Widening the Circle of Engagement Around Environmental Issues using Cloud-based Tools

Yehia Elkhatib*, Alastair L. Gemmell†, Claudia Vitolo‡, Mark E. Wilkinson§, Eleanor B. Mackay¶, Barbara J. Percy†† ‡‡, Gordon S. Blair*, Robert J. Gurney‡‡

*School of Computing and Communications, Lancaster University, Lancaster LA1 4WA, UK
†Met Office, Exeter EX1 3PB, UK
‡European Centre for Medium-Range Weather Forecasts, Shinfield Park, Reading RG2 9AX, UK
§James Hutton Institute, Aberdeen AB15 8QH, UK
¶Centre for Ecology & Hydrology, Lancaster LA1 4AP, UK
††Institute for Environmental Analytics, Reading RG6 6BX, UK
‡‡Department of Meteorology, University of Reading, Reading RG6 6BB, UK
*Corresponding author: {*i.lastname*}@lancaster.ac.uk

***Abstract*—Environmental data are being generated and collected at unprecedented rates. However, the diversity in form and format of these environmental assets poses challenges for collaborative and reproducible science. Moreover, access constraints that surround environmental data lead to difficulty in use and interpretation of results. Cloud computing offers high potential to break down such barriers and engender collaboration, attribution, reuse, and reproducibility. In this article we review the design of the Environmental Virtual Observatory pilot (EVOp) that was conceived as a cloud-enabled virtual research space for different users interested in environmental science, ranging from domain specialists to the general public. We discuss the key technologies and processes used: a hybrid cloud infrastructure; standard service interfaces; a unified service delivery platform; and a test-driven development cycle. We also discuss the methodology by showcasing one of the exemplars developed in EVOp, stressing the importance of weaving stakeholder engagement from the beginning and throughout the process. We also briefly highlight some of the lessons learnt of working in an interdisciplinary team.**

***Index Terms*—Cloud computing, Environmental information systems, Interdisciplinary teams, Participatory design, Virtual research environments**

I. Introduction

Data are being generated and collected at an incredible rate [1], [2]. However, computer scientists are faced with challenges more pressing than merely enhancing technologies: engagement and impact. This is especially true in issues relating to environmental science (ES), where important society-wide questions are still incredibly difficult to address despite uptake of technologies (*e.g.* HPC, IoT).

Environmental data comes from disparate sources over a variety of spatial and temporal scales with different resolutions and formats [3], [4]. They can be insufficient or incomplete [5], [6], hard to locate [6], [7], expensive to access [7], [8], disconnected from metadata [5], [7], and/or require significant pre-processing before they may be considered usable [9], [10]. It is thus significantly difficult for an individual outside the field of expertise to access data and predictive models[1] and gain meaningful information to answer his or her questions [11]. It is even challenging for many graduate students and early-career researchers who work in the field of expertise [10].

Despite such challenges and great interest from the public and policy makers, software solutions that enable improving collective understanding are fairly few and flawed [12], [13]. Access and use of many tools and data are limited by a variety of factors [8], [14], [15], such as a required high level of understanding of the model and driving data, the need to download the model or data, prerequisites for hardware and/or software configuration in order to run a model, or even merely prohibitive costs of access or execution.

Such an approach is no longer a realistic one to take [16], as environmental issues are increasingly becoming subjects of far-reaching commercial, regulatory and social policies (*cf.* the Paris Agreement), and of serious public interest and debate, with examples around the world such as floods in the UK [17] and sub-Saharan Africa [18], and droughts in Australia [19] and California [20], to name but a few. In this article, we report on the experiences of the Environmental Virtual Observatory pilot (EVOp) project [21] in addressing many of the above issues by not just developing a cloud-based system for integrating multi-sourced data and models, but also adopting development and stakeholder engagement practices that maximise the impact of the developed technology.

Funded as a two-year proof-of-concept, EVOp's vision was to demonstrate the possibility of developing tools that are rooted in environmental modelling and using a myriad of data sources, but that are easy to use by a wide variety of stakeholders (farmers, local communities, policy makers) and that are indeed useful for their various purposes. This paper provides insight into the processes and technologies that were employed in the EVOp project for this purpose.

[1]Henceforth, we use the term '*model*' to refer to mathematical / algorithmic representations of the environment used by environmental scientists to aid in understanding natural systems and foreseeing how they change.

For this project, four exemplars were chosen to answer environmental questions that are based on different user groups, each with varying levels of computer and scientific experience and focusing on different levels of scale. The EVOp web portal was developed to ensure universal access, easy and intuitive use, as well as visual presentation and interpretation of the results. It allows the user to investigate an environmental issue (*e.g.* hydrology) through exploring a range of historical and realtime datasets (*e.g.* observed rainfall, webcam imagery, indicative flood hazard thresholds), instantly running relevant predictive models using preset scenarios or specially defined ones, and comparing current and previous results.

The portal relies on three key advances in distributed computing: cloud computing, web service standards, and Web 2.0 technologies. These technologies are used to ensure a user experience that remains wholly focused on the environment-specific question at hand. At no time during the user's journey through the system should they worry about where datasets reside or how to access them, how to find related or generally relevant data, how to execute predictive models, or what to make of the results they bring up. Instead, the user should focus on such questions as "is my local area susceptible to flood after the past few days' rainfall?" or "what could be done to reduce diffuse pollution affecting the North Sea?". Nevertheless, the user can, if they wish to, find out more about the employed assets; *e.g.* a scientist who wants to know how the data are collected or how a model is calibrated.

We introduce the work done to realise the EVOp portal and describe the technologies behind it. We also comment on the success of different practices in increasing the magnitude of and widening the radius of engagement of the developed portal. From the few use cases we developed in EVOp, we present one use case to illustrate the added value in terms of better communication of the impact of human intervention on the environment. Our objective here is to provide additional knowledge in the community when it comes to realising new services, especially cloud-based ones, that address issues of society-wide interest.

Our high-level contributions are as follows:

- The design of a system architecture that caters to a set of requirements captured from the target user groups.
- How to employ key emerging technologies to implement the designed system, integrating various data and modelling resources of different sources. Moreover, this is implemented over a hybrid cloud infrastructure in order to avoid vendor lock-in.
- Experiences of stakeholder-oriented development methodology to deliver useful and usable products to a user base of varying expertise and capabilities.
- Lessons learned about an interdisciplinary team of scientists working on a topic of high public and political interest.

The rest of the paper is organised as follows. Section II discusses related work. Section III presents the requirements of the EVOp portal and its system architecture. Section IV details the implementation process focusing on the three enabling technologies: cloud computing, web service standards, and Web 2.0 technologies. Section V presents one of the exemplars used in developing the portal, focusing on the process followed to create it. Section VI presents an evaluation by revisiting the portal requirements, while Section VII discusses lessons learned. Section VIII concludes.

II. Related Work

There is a growing body of work on *Green ICT*, focusing on challenges of software sustainability [22], [23], energy efficiency [24], and automated control systems (*e.g.* [25]). However, there is little work in the computer science and software engineering (SE) literature on tool development / adoption and associated best practices for issues of society-wide interest. More specifically, there is little effort on creating software to facilitate shared and open ES processes, and support the wider societal dialogue around environmental issues. We now review some of these efforts, highlighting their limitations.

There are numerous works on software for ES, most of which are **stand-alone tools**. Examples: Whelan et al. [26] present desktop tools for tackling data compatibility and model interoperability issues through semantic mediation; Wang et al. [27] present a desktop tool for matching data from different sources; Dauwe et al. [28] present a multi-agent framework to fuse sensor data from different sources; the Penn State Integrated Hydrologic Model (PIHM) [29] is a prototype watershed model to predict water distribution.

