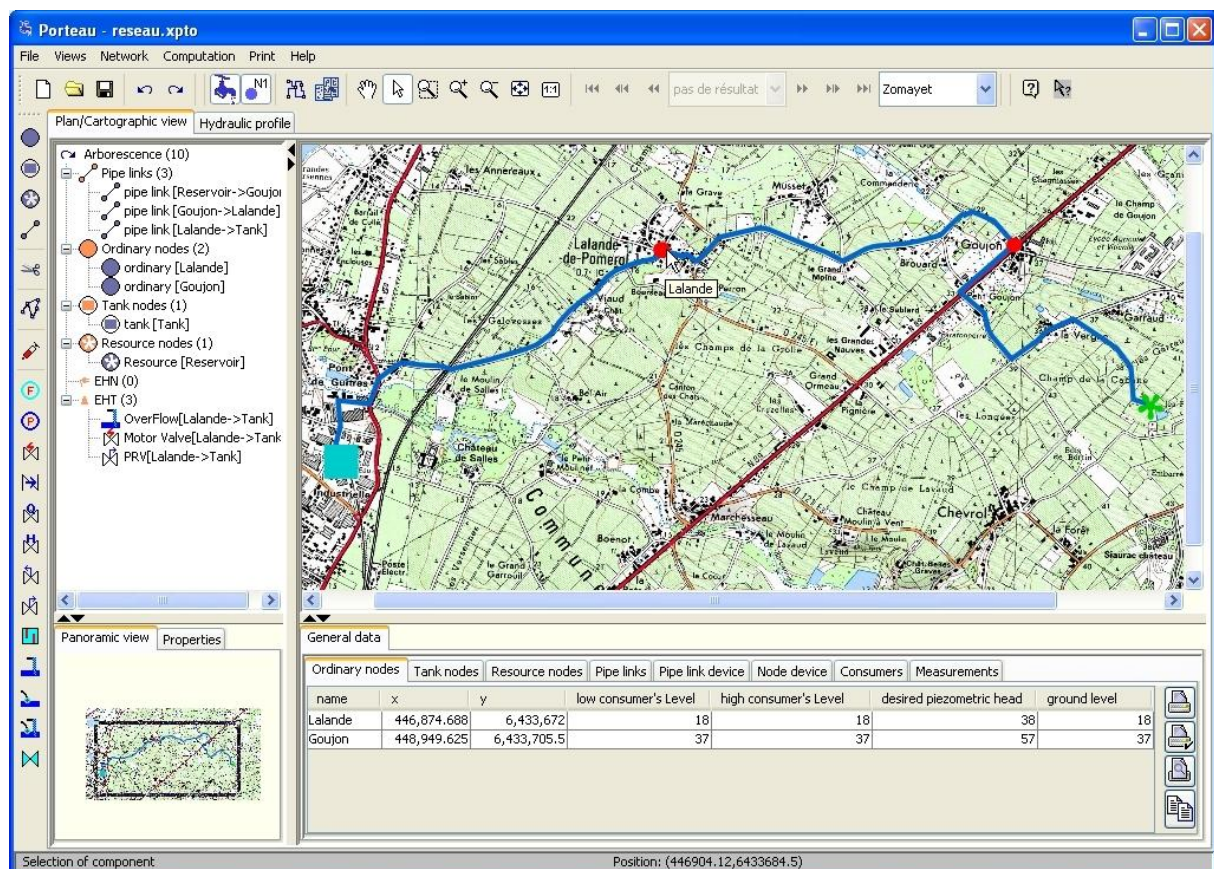


Figure 1. Screen of the Porteau software

The standard main user interface is split in four parts or views, namely, a data explorer, a graph view that can be superposed on a map, a panorama view and a spreadsheet.

2.3 Standard format for data exchange

End-users of “Porteau” have expressed their need for standard formats making data exchange between different software easier. In order to come up to their expectations, we worked out a data format suitable for Water Distribution Networks. An adaptable and well-documented structure is provided to facilitate information exchange. The choice of XML (eXtensible Markup Language) allows separating the characteristics of a document, i.e. its content (raw data), its structure (data organization):

- The structure is defined using a conceptual model where relations between objects are described. It controls and validates associated XML data files;
- Data export and import have been considered for different software such as EPANet, Excel, GIS.

