A cloud based tool for knowledge exchange on local scale flood risk

Wilkinson, M. E.; Mackay, E.; Quinn, P. F.; Stutter, M.; Beven, K. J.; Macleod, C. J. A.; Macklin, Mark; Elkhatib, Y.; Percy, B.; Vitolo, C.; Haygarth, P. M.

Published in:
Journal of Environmental Management

DOI:
10.1016/j.jenvman.2015.06.009

Publication date:
2015

Citation for published version (APA):
Wilkinson, M. E., Mackay, E., Quinn, P. F., Stutter, M., Beven, K. J., Macleod, C. J. A., Macklin, M., Elkhatib, Y., Percy, B., Vitolo, C., & Haygarth, P. M. (2015). A cloud based tool for knowledge exchange on local scale flood risk. *Journal of Environmental Management*, *161*, 38-50. https://doi.org/10.1016/j.jenvman.2015.06.009

Supplementary material

Electronic Supplement 1: Cloud infrastructure, models and visualisation tools

The LEFT utilises a cloud version of TOPMODEL (Beven and Kirkby 1979) which is an established quasi-physical processed based model. TOPMODEL was implemented in R and deployed as web processes following a standard suggested by the Open Geospatial Consortium (http://www.opengeospatial.org/standards/wps). The processing workflow consists of three main steps:

1. The user selects a catchment of interest and a model to use
2. Based on the selected location the system queries a local database and retrieves necessary model inputs and parameter sets.
3. Finally, the simulation returns time series of stream flow discharge in a machine readable format which is then translated into an interactive plot.

A key objective of EVOp is to lower the entry barrier and to enable universal access to data and models. Hence, all models and data are made accessible through standard web browsers. This is achieved at the browser side via dynamic HTML and HTML5 web elements (blocks of text, images, buttons, links to other websites etc.), CSS (Cascading Style Sheets; where a style defines how to display HTML elements), browser scripting using a number of JavaScript libraries, and background communications using AJAX (Asynchronous JavaScript and XML). AJAX is not a programming language but a standard means of exchanging data with a server and updating parts of a web page without reloading the whole page.

EVOp applications handle different types of data products, for instance, geospatial datasets and time series. Geospatial data is visualised using interactive layers superimposed over maps. We surveyed a lot of map APIs (Advanced Programming Interfaces) including Google Maps and Earth, OpenLayers, and Bing Maps. Google Maps was selected owing to its wealth in data, features, and its familiarity with the general public. Its programming interface, API v3, allows catchment shapefiles (polygons in ArcGIS v10) and points of interest (markers) to be easily added to the map. It does not require anything to be downloaded and installed (in contrast to Google Earth). The interactive nature of the geospatial layers allows us to expand the visualisation to include time series graphs over specific map locations. We experimented with a number of graphing libraries, such as D3, HighCharts, Protovis, Simile Timeplot and Sparklines.js. However, we chose the Flot plotting library for its programming flexibility and intuitive interactivity. Flot is a JavaScript plotting library for jQuery and supports time series as well as automatic data updating using AJAX (see http://www.flotcharts.org/; accessed August 2014).

Forcing model inputs, such as precipitation and evapotranspiration time series, along with observed stream discharge used for comparison and calibration purposes, are stored in a local PostgreSQL database. The time series are organised in tables based on an “ad hoc” schema and can also retain basic metadata information, important to determine data quality issues. The database can be queried based on time window and geographical location. Database queries and model simulations are dealt in R, while the process deployment is based on a python implementation of the OGC WPS standard (called PyWPS). The communication between R and Python is made possible by wrapping the processing workflow in a python script using the RPy2 connector.

Electronic Supplement Figure 1: Main quantitative questionnaire responses for the LEFT demo