Latest advances of the BASHYT framework: a web, GIS oriented, interface for SWAT

P. Cau, S. Manca, C. Soru, D. Muroni

Center for Advanced Studies, Research and Development in Sardinia

pierluigi.cau@gmail.com

CRS4 (http://www.crs4.it)
Parco Tecnologico POLARIS, 09010 Pula CA, Italy

Presentation outline

1. Description of BASHYT and LIVE Demo
2. Technologies
3. Interoperability and Interfaces
4. Conclusion

BASHYT is a web based software to expose **SWAT** results on the WEB. A free open to use service is available at: http://swat.crs4.it/Home

BASHYT is a Collaborative Working Environment (CWE): a easy to use and extensible development framework, for constructing spatially enabled web applications based on the SWAT model (but not only).

BASHYT is a *problem-solving* platform *for the* **Environmental Sciences** for the integration of

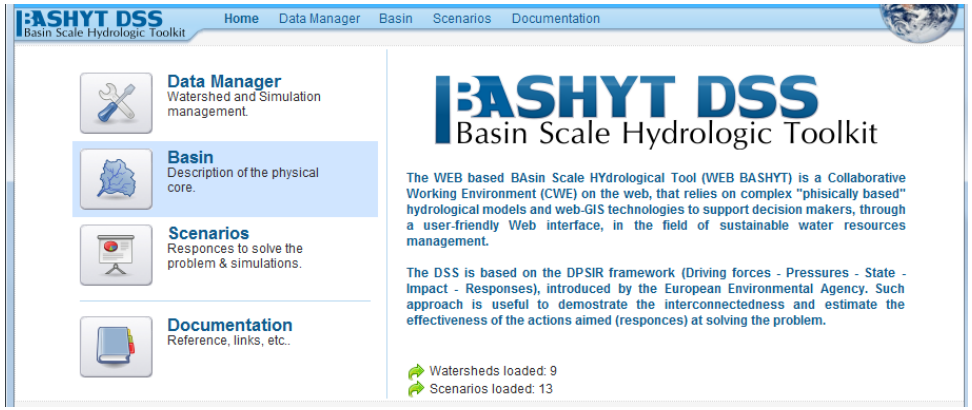
- *resources for*
  - *communication*
  - *computation*
  - *data storage*
  - *visualization*
- *simulation software*
- *instrumentation*
- *human know-how*

Objectives of the framework

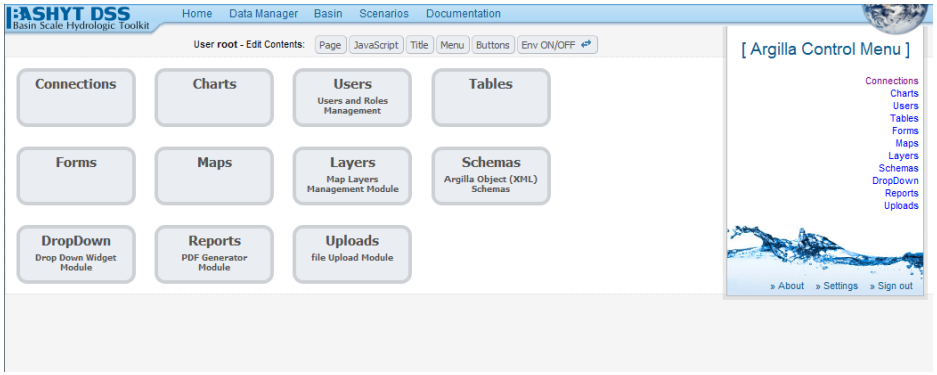
- ✓ Enable users the expose their SWAT simulation on the WEB and to ease the **report production mechanism**
- ✓ Share data, knowledge through a web based environment
- ✓ Expose **interoperability services** on the WEB to create a broader user experience
- ✓ Enable developer build applications based on other portals exploiting web interoperability services: **mesh up of web applications**
- ✓ **Bridge the gap between science and end users / citizens!**