XML allows more. In a student project we succeed to display just by a click the different views of the networks by browsers such as Netscape navigator or Internet Explorer.

3. MAIN PACKAGES AND CLASSES OF THE NETWORK MODEL

The CASE model contains the description of business classes and utility classes for data exchange and persistence and for interfacing with computational targets.

Main packages for business classes are given in Figure 2 below. They correspond to twelve concepts that were created to describe the water distribution system analysis and requirements.

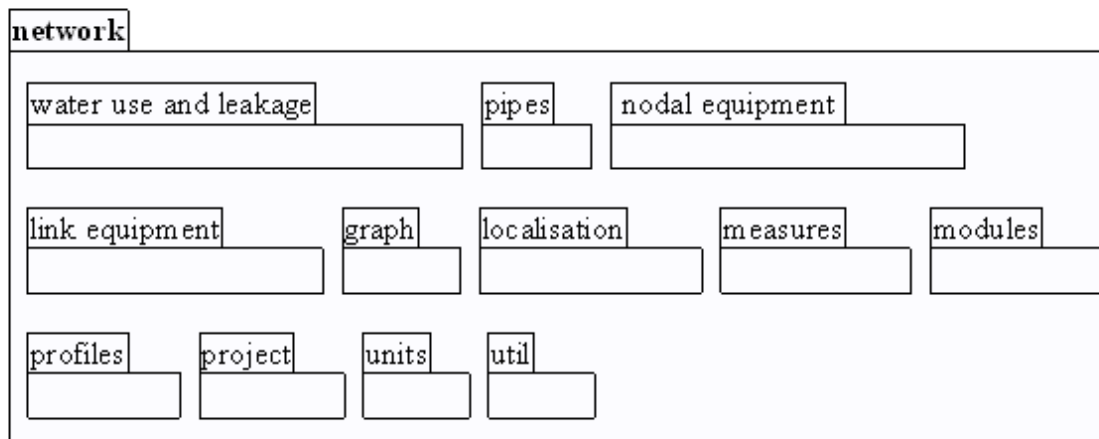


Figure 2. Package diagram of the Porteau network model

The graph package and its classes are central. Graph elements consist of nodes, links and hydraulic devices. The class diagram for the Link class and its main associations are shown in Figure 3. Each link possesses two extremity nodes (a start and an end node). It is represented by a straight segment in the schematic view and by a broken line in the map view. Hydraulic devices are either carried by a link at a given position or by a node. So a collection of link hydraulic devices may be placed on a link at different places. A link represents pipes in series. Given or equivalent characteristics may be retrieved by referring to the corresponding Pipe. Domestic use and background leakage (in progress) may be given “en route” associated to a link. Several specific “water use and leakage” classes were defined for domestic use, industrial use, local leakage, water exchange and specific irrigation use. Any element of the graph can be activated or not. The presence of two different attributes, namely, roughness classes and local roughness is mainly for calibration reasons.

Abstract classes are superclasses that cannot be instantiated. They are named in italics in Figure 3. This means that LinkEquipment and Node classes possess child classes that are specialized for appropriate purposes and representation. The Consumer and Pipe classes are defined in other packages than the graph one because they participate in others concepts and ideas.