Few efforts to date, however, capitalise on the advantages of **cloud computing**, particularly how to harness ease of access to shared infrastructure resources. Bhat et al. [30] discuss why and how to deploy cloud-powered geographical information systems, but their proposal is speculative and not evaluated. Sun [31] put forward a case study of migrating a preexisting system to the cloud and using general purpose web-tools (*e.g.* Google Fusion Tables) as a front-end. Sun's experience might be useful for similar migration projects, but it has significant drawbacks primarily in terms of usability and interoperability: the front-end tools are quite generic and, based on our experience, do not engage non-domain experts; the back-end is too bespoke and could not be used to plug in a new service or to compose existing ones. More advanced integration of data pipelines and workflows has been presented by Pipeline61 [32] but, in comparison to EVOp, it lacks an engagement-lead end-user ready interface for running integrated scientific modelling workflows.

In terms of SE processes for **interdisciplinary teams and public stakeholders**, there is gap in the literature in terms of designing and developing software to be used by a wide range of users of different computational abilities and interests. Demir and Krajewski [33] describe a platform that aggregates flood-related data to provide information to communities in Iowa, US. However, the authors do not comment about their development process, how the stakeholders (local communities or otherwise) are engaged, or how to do more than just view periodic flood forecasts (*e.g.* run a model on demand).

According to recent studies [34], [35], this gap in computation time and model flexibility poses a limitation to the usefulness of such environmental information systems. Quiroga et al. [36] developed an uncertainty-aware cloud-based modelling platform. The work focuses mainly on the computational gain and mentions little on the development process or access to different stakeholders.

As such, the computer science and SE community is in need of sharing experience on how to to use modern technologies (such as the cloud, fog, and IoT) to address a wide-reaching subject such as ES.

III. Design

This section introduces the design of the EVOp infrastructure and the requirements that have driven it.

A. Requirements

The EVOp vision was to provide a multifaceted portal that gives access to different information for various user groups. Our target user groups are: *a*) environmental scientists; *b*) policy makers; *c*) local communities directly involved such as farmers; and *d*) the general public. These user groups include both domain specialists and non-specialists, and they cover a very wide range of potential interests. However, the combination of knowledge from these users can aid in answering key environmental questions. For example, environmental scientists would want to find or upload data, use it to run predictive models, modify models to their requirements, and compose workflows that consist of a series of data manipulation procedures. An officer for a statutory authority would be seeking answers to a 'what if' question that would go some way to support their decision making processes. An engineering consultant carrying out a field survey would find it useful to check the current state of water levels, embankments, vegetation, culvert obstructions, etc. against recorded values and historical webcam images. A member of a local community might be interested in obtaining information about the impact of different farming and water management practices.

The above examples illustrate that the portal potentially has to provide very different information types from a single data set, and to present them in different ways that are accessible and convenient for the intended user. The portal also has to have access to a varied mixture of data sources for the users to explore and utilise. These include live data feeds (such as real time river level, temperature, etc.), historical time series or spatial datasets (*e.g.* rainfall measurements and digital elevation models) and others (*e.g.* webcam images). Such data sources could be managed either internally by the EVOp team or by external parties.

Beside providing information concerning different environmental topics, the portal should give users the capability to utilise the available datasets in a meaningful way. This entails running datasets through a selection of processing tools, including predictive models and workflows, that are executed by users on demand. This presents requirements of inclusiveness (of different data types and model environments) and elasticity.

The resulting set of requirements could be distilled as:

- *Flexibility*: The infrastructure should have fundamental support for assets of varied types and sources.
- *Scalability*: The portal should enable rapid access to additional resources to cater to user and asset management demands.
- *Interoperability*: The portal should rely on open standards to ensure interoperability in order to access external resources, to compose new services, etc.
- *Transparency*: The portal should abstract away as much implementation details from the user as possible.
- *Usability*: The portal should deliver services that are useful, easy, and intuitive to use by different communities.

B. Architecture

In light of the above requirements, we constructed the EVOp infrastructure using cloud computing technologies. These, along with other associated technologies, offer a number of advantages that translate to low operational costs at the infrastructure level and high flexibility at the application level.

A pillar of cloud architectures is the concept of 'everything as a service' (XaaS), which stemmed from service-oriented programming, where all resources are identifiable via a uniform view. This offers versatile resource management, allowing EVOp to support data assets of different origins: from in situ gauging stations, warehoused data stores, user provided, and external sources. The management of such resources is also transparent, *i.e.* details of where and how the data are held are hidden from the user without affecting their experience. Moreover, this design concept hides away implementation details from other system elements which facilitates easier sharing between scientists (*e.g.* workflows) which in turn promotes a culture of collaboration. It also allows for the data to be used in models and simulations without necessarily giving it away to the users, thus avoiding some of the delicate aspects of data ownership. This concept is extended to include tangible resources in Infrastructure as a Service (IaaS), where hardware arrangements could be obtained as and when required. This provisioning of hardware resources as a utility introduces elasticity, whereby the EVOp infrastructure is allowed to scale to meet user demand and maintain a minimum quality of service. This is made possible by handing over some of the key distributed systems management functions to a cloud provider. The return is assured levels of reliability, performance, and security which lets us focus on solving domain-specific problems. Furthermore, virtualisation technologies are used to dynamically deliver environments that are specifically customised to execute user workflows (*e.g.* GIS models).

IV. Implementation

The section provides technical details of three main parts of the implementation: *a*) the underlying infrastructure, *b*) the adopted web service standards, and *c*) the web technologies

employed by the user front end. The section ends with an overview of how the infrastructure looks and functions having been developed using the above technologies.

A. Cloud Infrastructure

An early realisation in the lifetime of the project was concerning the need to rise above the shortcomings of one cloud provisioning solution and to have means by which we can control upfront and running expenditures. We thus adopted a hybrid infrastructure comprised of both private and public cloud resources. The private cloud is hosted in Anonymous University and is operated by us using OpenStack. The public cloud resources are provided by Amazon Web Services (AWS). The pairing of OpenStack and AWS is a common one in the cloud computing world; AWS is arguably the most mature and feature rich public IaaS provider [37], and OpenStack is backed by many as the de facto open source alternative to the core AWS products, *i.e.* EC2 (utility computing) and S3 (storage service). This makes it possible, at least in theory, to use the same virtual machine (VM) images to start instances in either cloud. In an effort to promote portability and to avoid being tied in to one provider, we decided to use the cross-cloud library jclouds [38]. This open source software provides abstractions across many of the widely used cloud solutions.

B. Standard Web Services

The EVOp infrastructure is an ecosystem inhabited by different elements, such as datasets and analysis processes. Each of these inhabitants, according to the XaaS architectural modus operandi, is considered a system resource that is made accessible via a web service interface. This offers a great deal in terms of abstraction, transparency and delegation, and facilitates easy usage and management. One way to describe the benefits is to think about internal and external access to resources. Internal access, where management is involved, is vastly improved as all system resources are accessible in a uniform machine-readable manner. This not only simplifies housekeeping tasks but also enables advanced management tasks to improve availability, fault recovery, etc. Externally, where access to resources does *not* involve management, is similarly transformed by the ability to consume a resource at a convenient level without worrying about more details than necessary. Moreover, separation of concerns brought on by abstraction is in itself a significant benefit.

In line with this XaaS architectural ethos, all EVOp web services interfaces are of a uniform view, designed according to the Representational State Transfer (REST) architectural principles, except where current standards do not accommodate REST (examples will be given shortly). REST is a resource-oriented architectural style, as opposed to the transaction-oriented SOAP web services. The latter require high communication and operation overheads in order to maintain transaction state on the server, *i.e.* the machine hosting a web service to process a request. This has a knock on effect on performance, scalability, and fault tolerance of these services. In contrast, RESTful web services remain completely stateless with all data required to transition between different states being included in the service request. This promotes loosely coupled services. In other words, adopting RESTful services draws a clear line between the client and the server.