BASHYT works in tandem with the AvSWAT/ArcSWAT GIS desktop programs

Home page / access point to the portal

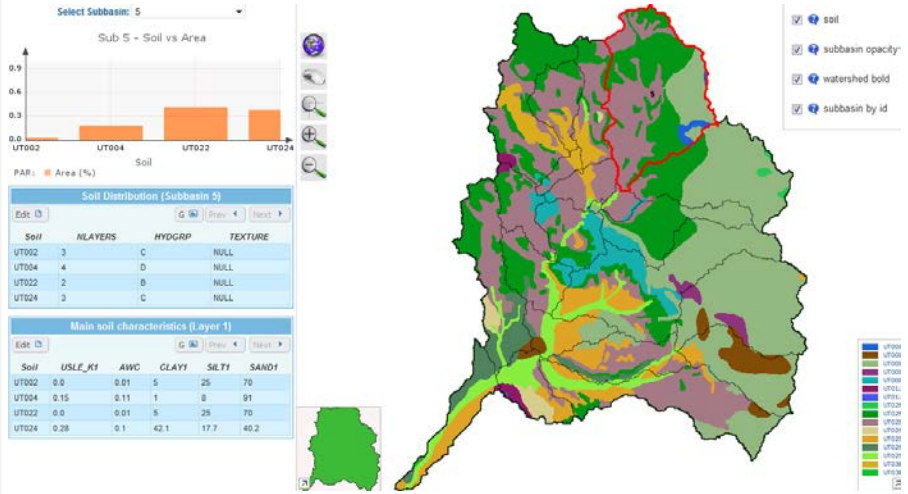


Module section to design Applications



http://swat.crs4.it/Home

BSHYT digests SWAT data and expose on the WEB dynamic reports

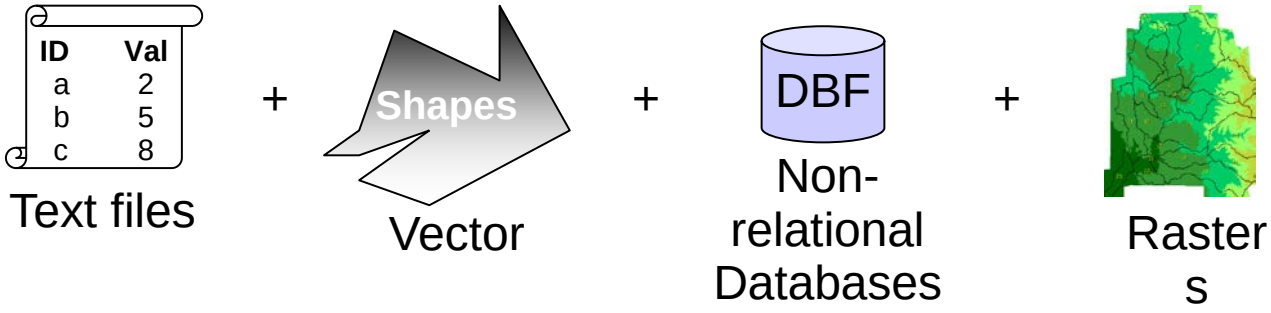


The main features

- ✓ All the model-related data are organized **into complex Relational DB infrastructures**.
- ✓ Exploit user-roles policies to define complex security and access strategies and differentiate the interfaces;
- ✓ Applications can be edited directly using the browser (wiki like), no external plug-in is required;
- ✓ The BASHYT is developed using the **Argilla Java framework** and is exposed via the Tomcat servlet container
- ✓ Argilla provides a live programming web template environment, based on Apache Velocity
- ✓ The GIS rendering is based on the Mapserver technology (server-side) and visualized by **msCross**, a AJAX client-side interface

SPRITE/SWATSL: SWAT Data processing

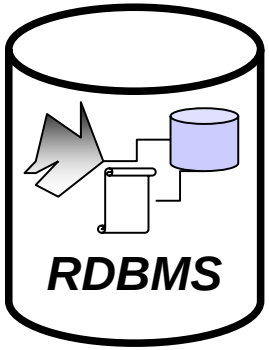
The SWAT model uses several different data formats



BASHYT needs to access data through a relational database

A client-server procedure, namely SPRITE and SWATSL, imports the data into relational databases on the server.

Imported data include: *output.rch, output.sub, AvSwatDB/*.dbf, ESRI Shapefiles, Toporep.txt, etc.*



BASHYT natively implements JDBC connections.

The SpatiaLite engine is used for data archiving of GIS data.

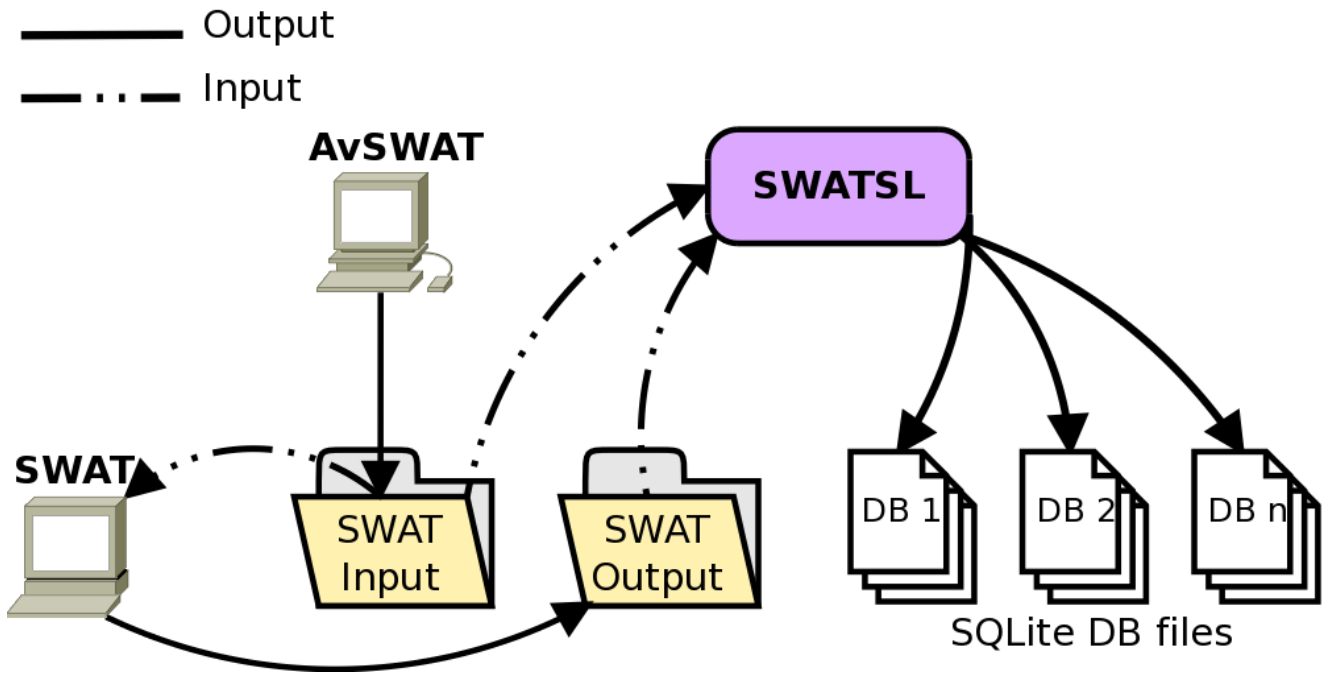
SPRITE/SWATSL data flow

SWATSL is a C++ library that hide the complexity of the SWAT files architecture providing a uniform structured dataset for the user

SWATSL builds one or more SQLite database files and populate them with SWAT simulations (output/input)

After SWATSL has done its work, data can be accessed using common SQL queries

SPRITE is a Java windows program to retrieve I/O from a SWAT project and upload the data to BASHYT



BASHYT Interoperability

BashytAPI

The **BashytAPI** is the client library developed in the Java programming language to access and use the CWE web service interface.