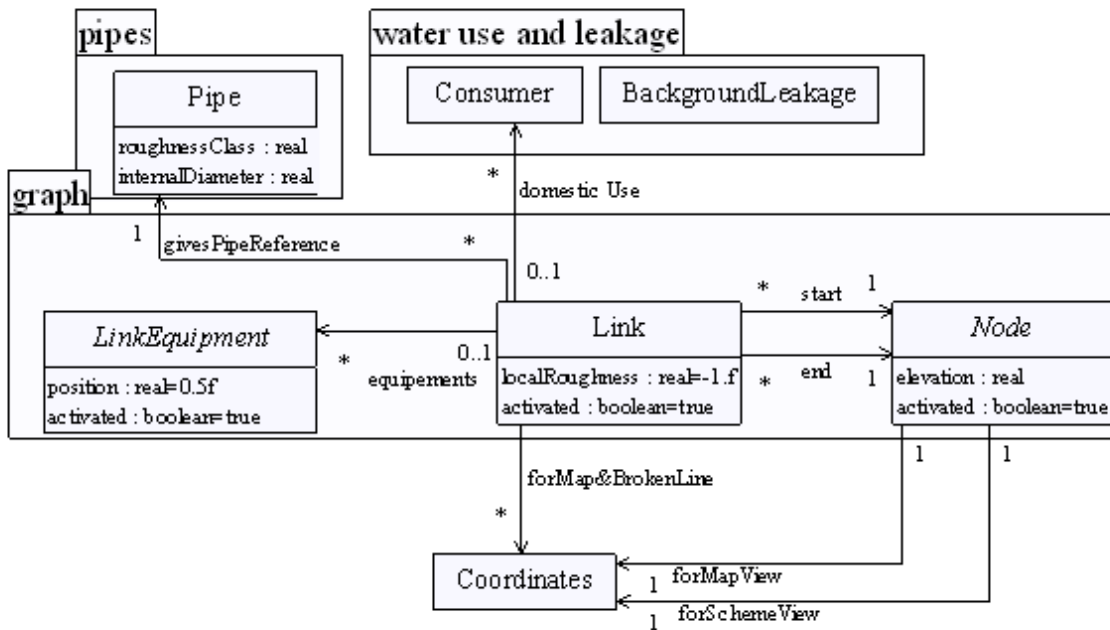


Figure 3. Class diagram for the Link class and its main associations

The Node class is also another main concept and a class diagram is given in Figure 4. Node is a parent class that can inherit any of the three child classes: Reservoir, Junction and Tank. Common attributes and methods are defined at Node level. For example, the four associations toward Coordinates, Service and Sector correspond to private member data that are common for every Node. A Junction node instance can be associated to any water use while Tank node water uses are restricted to Domestic and Industrial.

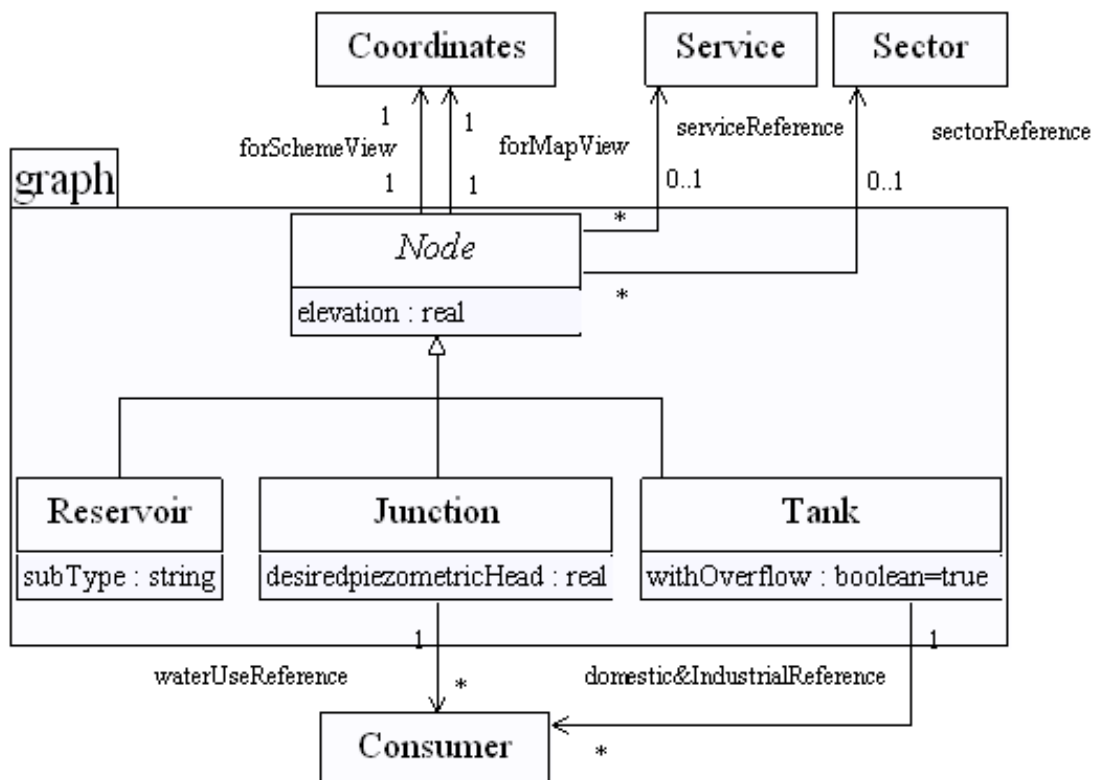


Figure 4. Class diagram for the Node class, its inheritance or child classes and its main associations

Below are some illustrating metric results for the Java code and the network model layer. The Java code for the GUI and the calculation modules layers are not counted here.