Such loose coupling has a huge knock-on effect on infrastructure scalability and manageability. As application state is not maintained by the server, there is much less load on it. The client, however, can invoke the server as much as required to change the state throughout the steps of a scientific experiment, the different runs of a simulation, etc. Moreover, this greatly simplifies complicated infrastructure management tasks such as load balancing and failure recovery. In order to optimise performance, end user requests are routed to any available hosted service regardless of previous interactions. Similarly, failed VMs are easily replaced. Hence, service migration is graceful and requires no advance resource reservation, shared block devices, or any similar techniques.

Consequently, we find the RESTful approach to architect web services very suitable for different types of scientific applications, especially embarrassingly parallel ones such as Monte Carlo simulations, parameter sweeps, uncertainty analysis, etc. where there is no need to share state between different transactions.

Nonetheless, adopting the RESTful style for web services was not without difficulties. The main stumbling block was that most of the standards in the geospatial analysis community are specified using SOAP services. Conforming to these standards is of high priority to us for all model implementations. The ones we adopt are Web Processing Service (WPS) and Sensor Observation Service (SOS)[2]. This meant not having a completely RESTful architecture in order to enable easy integration of models and composing more sophisticated OGC-compliant web services. We find this a fair compromise for the time being.

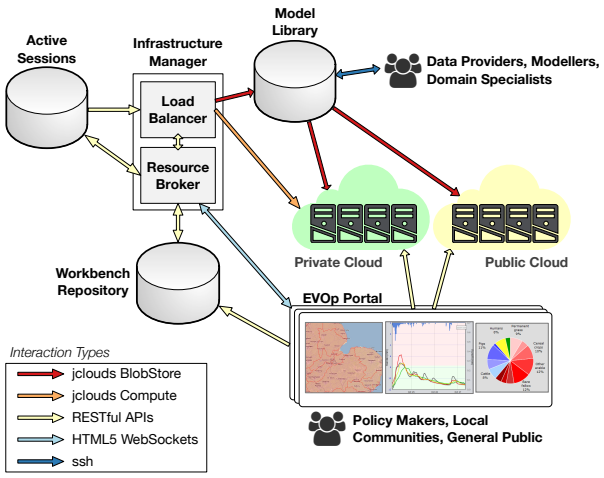

Fig. 1. EVOp infrastructure components and data flows

[2]More information is available in [39] and [40].

C. Web 2.0 Technologies

Service delivery is a commonly misjudged portion of digital science projects. Traditionally, the importance of data visualisation has been underestimated which lead to services that are inaccessible to many scientists, let alone users from outside the domain. Sometimes, aspects such as user experience are overemphasised whilst focusing less attention to the added value the project is supposed to offer.

In EVOp, we believe in the importance of high quality data visualisation, intuitive and familiar user interaction, and low entry barrier user interfaces. Such features have significant influence on user adoption, and can thus make or break the user's perception of the value of an offered service especially over the web. Furthermore, portal users, including scientists, are not expected to be IT experts and hence would rather not tussle with compatibility issues, security restrictions, and other hurdles that were detrimental to the uptake of previous e-research efforts. We thus made it a priority to involve our target users: involve them in the beginning and continuously. Nonetheless, the feedback from the different stakeholders were also used to ascertain a balance between ease of use and high added value for all users.

In order to achieve such balance, we developed a set of web-based tools that enable users to explore data sources, run models in the cloud, visualise the output of model runs, and assist in comprehending that output. Delivering the EVOp services through a web-based portal is key for our purposes. The services are universally accessible by all target groups using a modern web browser. Users do not need to install special software. The users could use any web-enabled device from any location. This enables access using mobile devices, which is particularly convenient for field-based and travelling researchers.

The portal hosts a number of models that require different inputs and produce different outputs. For interacting with each model (*i.e.* manipulating the input and parameters) and to help bring out relationships or patterns within the data, a bespoke web interface was developed to suit the particular factors in question. This is achieved using dynamic HTML, native HTML5 page elements, HTML5 WebSockets message exchange, and browser scripting using publicly available Javascript libraries. These libraries, which include jQuery and qTip2, were chosen on the basis of providing intuitive user interaction and high customisability, such as the ability to tie in with dynamically loaded content. Geospatial data are visualised using interactive layers superimposed over maps. The Google Maps API is used for this purpose due to its wealth in mapping data, customisation features, and the familiarity that many of the potential users have with it. Geospatial layers are made interactive in order to include time series graphs and other sorts of visualisations over specific map locations. This was accomplished using the Flot Javascript library. Flot was selected due to its programming flexibility, intuitive interactivity, and support of automatic data updates using AJAX.

D. Infrastructure Summary

We now summarise the infrastructure that culminated using the above technologies. Figure 1 depicts the data flow between the different infrastructure components which enable seamless user interaction.

The Model Library (ML) is populated by domain specialists (*e.g.* hydrologists) in liaison with data providers. For enduser-facing services, the process starts with offline calibration and testing of a model against a certain dataset (*e.g.* TOPMODEL on the rainfall data of the Eden catchment in the north west of England). The outcome of this process is a VM *image* optimised to run a fine tuned set of models that are exposed as web services and equipped with all required data. This streamlined execution bundle is then stored in the ML to be instantiated upon demand. An image could be updated to include more historical data or to adjust the implementation of a model in some way. The alternative path is to use a generic image from the ML to serve as a model incubator. Using these images, more experimental models are installed and exposed as web services deployed according to the OGC WPS standard. They can then be calibrated for different modelling scenarios and using different datasets. This has some effect on execution performance when compared to a streamlined execution unit, but is a useful testing ground for modelling scientists.

Once a user navigates to one of the modelling widgets (an example of these will be provided in the following section), a connection is created with the Resource Broker (RB) module of the Infrastructure Manager. RB responds with an address of a cloud instance that is suitable for the type of computation required, along with some session information. This communication is done in the background using HTML5 WebSockets which facilitates event-based asynchronous duplex communication without the need for periodic polling or streaming, which are costly and inefficient modes of background browser traffic exchange. This reduces network overhead and browser memory usage, and enables RB to manipulate the user session more efficiently. For instance, this is used to update the set of active sessions in order to balance load by sensing when user sessions end. It also allows RB to push any session updates to the user's browser, such as in the case of migrating the user to a new cloud instance.

The Load Balancer (LB) monitors the health status of running instances with two objectives: minimise costs and maintain instance responsiveness. To minimise cost, user requests are served by default using private instances. Upon saturation of private cloud resources, LB initiates *cloudbursting* mode where public cloud instances are used beside private ones. This is reversed upon detecting underuse, migrating users back to use private instances. For the latter objective, *i.e.* maintaining responsiveness, instance statistics are observed, namely CPU utilisation, disk reads and writes, and network usage. Degradation in these metrics, such as sustained high CPU utilisation or zero outbound network usage whilst receiving inbound traffic, triggers LB into starting a new instance and redirecting users that were being served by the seemingly malfunctioning

instance to the newly created one. LB also monitors the state of active user sessions and redistributes users on running cloud instances accordingly. RB is used to push updated session information in order to redirect user calls.

V. Use Case Development

Requirement collection for the EVOp portal was not a trivial task. This is due to the novelty of the application and the need to create highly customised web tools. Furthermore, the EVOp team included researchers from different backgrounds (environmental, computer and social science) [41]. Frequent meetings were required to discern any changes that need to be done at an early phase. In this section we describe the methodology used to build added value on top of the EVOp infrastructure, and we detail one of the use cases developed within the project.

A. Methodology

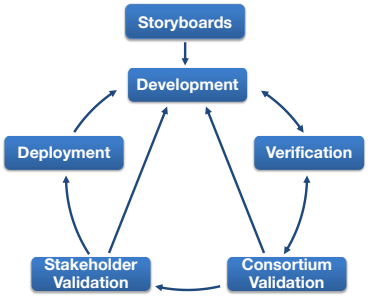

Fig. 2. The Test-Driven Development Cycle.