The **BashytAPI** offers a uniform way of identifying and accessing to resources, and thus increasing the interoperability between applications.

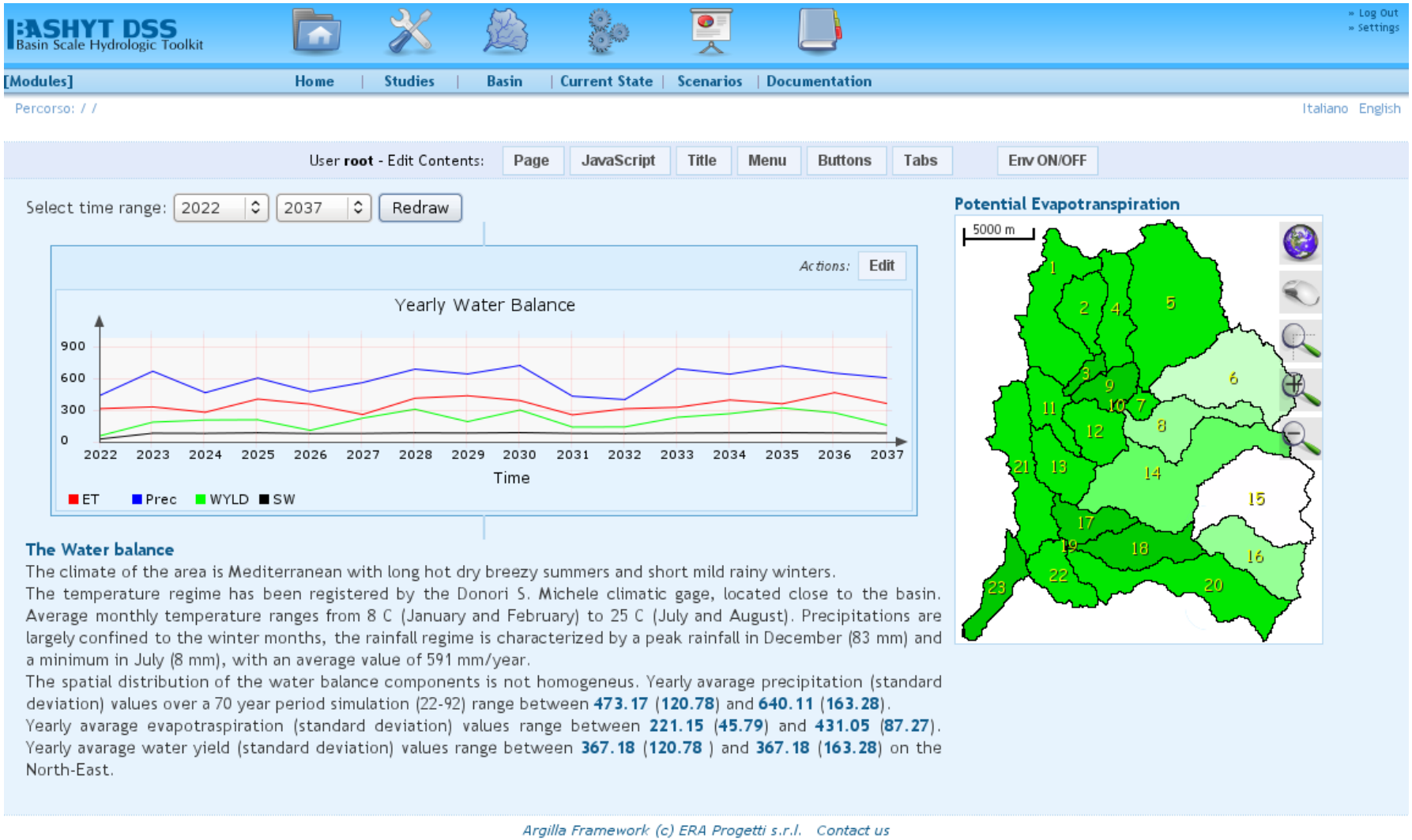
The argillaAPI explicitly targets the needs of other web environment (e.g. **eGLE** or **gSWAT)**

The WEB Front End

The application level of the CWE portal exposes a set of web applications and services:

- **Data Manager**
- **Basin**
- **Scenarios**
- **Documentation**
- **Argilla Control Menu**
  - **the Module section**: Connections, Users, Charts, Tables, Maps/Layers, Forms, etc.

Example of a Web applications



The Argilla engine

The engine allows to **integrate several client and server technologies** in a single **development envir**onment, fully programmable and **accessible** by the **web browser**.