Table 1. Metrics of the network model layer

	network	graph	modules	UML base total
Number Of Classes (NOC)	148	11	63	218
Number Of Attributes (NOF)	252	50	97	522
Number Of Methods (NOM)	1407	264	451	1738
Number Line Of Code (NLOC)	21071	3490	8175	45305

Design patterns have been used, for example in the regulation rule implementation with the “Composite” structural pattern.

4. CALCULATION TARGETS

For the moment, three calculation targets are available. The module Zomayet carries out an extended period simulation for hydraulic modelling. It consists of a determinist analysis of the network. On the other hand, the Opointe module realizes a stochastic model for peak period that predicts, at a given alpha risk (type I error), upper limits for link flow rates that will not be exceeded and lower limits for the nodal pressures. This calculation target has been recently extended for irrigation networks and appropriate Peak Demand Diversity Curves. Finally, the quality module solves advection-reaction equations for understanding of minimum, mean and maximum water ages, source tracking and the fate of waterborne component concentration (free chlorine for example). These modules are compiled in distinct DLLs and Java Native Interface (JNI) is used to communicate between the modules and the network model. The “modules” package is specialized for pre-treatment and post-treatment. In Figure 5, an example is given with the Opointe class and specific input data in modules.opointe.data and results in opointe.results packages.