The EVOp development process is an iterative and interactive process that is heavily depended on *incremental development and frequent verification*. We used the Agile-based test-driven development methodology (see Figure 2). A *storyboard*, *i.e.* a stepped illustration of a fully defined user scenario, was outlined by partner domain specialists (referred to as the *storyboard owners*). The detailed visual steps provided by storyboards allowed us to collect not just the core functional requirements but also well-defined usage contexts, user interface layout and interaction, and full-length experiential user flow. Based on these, prototypes were developed and iteratively improved and built upon following processes of verification and validation. *Verification* is the process of checking than an artefact developed (here, the virtual observatory) is technically correct and addresses the requirements laid out in the storyboard. This involves unit tests to ensure good and failsafe execution of software components, and integration tests to examine full features that span several components. This technical verification is one of the main communication practices between the development team and the storyboard owners, occurring at the end of each development cycle which usually takes between a day to a week depending on the size of the task at hand. In contrast, *validation* relates to confirming that an artefact developed does indeed serve the intended end-users the way the storyboard describes. The main objective here is not technical soundness as much as it is utility (*i.e.* how useful it is) and usability (*i.e.* how convenient and intuitive it is to use). This dialogue goes in both directions between researchers and stakeholders (Figure 3), and is centered around the issue addressed by the proposed tool and its general context and implications. This sort of participatory engagement is reported to be the most successful [42]. Validation cycles are longer than verification ones. They are carried out within the wider project consortium (every 1-2 months or so) and with the stakeholders through evaluation workshops (once or twice a year).

Fig. 3. Discussions at a public stakeholder meeting.

We now provide detail on one of the local storyboards. Details of another storyboard that was developed are available in [43].

B. Local Flooding Storyboard

The local scale EVOp exemplar, which henceforth will be referred to as the Local EVOp Flooding Tool (LEFT), was developed with stakeholders from three largely rural catchments in England, Scotland and Wales. These catchments were selected owing to a number of environmental issues, previous track record of engagement through the research teams and deployments of in situ environmental sensors. Workshops were held in each catchment to come up with an example tool (using the storyboard approach previously mentioned) that is owned and defined by the local stakeholders.

Flooding was found to be an issue in all three catchments; Morland in Cumbria (England), Tarland in Aberdeenshire (Scotland), and Machynlleth in Powys (Wales) all had suffered from floods within the past five years. Workshop groups mainly consisted of villagers, farmers and catchment managers. Some workshops also included other stakeholders such as the insurance industry. Villagers were interested in further information to help them react to a flood, but also had an interest in what caused the flooding. Farmers wanted to understand if their farming practices increased the risk of flooding, and options that could reduce that risk (working with the catchment managers and associated environmental stewardship scheme funding packages). There was a desire from local catchment stakeholders to have live access to rainfall and river level sensors in their catchments and to look at future flooding scenarios.

A storyboard was accordingly drawn to show the process a user would go through to use LEFT for gaining knowledge

from local datasets and then being able to use these datasets to feed cloud-based models and visualisation systems.

The storyboard would start with key purposes that stakeholders identified during the workshops, such as "how do I decide when my property is at risk of flooding?". It would then start to be substantiated by identifying a user's journey through the tool: starting with selecting the feature they desire and any associated data (based on location, for instance), the display and layout of results, and any subsequent interactions to further explore or interpret the outcomes. From this, a set of basic requirements are captured that will be later used to ensure successful tool delivery. The tool was developed iteratively during the project to reflect the needs, interests and capabilities of the different stakeholders.

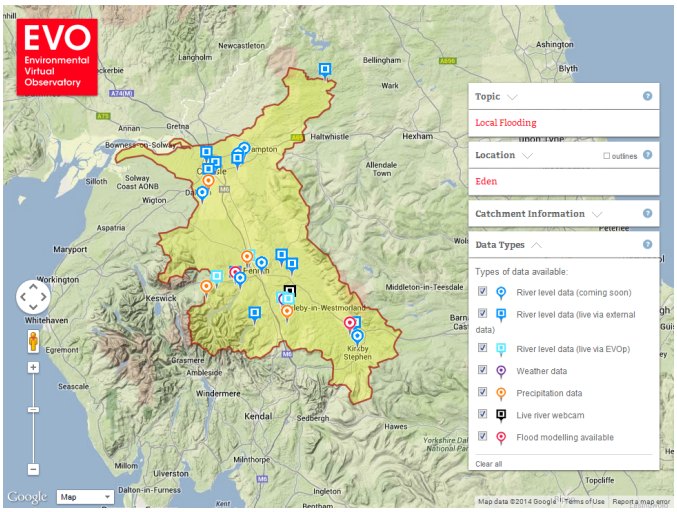

Fig. 4. The home panel for LEFT. Users can select certain data types to view, then click through for deeper exploration.

First, an interactive mapping backdrop was developed as the LEFT landing page (Figure 4), on top of which datasets (both static and live) and other assets (such as webcam feeds) were overlaid on the map as geotagged markers. This provides users with the ability to instantly identify assets of interest based on geographical location. For most users, this entails exploring their local catchment and gathering information from various data sources. Google Maps API was used for this part of the work (after considering alternatives like OpenLayers and Bing) due to its wealth in data, features, and familiarity for target users.

The interactive nature of the geospatial layers provides the ability to reveal new interfaces to the user. Such interfaces vary based on the type of asset. The stakeholder workshops highlighted high interest in being able to visualise live data. Based on this information, we developed visualisation widgets that assist users with interpreting the data they were looking at. For example, live data (such as those fed by in situ sensors) were presented as time series graphs. In some instances, we combined more than one data source to present bespoke, multimodal visualisations and user interaction facilities. In one instance, different sensors were used to plot water temperature

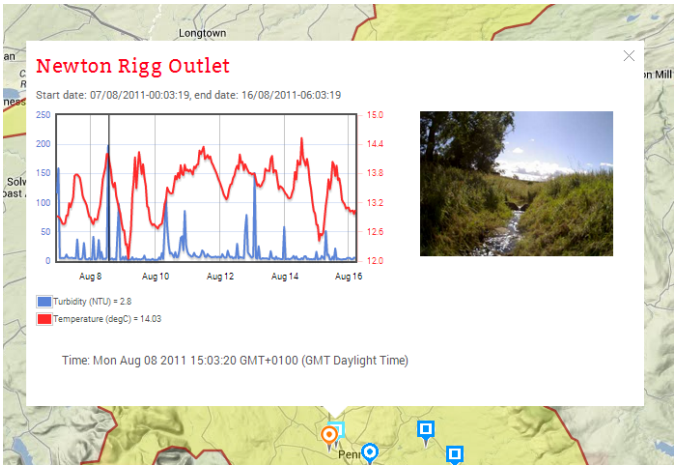

Fig. 5. An example of using data mashups for multimodal visualisation. The widget in this screenshot uses synchronised data from in-stream water sensors and stream-side webcams.

and turbidity linked with the corresponding webcam image taken roughly at the same time (Figure 5).

Having developed this framework where assets are laid on a map and widgets opened upon interaction, we dedicated a lot of effort to develop the LEFT modelling widget.

The widget is an example of a bespoke user interface that offers rich interaction functionality to leverage the computational power provided by the cloud. The widget was built by a team of domain experts, hydrological modellers, web designers and developers following the TDD methodology discussed in section V-A. This widget, shown in Figure 6, contains a number of different options for the user to choose from: the datasets available at this location, the hydrologic model to use, and the model's parameters. In order to alleviate some of the complexity of adjusting the model, the user could also select from four land use and management change scenarios. These scenarios, developed with stakeholders, were used to illustrate how changes to land use and land management practices are likely to impact flood risk at the catchment outlet. The widget also shows detailed textual and animated help to provide background information and educate the user about the model and scenarios.