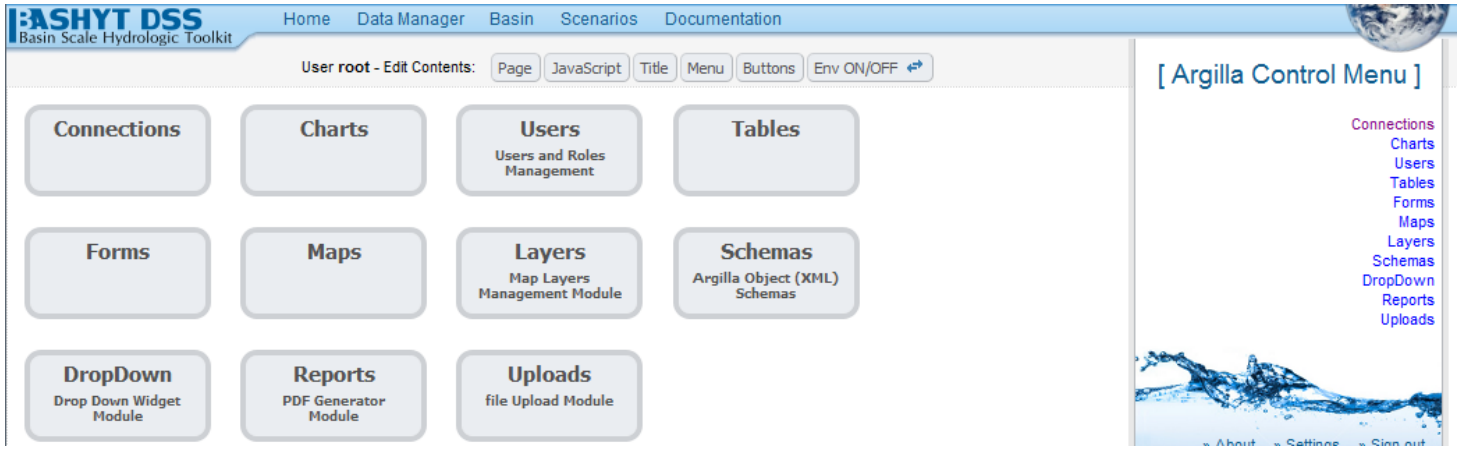
Developers can write **server side codes, and use the framework tool for debugging and validation**. The Velocity Template allows a strong integration with low-level API written in Java (working as PHP does).

All web applications and pages exposed are described in a **structured and hierarchical way within the virtual filesystem**:

- In such hierarchy, each folder is a node of the portal: each node is accessible from the browser via a specific URL, and contains (virtual) data files such as the Velocity scripts, HTML, JavaScript, which contribute to the composition of the page

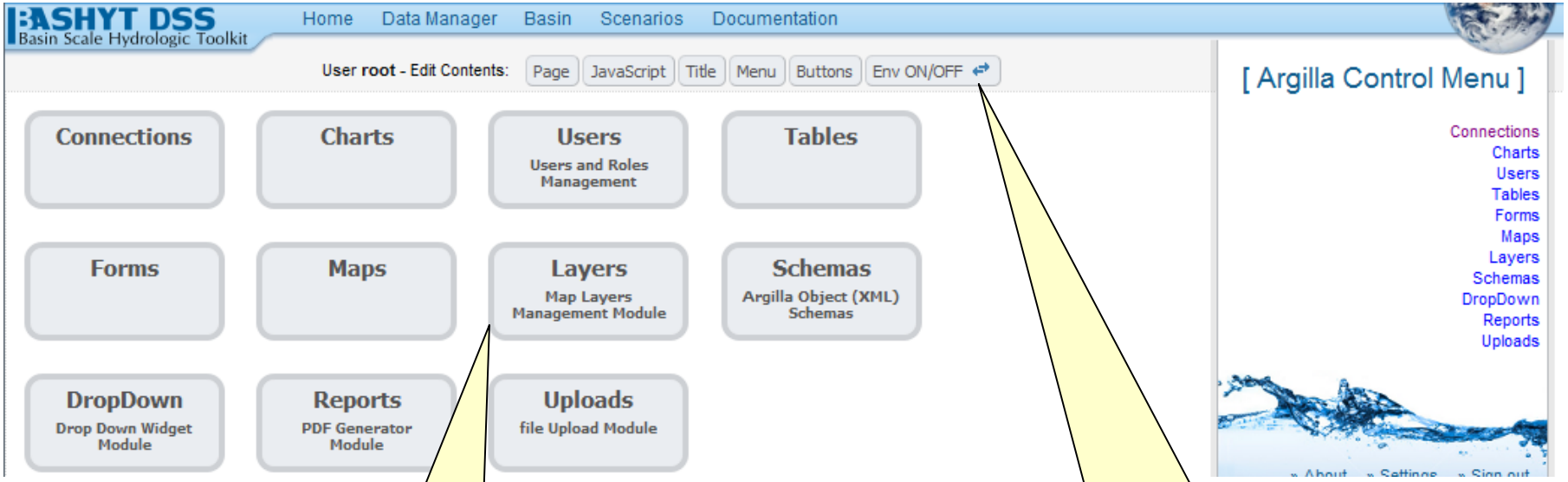
The Modules

The module section exposes though easy to use web interfaces a variety of services to **shape XML objects for charts, maps, tables, PDF, and forms production**.



Modules permit the massive use **of preset schemas** stored in the database in a structured XML form. Each object refer to its schema and describes parameters (e.g. to control layout) and data sources.