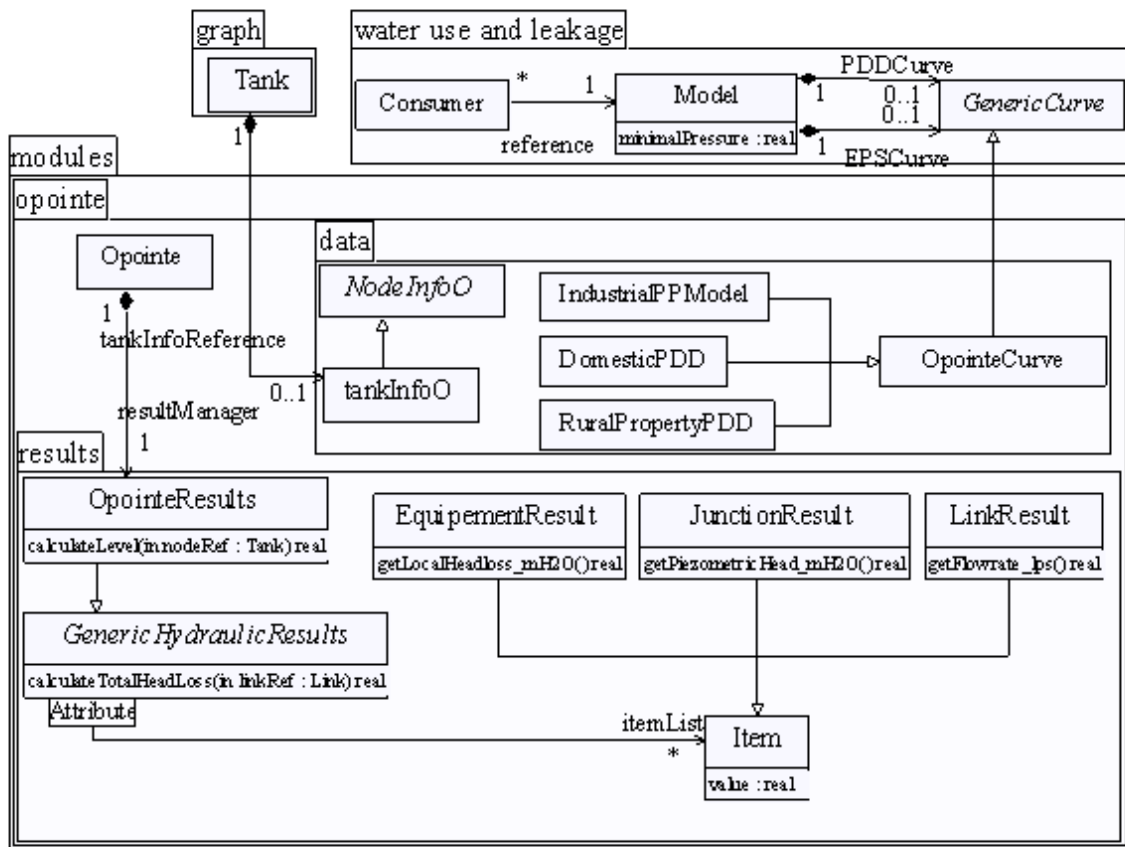


Figure 5. Class diagram for the Opointe class and its main associations

At Nodes and Links levels (in Fig. 5), Consumers instances are associated to a consumer Model that refers to specific curves and data for both hydraulic targets. An example for the Opointe context is the DomesticPDD child class of OpointeCurve, a specific class for domestic use that contains Peak Demand Diversity Curve information. Another point is that the same generic hydraulic results are defined for an Opointe result and a Zomayet steady-state result. The class OpointeResults specializes some operations. Finally, the minimalPressure real attribute of the Models class has been recently added for pressure-driven analysis of water distribution systems. This consists of options that are currently developed for Porteau. Similarly, a failure probability is easy-to-add at each link and reliability analysis.

5. CONCLUSION

In this paper, we present the main results on designing the Porteau hydraulic toolkit for Water Distribution Analysis. It is object-oriented coded and a Computer-Aided Software Engineering (CASE) tool has been used. This year the software becomes free to use and a Porteau open-source project is planned.

Technology choices have been explained. Java was used for the network model layer and the GUI. The calculation modules are coded in C++ with Java Native Interface to communicate with the Java Runtime Environment. The IBM Ilog Jviews Enterprise library was chosen for graph drawing. We are currently coding our proper diagram solution to get a complete independent and open source solution. XML (eXtensible Markup Language) completes the solution for data persistence and saving.

The twelve packages that compose the network model layer are presented. A focus is made on the graph package. It consists of links, nodes and equipments. Hydraulic devices are either carried by a link at a given position or by a node. Domestic use and background leakage (in progress) may be given "en route" associated to a link. A Junction node instance can be associated to any water use while Tank node water use is restricted to Domestic and Industrial use. Approximately, 150 classes, 250 attributes, 1,400 methods and 21,000 lines of code have been defined for this layer.

The calculation targets are described with an emphasis on the stochastic hydraulic model Opointe. Specialized classes for pre-treatment and post-treatment are explained. An example of extension ease with pressure-driven modelling and a reliability measure is given.

Porteau has been designed for water distribution and irrigation systems. Possible extensions should concern cold and hot water private networks such as hospitals, old people's homes and thermal stations.

References

- [1] EPANET 2 distribution, USEPA, "<http://www.epa.gov/nrmrl/wswrd/dw/epanet.html>" accessed on May 10, 2011.
- [2] L.A. Rossman and J.E. Van Zyl, "The open sourcing of Epanet.", in *WDSA 2010 conference*, Tucson, AZ, USA, in CD.
- [3] Van Zyl and Y. Chang, "Programming styles for an open source Epanet project.", in *WDSA 2010 conference*, Tucson, AZ, USA, in CD.
- [4] J.E. Van Zyl, J. Borthwick and A. Hardy, "OOTEN: An objected-oriented programmers toolkit for EPANET.", In: *Maksimović, C., Butler, D. and Memon, F.A. (eds.) Advances in Water Supply Management*, A.A. Balkema Publishers, supplementary paper, 2003.
- [5] M. Guidolin, P. Burovskiy, Z. Kapelan and D. Savić, "CWSNet: An object-oriented toolkit for water distribution system simulations.", in *WDSA 2010 conference*, Tucson, AZ, USA, in CD.
- [6] PORTEAU, Cemagref. "<http://porteur.cemagref.fr/>", in French accessed on May 10, 2011.
- [7] Free GUI Libraries and Source Code, "<http://www.thefreecountry.com/sourcecode/gui.shtml>", accessed on May 10, 2011.