For this use case, two hydrological models were deployed in the cloud to test the conceptual land use scenarios: TOPMODEL [44], an established quasi-physical processed based model, and the multi-model ensemble FUSE [45]. Model calibration was carried out offline to ensure that input data and parameters were in the correct format and the model could adequately reproduce observed discharge at the outlet of the catchment. Once all selections are done by the user, the model is run instantly on demand in the cloud and the returned results are rendered as a hydrograph plotted using Flot (Figure 6). Changes in the flood hydrograph could be examined by running the model under different predefined scenarios or specific parameter combinations to allow comparison between model runs and provide an understanding of the stream's response at

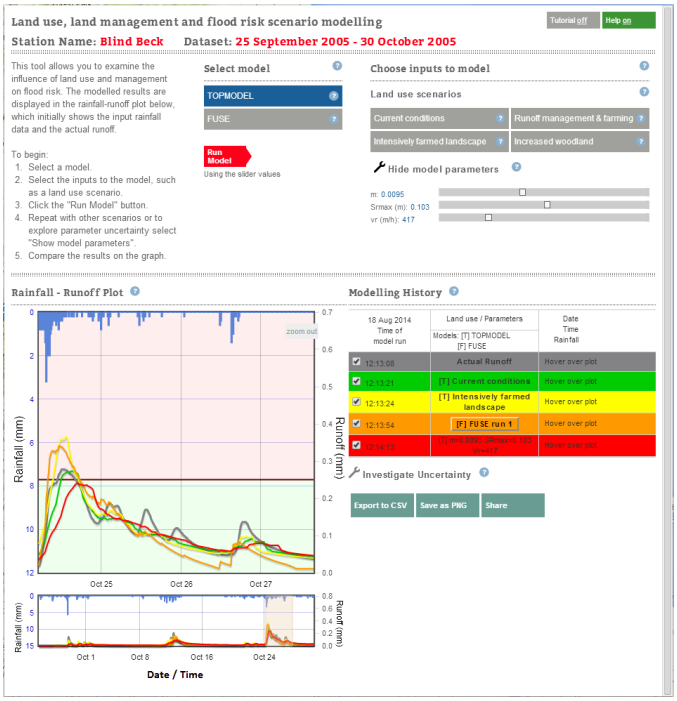

Fig. 6. The LEFT modelling widget provides: location-specific data streams, answers to exploratory scenarios using different pre-calibrated models, help on comparison of different model outputs, more fine-tuned model calibration for domain experts, and current policy guidance on flooding thresholds.

the catchment outlet to changes to land use and management. The predefined scenarios can be selected using the buttons in the top-right of Figure 6. These are especially designed for policy makers and the general public. In contrast, users who are more familiar with the models could explore model parameter sensitivity through HTML sliders included in the widget just below the scenario selection buttons. The sliders default to the settings for each scenario to allow users to compare how changes to these values alter the model outputs. This is an example of a multi-faceted interface that enables the widget to cater to the needs of different user groups.

The workshop attendees in particular identified the huge benefits from using a 'cloud' platform, *e.g.* accessing data that is not normally available to them, as well as using tools that they do not usually have on their own desktops. This allowed land owners to explore future land use scenarios, allowing them to conceptually understand the associated flood risk. There was universal agreement that the EVOp potentially provides a tool that holds both educational and scientific value. The last evaluation workshops saw enthusiasm from stakeholders to develop new tools based on new storyboards (*e.g.* what would be the impact of this scenario on catchment water quality).

VI. Critical Reflection

In this section, we reflect on the EVOp infrastructure and development experience and discuss it in light of the requirements set forth in subsection III-A. These are: flexibility, scalability, interoperability, transparency, and usability.

The service-oriented and cloud-based aspects of the EVOp architecture lend much **flexibility** and **transparency** at different levels. To illustrate what the former affords, we break down resources into *soft* and *hard* assets. For *soft* assets (such as data sets and models), XaaS hides from the user details of where resources are held and how they are managed. Such abstraction translates to a better user experience as complicated issues are offloaded allowing the users to focus on solving domain-specific problems. By the same token, XaaS also enables delegation which allows models to process datasets without necessarily giving them away, avoiding some of the thorny issues of data ownership. Moreover, XaaS enables versatile management, allowing EVOp to support data assets of different origins: in situ gauging stations, warehoused data stores, and external sources. It also promotes a mashup culture where resources can be shared, reused, and combined to create more sophisticated assets. Obviously, this involves dealing with resources of varied types and sources and making them addressable and manageable via APIs. For *hard* or tangible assets (where XaaS is essentially IaaS), hardware resources could be arranged as and when required. This provisioning of hardware resources as a utility abstracts away the complexities of distribution, reliability and availability. It also offers **scalability** through elasticity, whereby the infrastructure is allowed to scale to meet user demand and maintain an acceptable quality of service. Consider for instance uncertainty analysis where a model is repeatedly executed using ranges of values for input parameters in order to compensate for any sources of error in how well the data represents the real variables, *e.g.* topographical representation of a river catchment. This requires substantially more computational resources than a single execution. By providing such resources on demand, IaaS presents such a great advantage when compared to both grid and cluster computing where usage quotas are a common hindrance for resource-intensive computations. Another example is flash crowds, *i.e.* extremely large and unexpected number of portal users. IaaS enables us to manage such events with great ease and maintenance of high Quality of Service (QoS). Several additional techniques could be used here to ensure high QoS, such as prefetching data records and preemptively bootstrapping cloud instances as soon as a user visits the portal. This results in additional operational overheads, but is usually not significant enough in comparison to the gain in user experience.

In another regard, IaaS provides **flexibility** through virtualisation. This technology allows VMs to be furnished according to specific requirements. For EVOp, the team developed four different models each of which was used different programming languages and software requirements. For instance, TOPMODEL was developed as a library (package) for the R data analysis platform then deployed as an WPS endpoint using the Python-based implementation PyWPS [46], [47] and the R_Python connector RPy2 [48]; another model (export coefficient model) was implemented as a combination of PHP on an Apache server and JavaScript on the client side. This ability to produce any software environment with practically

no restrictions is a great enabler especially when considering the cross-disciplinary aspirations of a virtual observatory where different developers from diverse backgrounds and working conventions are to be expected. One thing to plan for is the choice of the technologies to prepare and deploy VMs. For virtual appliances that are managed as full (*pre-baked*) machines images, the choice of image format is crucial to ensure the ability to operate across different IaaS infrastructures. Additional technologies could also help in this regard, most worthy of note are Configuration Management Tools (CMTs) such as Chef and Puppet which allow the definition of an infrastructure of VMs as code. This is quite useful for lightweight and automated deployment configuration.

Using the jclouds cross-cloud API was vital to maintain **infrastructural interoperability** [49], [50]. This proved quite useful when the infrastructure provider or its utilisation model needs to be adjusted. For example, changing the scheduling policy from 'all computations on private cloud until saturation' to something more selective such as 'streamlined models to AWS and experimental ones to the private cloud'. More importantly, though, it is necessary to have a federated open approach as it is impossible to commit the national and international ES community to any one commercial provider. As for semantic interoperability, we were attentive to the need of complying to the open standards already established in the geospatial modelling community. Adhering to the OGC standards (namely Web Processing Service) to specify exactly how web service inputs and outputs should be required additional effort from the domain specialists implementing the web services. However, adhering to this ensured future compatibility; *i.e.* the ability to easily plug different modelling services without the need to neither re-align the input data nor tweak the tools used for visualising the output.