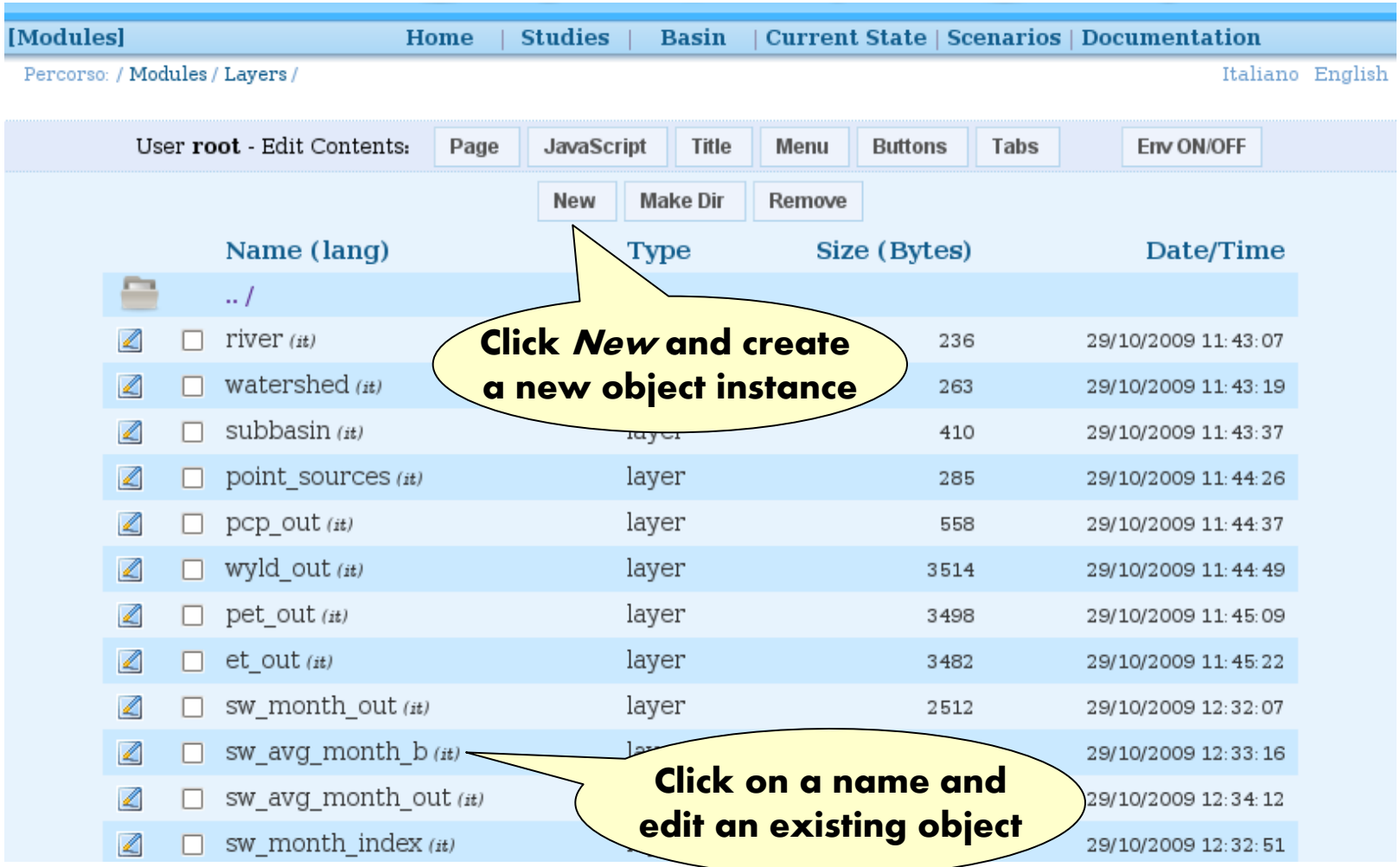
The CWE report production environment



Click on this object to show the list of available layers

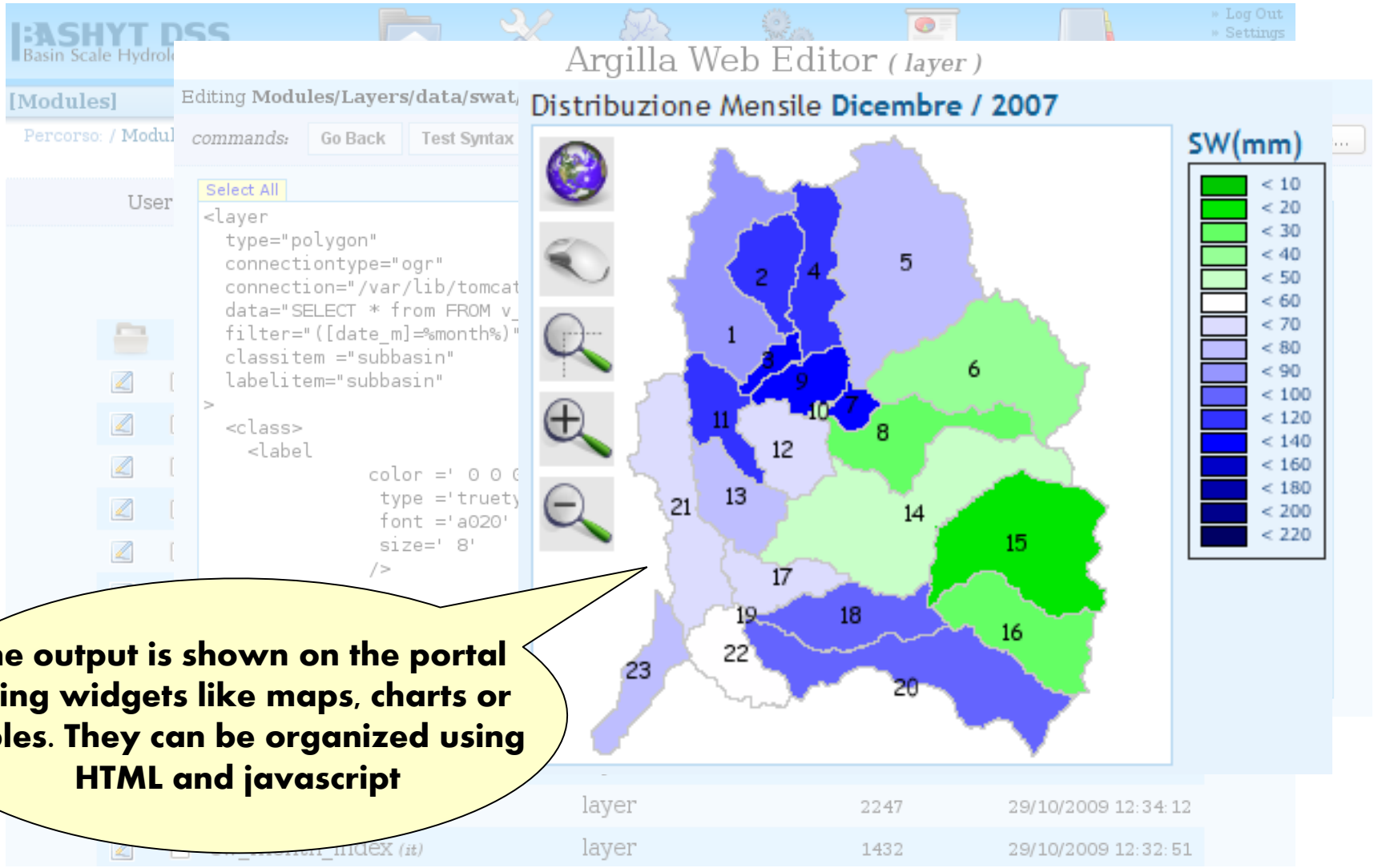
Users with the *administrator* role can controll the page layout direcly on the WEB

The report production



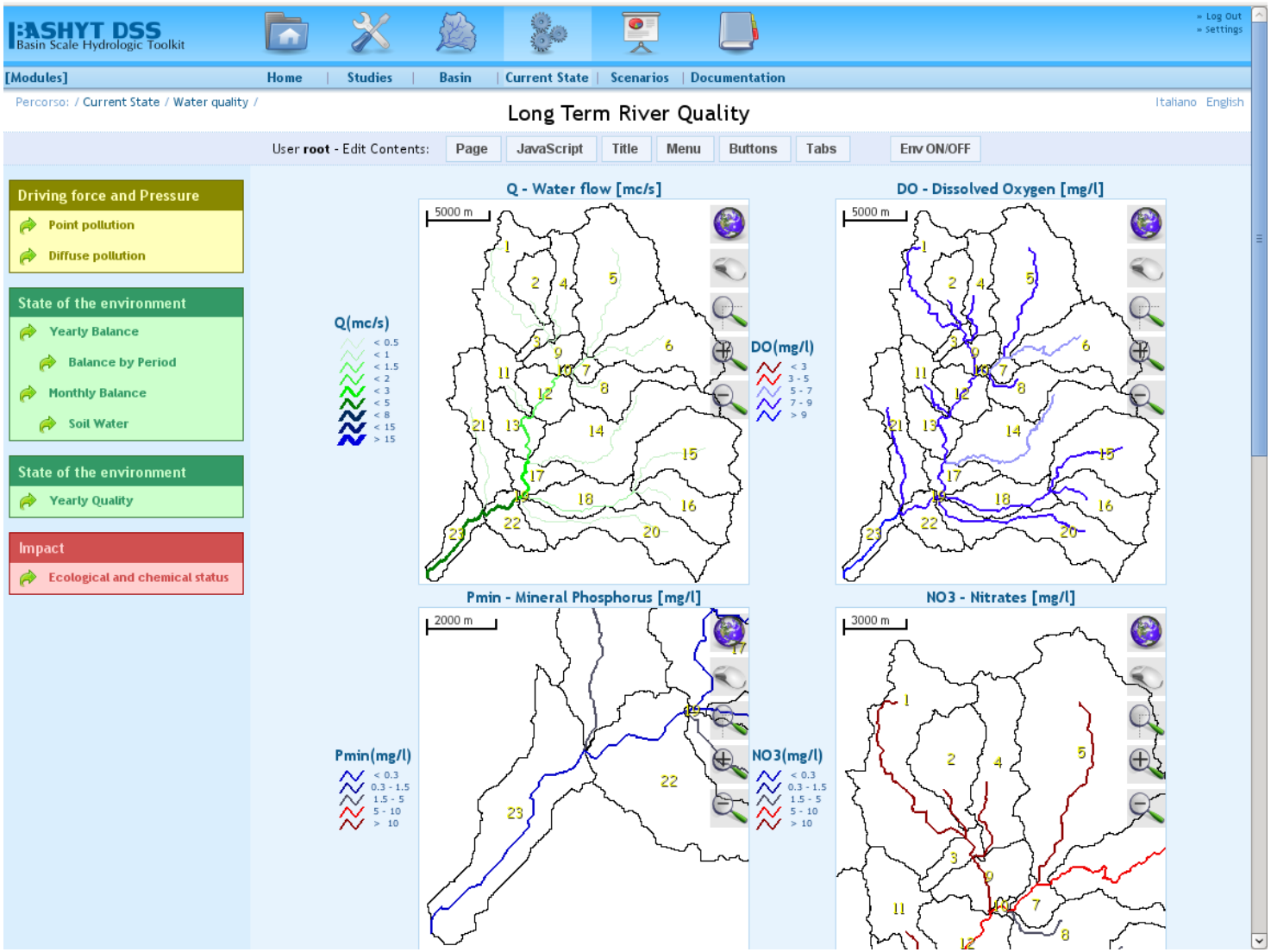
The report production

The BASHYT/Argilla production environment

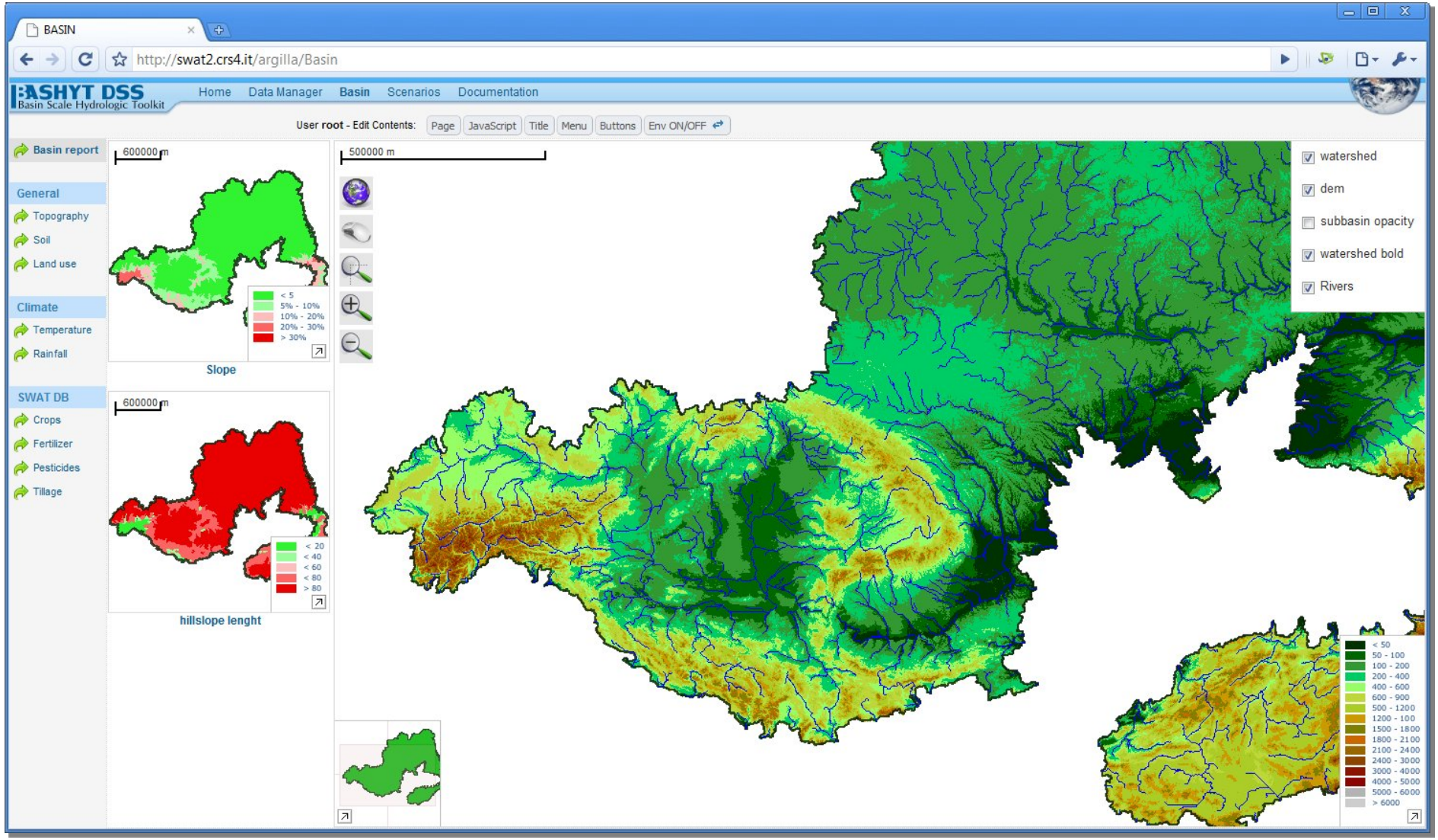


The output is shown on the portal using widgets like maps, charts or tables. They can be organized using HTML and javascript