Usability is one of key concerns, and hence we were constantly driven by maintaining optimal service delivery and low entry barriers. As described in Section V, we dedicated significant effort to continuously engage with different stakeholder groups and use their feedback to guide the following iterations of the work [51]. The feedback from the stakeholder workshops were supportive of our approach: more than 75% of users found the tool to be both useful and easy to use with a good look and feel [52]. One aspect brought up by the stakeholders during the workshops is the lack of presentation of uncertainty bounds. The tool output as they are provide a conceptual understanding of the different scenarios, but uncertainty could easily be explored beyond the project using the same technical setup described in Section IV and using the 'percentile' feature in Flot. Embracing a test-driven design through the use of storyboards defined from the perspective of potential users was instrumental. It allowed us to launch the development cycle with clearly defined targets based on the expectations and capabilities of the target user groups. This along with an Agile approach helped us avoid drawn-out development cycles that are not checked on a regular basis. This is a common trap that usually results in a product that is not useful for the target users (*i.e.* does not help answering their questions) no matter how easy to use it may be. We found this combination to be extremely effective for dealing with the mix of divergent storyboards and diverse domain experts. An important lesson we would like to share here, referring back to Figure 2, is to keep the verification loop rather short and rapid in comparison to the validation (internal feedback) loop. One way of achieving this is to always involve storyboard owners in the prototyping process even before verification milestones are due [53].

VII. Lessons Learned

Through building the EVOp portal as a *virtual research environment* [54], we were able to successfully lower the barrier for various user groups by providing universal access to models and data, means of executing models and manipulating data, and interpreting and sharing results. Outside of the technical lessons already discussed, a number of lessons were learned through the EVOp project, which we believe would be valuable to the wider distributed computing community.

Only use what is needed

By now, there is a rich tradition of developing cloud applications and complex distributed systems [55], specifically those handling predictive modelling and similar workloads. However, not all available tools and approaches are suitable, and forcing the latest SE trend would not necessarily obtain positive results. An example from EVOp is the use of collaborative research tools *e.g.* Jupyter [56]. Such tools are extremely powerful, but they do not necessarily create a productive environment for scientists from different backgrounds and subfields (see the following lesson). Moreover, they are not readily accessible to non-experts, lamenting the barrier between different user groups. Our experience shows success using a simple architecture based on long-tested methods, such as an XaaS strategy and employment of Web 2.0 tools, and instead focussing our collective creative energy on developing participatory design processes (see Section V-A).

Disciplinary boundaries persist

Building software systems does not solve all issues. A major challenge we faced in EVOp is crossing disciplinary boundaries, which predominantly persist in ES and are manifested in a number of different ways such as terminology, data collection and processing methods, data formats, and of course scientific goals. We found that storyboarding helps a lot in reaching some form of consensus on high-level goals and methods. However, what helped further were: frequent verification meetings, and creating enough flexibility in the system to support different working methods.

Efficient communication

Although the TDD cycle was a foreign concept to most stakeholders from non-SE backgrounds, it was readily adopted by all – expert software engineers and non-programming stakeholders alike – once explained, confirming similar findings in the literature (*cf.* [57]). One of the benefits of following the

cycle across the whole project is making sure that software engineers do not dominate the process, and that different stakeholders are being made aware of progress in other areas through a familiar process that is predominantly social [58].

Create means to both educate and learn from stakeholders

People are extremely interested in ES; from local and central government officials and agencies, water and energy sectors, insurance industry, to the wider public. Many of these various groups seek *simple answers to complex questions*. Stakeholder awareness has already been highlighted in the literature [59], [58], but from our experience this is not sufficient to ensure active engagement. A certain degree of education is required beyond mere awareness (Figure 7). Our development cycles were much more productive after the first two stakeholder meetings where the intricacies of the used prediction models and data were explained and discussed in detail. This lead to a better understanding on all sides and ultimately to widening the circle of engagement around the different ES themes addressed in EVOp. This is of crucial importance as empowering individuals is a proven path towards positive action [60], [61].

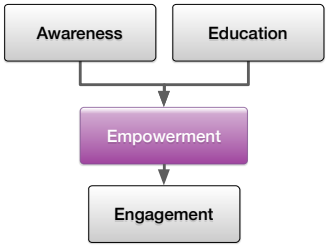

Fig. 7. Awareness is not enough to ensure engagement.

VIII. Conclusions & Future Work

In face of increasingly difficult environmental challenges, the scientific community is looking for collaborative information infrastructures that facilitate collecting and processing data, chaining and executing models, and visual interpretation of results for different audiences. Cloud computing along with a number of associated technologies offer great potential in realising such infrastructures. This article presented the EVOp project and its portal, which provides access to data and tools that help different stakeholders in engaging with pressing environmental issues. We specifically focused on the architectural design, Agile test-driven development process, and enabling technologies that underpin the virtual observatory. These are of relevance to others working on similar integrated tools both within the wider scope of ES and beyond.

EVOp's underlying infrastructure is a tailored hybrid one of owned and leased cloud resources. This infrastructure not only does the computational heavy-lifting, but also offers the flexibility to integrate and support varied resources including cloud-hosted data sets and legacy web services. The system also has a low entry barrier compared to previous systems: modellers are able to create execution units using a development environment of their choice, and users are able to execute them from a simple web app with no expectation of user device. The EVOp architecture was fashioned to focus on assets rather than on transactions, thus enabling efficient resource management and easy fault mitigation.

More importantly, we shared lessons learned to help others who are also working to use cloud tools for widening stakeholder engagement around public debates such as that surrounding ES. Making initial progress was definitely not easy, as each stakeholder was pulling in a different direction. This is not least due to the confusion created by a divergent atmosphere where people working on the same problem could be doing so in isolation without necessarily sharing data or knowledge. Having a well-detailed and well-managed test-driven development cycles, with a specific focus on communication and education, helped in pulling efforts together.

EVOp served as a demonstrator built to meet diverse and variable resource requirements. There already are a number of efforts building on the wealth of experience garnered. First, various efforts are under way to invest in infrastructures for supporting different uses in ES. Second, we are planning to expand the spectrum of tools by supporting domain specialists to expose more models as web services. More importantly, we are looking to increase the room for customisation by supporting workflow composition. So far, we have been building web based prototypes based on very specific use cases outlined by storyboards. A workflow is a conglomerate scientific process composed of a directed acyclic graph of basic execution units (*e.g.* executables, scripts, web services, etc.). Workflows allow 'advanced' users (*i.e.* domain specialists from the scientific or governmental communities) to create complex experiments that can be easily tweaked and replayed, offering reproducibility and traceability.

Acknowledgment

This work was supported by NERC UK under grant reference NE/I002200/1, '*Pilot Virtual Observatory*'. Thanks go to the rest of the project team: Lucy Ball (Centre for Ecology & Hydrology – CEH), Keith J. Beven (Lancaster University), John Bloomfield (BGS), Paul Brewer (University of Aberystwyth), Wouter Buytaert (Imperial College London), Lucy Cullen (CEH), Julie Delve (CEH), Bridget Emmett (CEH), Jim Freer (Bristol University), Sheila Greene (CEH), Philip M. Haygarth (Lancaster University), Penny Johnes (Bristol University), Jane Lewis (University of Reading), Christopher J. A. Macleod (James Hutton Institute), Mark G. Maklin (Aberystwyth University), Keith Marshall (James Hutton Institute), Adrian McDonald (Leeds University), Nick Odoni (Bristol University), Paul F. Quinn (Newcastle University), Sim M. Reaney (Durham University), Gwyn Rees (CEH), Marc Stutter (James Hutton Institute), Doerthe Tetzlaff (Aberdeen University), Nicola Thomas (Aberystwyth University), John W. Watkins (CEH), and Bronwen Williams (CEH). The work was also partially supported by EPSRC UK under grant reference EP/N027736/1, '*Models in the Cloud: Generative Software Frameworks to Support the Execution of Environmental Models in the Cloud*'.