Web Interactive Interfaces



The Black Sea Catchment

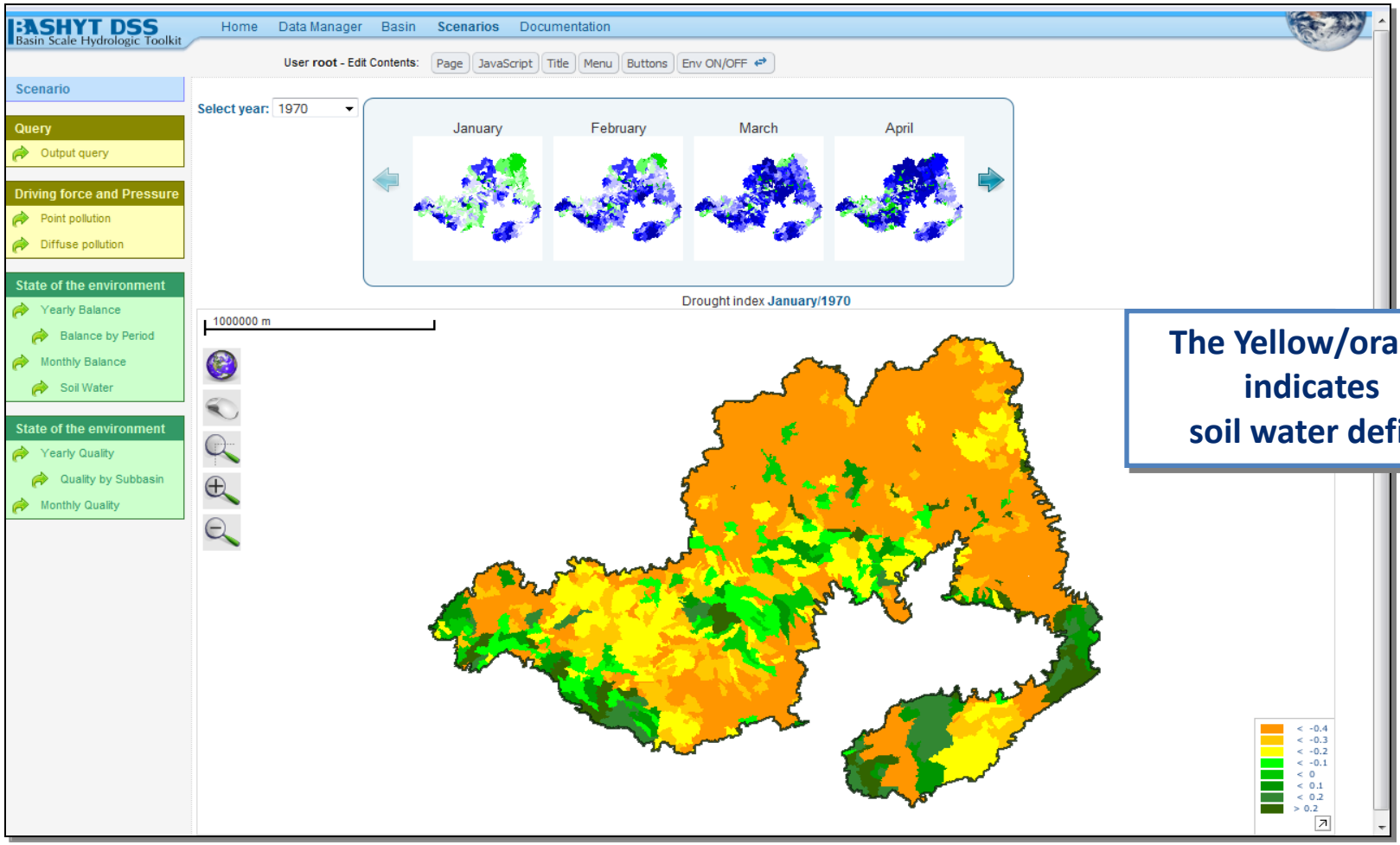


The Black Sea Catchment

Modeling Environmental Dynamics: water quality and quantity states on rivers

Modeling Environmental Dynamics: the agricultural drought for the Black Sea catchment

The Black Sea Catchment



The Yellow/orange indicates soil water deficit

Future work will be done to:

- improve web services (**O.G.C.** services: **WFS, WMS**, …)
- expose **catalogs of geodata** and simulations

We will be working on **scenarios production mechanisms** to be run directly on the web

We will integrate climate (ensemble, ERA 40, etc.) data into BASHYT data flow from web data providers.

BASHYT works in tandem with the pre-processing **AvSWAT** and **ArcSWAT** GIS production environment.

BASHYT is the first web interface to SWAT that exposes a **fully programmable environment** to construct spatially enabled applications on the WEB.

It has been widely tested on real case studies on available datasets: Gange (India), Cedrino (Italy), San Sperate (Italy), Black Sea Catchment, ...

first working version of BASHYT is up and running at:

http://swat.crs4.it/

Register to use the system

This work has been supported by:

EnviroGRIDS (http://envirogrids.net/): With 30 partners distributed in 15 countries, the project is contributing to the Global Earth Observation System of Systems (GEOSS) by promoting the use of web-based services to share and process large amounts of key environmental data……