REFERENCES

[1] A. J. G. Hey and A. E. Trefethen, "The data deluge: An e-science perspective," in *Grid Computing - Making the Global Infrastructure a Reality*, F. Berman, G. C. Fox, and A. J. G. Hey, Eds. Wiley and Sons, 2003, pp. 809–824, Chapter 36.

[2] R. G. Baraniuk, "More is less: Signal processing and the data deluge," *Science*, vol. 331, no. 6018, pp. 717–719, 2011.

[3] W. Buytaert, S. Baez, M. Bustamante, and A. Dewulf, "Web-based environmental simulation: Bridging the gap between scientific modeling and decision-making," *Environmental Science & Technology*, vol. 46, no. 4, pp. 1971–1976, 2012.

[4] E. P. White, E. Baldridge, Z. T. Brym, K. J. Locey, D. J. McGlinn, and S. R. Supp, "Nine simple ways to make it easier to (re) use your data," *Ideas in Ecology and Evolution*, vol. 6, no. 2, 2013.

[5] B. Nelson, "Data sharing: Empty archives," *Nature*, vol. 461, pp. 160–163, Sep 2009.

[6] C. Tenopir, S. Allard, K. Douglass, A. U. Aydinoglu, L. Wu, E. Read, M. Manoff, and M. Frame, "Data sharing by scientists: Practices and perceptions," *PLoS ONE*, vol. 6, no. 6, p. e21101, 06 2011.

[7] C. J. Volk, Y. Lucero, and K. Barnas, "Why is data sharing in collaborative natural resource efforts so hard and what can we do to improve it?" *Environmental Management*, vol. 53, no. 5, pp. 883–893, 2014.

[8] T. Karpouzoglou, Z. Zulkafli, S. Grainger, A. Dewulf, W. Buytaert, and D. M. Hannah, "Environmental virtual observatories (evos): prospects for knowledge co-creation and resilience in the information age," *Current Opinion in Environmental Sustainability*, vol. 18, pp. 40–48, 2016, sustainability governance and transformation.

[9] P. Beaumont, P. A. Longley, and D. J. Maguire, "Geographic information portals – a UK perspective," *Computers, Environment and Urban Systems*, vol. 29, no. 1, pp. 49–69, 2005.

[10] R. R. Hernandez, M. S. Mayernik, M. L. Murphy-Mariscal, and M. F. Allen, "Advanced technologies and data management practices in environmental science: Lessons from academia," *BioScience*, vol. 62, no. 12, pp. 1067–1076, 2012.

[11] S. E. Hampton, C. A. Strasser, J. J. Tewksbury, W. K. Gram, A. E. Budden, A. L. Batcheller, C. S. Duke, and J. H. Porter, "Big data and the future of ecology," *Frontiers in Ecology and the Environment*, vol. 11, no. 3, pp. 156–162, 2013.

[12] B. Pernici, M. Aiello, J. vom Brocke, B. Donnellan, E. Gelenbe, and M. Kretsis, "What IS can do for environmental sustainability: A report from CAiSE'11 panel on green and sustainable IS," *CAIS*, vol. 30, 2012.

[13] R. Verdecchia, F. Ricchiuti, A. Hankel, P. Lago, and G. Procaccianti, "Green ICT research and challenges," in *Advances and New Trends in Environmental Informatics*. Springer, 2017, pp. 37–48.

[14] V. Thomas, C. Remy, M. Hazas, and O. Bates, "HCI and environmental public policy: Opportunities for engagement," in *CHI*. ACM, 2017, pp. 6986–6992.

[15] W. A. Simm, F. Samreen, R. Bassett, M. A. Ferrario, G. Blair, J. Whittle, and P. J. Young, "SE in ES: Opportunities for software engineering and cloud computing in environmental science," in *ICSE Software Engineering in Society Track*. ACM, 2018, pp. 61–70.

[16] Q. Gu, P. Lago, and S. Potenza, "Aligning economic impact with environmental benefits: A green strategy model," in *Workshop on Green and Sustainable Software*. IEEE Press, 2012, pp. 62–68.

[17] M. Wilkinson, K. Beven, P. Brewer, Y. Elkhatib, A. Gemmell, P. Haygarth, E. Mackay, M. Macklin, K. Marshall, P. Quinn *et al.*, "The Environmental Virtual Observatory (EVO) local exemplar: A cloud based local landscape learning visualisation tool for communicating flood risk to catchment stakeholders," in *The European Geosciences Union General Assembly Conference*, vol. 15, 2013, p. 11592.

[18] E. Osuteye, C. Johnson, and D. Brown, "The data gap: An analysis of data availability on disaster losses in sub-Saharan African cities," *International Journal of Disaster Risk Reduction*, vol. 26, pp. 24–33, 2017.

[19] A. I. J. M. van Dijk, H. E. Beck, R. S. Crosbie, R. A. M. de Jeu, Y. Y. Liu, G. M. Podger, B. Timbal, and N. R. Viney, "The millennium drought in southeast Australia (2001–2009): Natural and human causes and implications for water resources, ecosystems, economy, and society," *Water Resources Research*, vol. 49, no. 2, pp. 1040–1057, 2013.

[20] C. Tortajada, M. J. Kastner, J. Buurman, and A. K. Biswas, "The California drought: Coping responses and resilience building," *Environmental Science & Policy*, vol. 78, no. Supplement C, pp. 97–113, 2017.

[21] G. S. Blair and Y. Elkhatib, "A Cloud-based Virtual Observatory for Environmental Science," *OpenWater Symposium*, p. 102, Apr 2011.

[22] P. Lago, R. Kazman, N. Meyer, M. Morisio, H. A. Müller, and F. Paulisch, "Exploring initial challenges for green software engineering: Summary of the first greens workshop," *SIGSOFT Software Engineering Notes*, vol. 38, no. 1, pp. 31–33, Jan 2013.

[23] R. Chitchyan, I. Groher, and J. Noppen, "Uncovering sustainability concerns in software product lines," *Journal of Software: Evolution and Process*, vol. 29, no. 2, 2017.

[24] G. A. García-Mireles, M. A. Moraga, F. García, C. Calero, and M. Piattini, "Interactions between environmental sustainability goals and software product quality: a mapping study," *Information and Software Technology*, 2017.

[25] M. M. Gallardo, P. Merino, L. Panizo, and A. Linares, "A practical use of model checking for synthesis: generating a dam controller for flood management," *Software: Practice & Experience*, vol. 41, no. 11, pp. 1329–1347, 2011.

[26] G. Whelan, K. Kim, M. A. Pelton, K. J. Castleton, G. F. Laniak, K. Wolfe, R. Parmar, J. Babendreier, and M. Galvin, "Design of a component-based integrated environmental modeling framework," *Environmental Modelling & Software*, vol. 55, pp. 1–24, 2014.

[27] D. Wang, Y. Xu, P. Thornton, A. King, C. Steed, L. Gu, and J. Schuchart, "A functional test platform for the community land model," *Environmental Modelling & Software*, vol. 55, pp. 25–31, 2014.

[28] S. Dauwe, T. Van Renterghem, D. Botteldooren, and B. Dhoedt, "Multiagent-based data fusion in environmental monitoring networks," *International Journal of Distributed Sensor Networks*, vol. 2012, 2012.

[29] L. N. Leonard, C. Duffy, and G. Bhatt, "Data-intensive Hydrologic Modeling: A Cloud strategy for integrating PIHM, GIS, and Web-Services," in *The American Geophysical Union Fall Meeting*, 2010.

[30] M. A. Bhat, R. M. Shah, and B. Ahmad, "Cloud computing: A solution to geographical information systems," *International Journal on Computer Science & Engineering*, vol. 3, no. 2, 2011.