CLIMB - **Climate Induced Changes on the Hydrology of Mediterranean Basins ((**http://www.climb-fp7.eu/): Innovative scientific and technological measures will play an important role in addressing projected climatic changes and their impacts on the freshwater resources of the …..

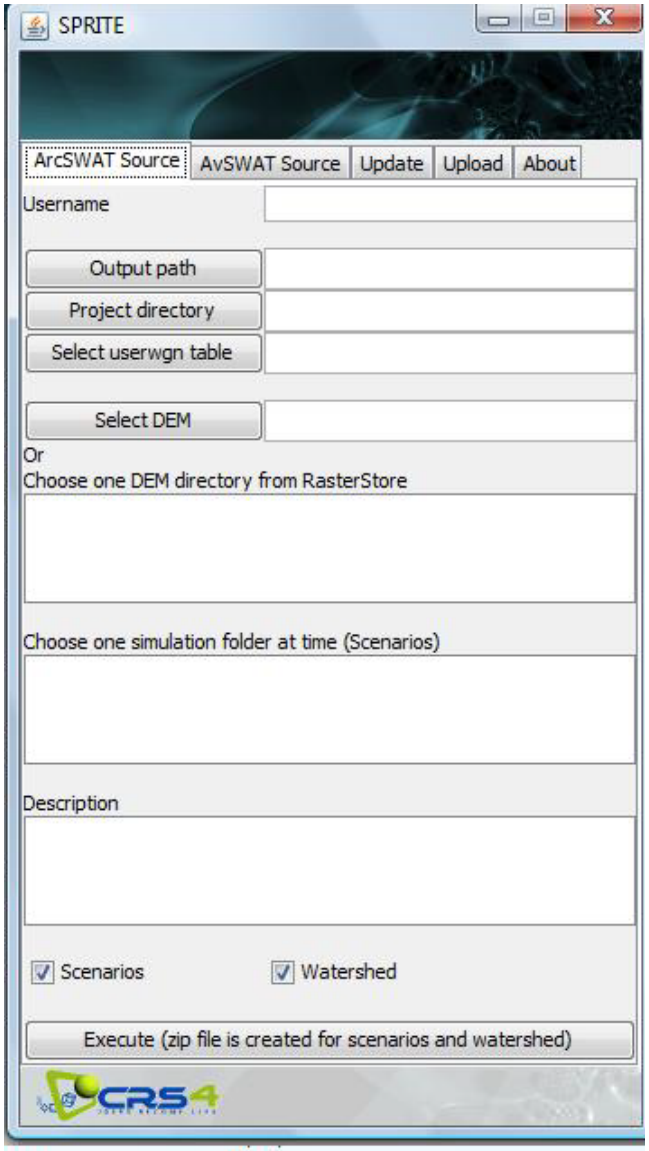
Regione Autonoma della Sardegna – RAS (Italy).

SPRITE

Sprite is a stand alone Java program that process AvSWAT/ArcSWAT projects to extract the necessary information to be uploaded on BASHYT:

The main tasks performed by SPRITE are:

- *Extract* a minimum dataset
- *Transform* it (normalize its content)
- Populate a XML metadata file
- archive the data in 2 zip folders:
  - Watershed
  - Scenarios
- connect and *upLoad* the data to any BASHYT server



SWATSL

SWATSL is the server side application and work also as a standard ETL. It is programmed in C and its purpose is:

1. *Extract* the data,
2. *Transform* it to fit the operational needs
3. **Create an empty logical schema** of the geo-relational database (a spatialite db file). Such schema is fixed.
4. *Populate* **it**. SWATSL will import the data within the schema.

- The transform stage applies a series of rules and functions to the extracted data from the source to derive the data for loading into the db file.
- SWATSL can be commanded from the application side, so each user of the portal with the privileges will be allowed to run it to import the uploaded projects within the system.