[31] A. Sun, "Enabling collaborative decision-making in watershed management using cloud-computing services," *Environmental Modelling & Software*, vol. 41, pp. 93–97, 2013.

[32] D. Wu, L. Zhu, X. Xu, S. Sakr, D. Sun, and Q. Lu, "Building pipelines for heterogeneous execution environments for big data processing," *IEEE Software*, vol. 33, no. 2, pp. 60–67, Mar 2016.

[33] I. Demir and W. F. Krajewski, "Towards an integrated flood information system: Centralized data access, analysis, and visualization," *Environmental Modelling & Software*, vol. 50, pp. 77–84, 2013.

[34] J. Leskens, M. Brugnach, A. Hoekstra, and W. Schuurmans, "Why are decisions in flood disaster management so poorly supported by information from flood models?" *Environmental Modelling & Software*, vol. 53, pp. 53–61, 2014.

[35] K. Keahey and M. Parashar, "Enabling on-demand science via cloud computing," *Cloud Computing*, vol. 1, no. 1, pp. 21–27, May 2014.

[36] V. M. Quiroga, I. Popescu, D. P. Solomatine, and L. Bociort, "Cloud and cluster computing in uncertainty analysis of integrated flood models," *Journal of Hydroinformatics*, vol. 15, no. 1, pp. 55–70, 2013.

[37] L. Leong, D. Toombs, B. Gill, G. Petri, and T. Haynes, "Magic Quadrant for Cloud Infrastructure as a Service," http://www.gartner.com/technology/reprints.do?id=1-1IMDMZ5&ct=130819, Gartner, Inc, Tech. Rep. G00251789, Aug 2013.

[38] Apache, "jclouds multi-cloud library," https://jclouds.apache.org/.

[39] C. Vitolo, W. Buytaert, Y. Elkhatib, A. L. Gemmell, S. M. Reaney, and K. Beven, "Cloud-enabled Web Applications for Environmental Modelling," in *The American Geophysical Union Fall Meeting*, Dec 2012.

[40] C. Vitolo, Y. Elkhatib, D. Reusser, C. J. Macleod, and W. Buytaert, "Web technologies for environmental big data," *Environmental Modelling & Software*, vol. 63, no. 0, pp. 185–198, Jan 2015.

[41] B. Emmett, G. Rees, L. Ball, S. Greene, E. Mackay, C. Huntingford, D. Field, M. Bicak, R. J. Gurney, A. Mcdonald, K. Beven, G. S. Blair, J. P. Bloomfield, W. Buytaert, J. Delve, Y. Elkhatib, J. Freer, A. L. Gemmell, P. M. Haygarth, P. J. Johnes, M. Macklin, C. J. A. Macleod, N. Odoni, B. Percy, P. Quinn, S. M. Reaney, M. Stutter, B. Surajbali, D. Tetzlaff, N. Thomas, C. Vitolo, M. E. Wilkinson, and B. Williams, "Environmental Virtual Observatory Pilot Project," Center for Ecology & Hydrology, Tech. Rep., Mar 2014.

[42] A. Voinov and F. Bousquet, "Modelling with stakeholders," *Environmental Modelling & Software*, vol. 25, no. 11, pp. 1268–1281, 2010, thematic Issue - Modelling with Stakeholders.

[43] S. Greene, P. J. Johnes, J. P. Bloomfield, S. M. Reaney, R. S. Lawley, Y. Elkhatib, J. Freer, N. Odoni, C. J. A. Macleod, and B. J. Percy, "A geospatial framework to support integrated biogeochemical modelling in the united kingdom," *Environmental Modelling & Software*, vol. 68, no. 0, pp. 219–232, Jun 2015.

[44] K. Beven, R. Lamb, P. Quinn, R. Romanowicz, J. Freer, V. Singh *et al.*, "Topmodel," *Computer Models of Watershed Hydrology*, pp. 627–668, 1995.

[45] M. P. Clark, A. G. Slater, D. E. Rupp, R. A. Woods, J. A. Vrugt, H. V. Gupta, T. Wagener, and L. E. Hay, "Framework for understanding structural errors (fuse): A modular framework to diagnose differences between hydrological models," *Water Resources Research*, vol. 44, no. 12, 2008.

[46] J. Cepicky, "PyWPS 2.0.0: The presence and the future," *Geoinformatics FCE CTU*, 2007.

[47] J. Cepicky and L. Becchi, "Geospatial processing via internet on remote servers-PyWPS," *OSGeo Journal*, vol. 1, no. 5, pp. 11–17, 2007.

[48] rpy2, "R in python," https://rpy2.bitbucket.io/, 2016.

[49] Y. Elkhatib, "Mapping Cross-Cloud Systems: Challenges and Opportunities," in *Conference on Hot Topics in Cloud Computing*. USENIX Association, Jun 2016, pp. 77–83.

[50] A. Elhabbash, F. Samreen, J. Hadley, and Y. Elkhatib, "Cloud brokerage: A systematic survey," *ACM Computing Surveys*, vol. 51, no. 6, pp. 119:1–119:28, Jan 2019.

[51] A. Jakeman, R. Letcher, and J. Norton, "Ten iterative steps in development and evaluation of environmental models," *Environmental Modelling & Software*, vol. 21, no. 5, pp. 602–614, 2006.

[52] M. E. Wilkinson, E. B. Mackay, P. F. Quinn, M. Stutter, K. J. Beven, C. J. MacLeod, M. G. Macklin, Y. Elkhatib, B. Percy, C. Vitolo, and P. M. Haygarth, "A cloud based tool for knowledge exchange on local scale flood risk," *Journal of Environmental Management*, vol. 161, pp. 38–50, 2015.

[53] M. E. Kragt, B. J. Robson, and C. J. Macleod, "Modellers' roles in structuring integrative research projects," *Environmental Modelling & Software*, vol. 39, pp. 322–330, 2013, thematic Issue on the Future of Integrated Modeling Science and Technology.

[54] M. Barker, S. D. Olabarriaga, N. Wilkins-Diehr, S. Gesing, D. S. Katz, S. Shahand, S. Henwood, T. Glatard, K. Jeffery, B. Corrie, A. Treloar, H. Glaves, L. Wyborn, N. P. C. Hong, and A. Costa, "The global impact of science gateways, virtual research environments and virtual laboratories," *Future Generation Computer Systems*, vol. 95, pp. 240 – 248, 2019.

[55] G. S. Blair, "Complex distributed systems: The need for fresh perspectives," in *ICDCS*, July 2018, pp. 1410–1421.

[56] M. Ragan-Kelley, F. Perez, B. Granger, T. Kluyver, P. Ivanov, J. Frederic, and M. Bussonnier, "The Jupyter/IPython architecture: a unified view of computational research, from interactive exploration to communication and publication." *The American Geophysical Union Fall Meeting*, Dec 2014.

[57] D. S. Janzen and H. Saiedian, "A leveled examination of test-driven development acceptance," in *ICSE*, May 2007, pp. 719–722.

[58] D. Renzel, I. Koren, R. Klamma, and M. Jarke, "Preparing research projects for sustainable software engineering in society," in *ICSE Software Engineering in Society Track*, May 2017, pp. 23–32.

[59] E. Jagroep, J. Broekman, J. M. E. M. van der Werf, P. Lago, S. Brinkkemper, L. Blom, and R. van Vliet, "Awakening awareness on energy consumption in software engineering," in *ICSE Software Engineering in Society Track*, May 2017, pp. 76–85.

[60] P. Lago and T. Jansen, "Creating environmental awareness in service oriented software engineering," in *ICSOC*. Springer, 2010, pp. 181–186.

[61] S. C. El Idrissi and J. Corbett, "Green is research: A modernity perspective," *Communications of the Association for Information Systems*, vol. 38, no. 1, p. 30, 2016.