IoT-Hub: New IoT Data-Platform for Virtual Research Environments

Rosa Filguiera*, Rafael Ferreira da Silva‡, Ewa Deelman‡, Vyron Christodoulou§, Amrey Krause*
* University of Edinburgh, EPCC, Edinburgh, UK. Email: {r.filgueira, a.krause}@epcc.ed.ac.uk
‡ University of Southern California, ISI, Marina Del Rey, CA, USA. Email: {rafsilva, deelman}@isi.edu
§ British Geological Survey, The Lyell Centre, Edinburgh, UK. Email: vyronc@bgs.ac.uk

***Abstract*—This paper presents IoT-Hub a new scalable, elastic, efficient, and portable Internet of Things (IoT) data-platform based on microservices for monitoring and analysing large-scale sensor data in real-time. IoT-Hub allows us to collect, process, and store large amounts of data from multiple sensors in distributed locations—which could be deployed as a backend for Virtual Research Environments (VRE) or Science Gateways. In the proposed data-platform, all required software, which involves a variety of state-of-the-art open-source middleware, is packed into containers and deployed in a cloud environment. As a result, the engineering and computational time and costs for deployment and execution is significantly reduced.**

***Keywords*—IoT, Science Gateway, Virtual Research Environment, Data-Frameworks, Containers, Data Science, Microservices**

I. Introduction

The emergence of the Internet of Things (IoT) is introducing a new era to the realm of computing and technology [1]. The proliferation of sensors and actuators that are embedded in things enables these devices to understand the environments and respond accordingly more than ever before. Additionally, it opens unlimited possibilities to domain scientists and/or data scientists for building models and analyses that turn this sensation into big benefits to science and society. Real-time processing of big data streams will gain importance as embedded technology increases and we continue to generate new types and methods of data analysis [2], particularly in regard to IoT. However, this revolutionary spread of IoT devices creates big challenges, such as choosing, deploying, and managing adequate data-frameworks for data-intensive computation in science, engineering, and many other fields.

Virtual Research Gateways (VREs), also known as Science Gateways [3], are web-tools accessible from anywhere. They usually provide an integrated view of all available resources with pervasive data access control, handle continuity between sessions and support collaboration with shared data and methods. VREs open up opportunities for sharing and comparing both experiment data from experiments, observations, and model runs and analytic interpretations of these data. They are very popular in a variety of scientific communities (e.g. seismology or astronomy) since they hide many technical and management details whose use are not straightforward for non-experts. The connection to and between VREs and science automation technologies has gained a lot of attention in the last years. However, that is not the case for the VREs, IoT, and all new middleware for emerging data-intensive analytics.

In this paper, we present IoT-Hub, an integrated, comprehensive, elastic, and portable data-platform based on microservices. IoT-Hub combines the benefits of several well-known data-frameworks with Docker containers. The current implementation of IoT-Hub includes a service-pipeline composed by *Apache Kafka*, *Apache Spark*, *Elasticsearch*, and *Kibana* middleware that enables automated gathering, preprocessing, storing, and visualization of IoT streams in a scalable, efficient, and robust manner. IoT-Hub acts as a backend for VREs to run stream-based applications, deploying cloud resources upon request. It reduces the engineering time and effort (and possible human errors) required by scientists or VRE administrators to build such complex systems. Our hypothesis is that if we provide scientific communities with portable and elastic platforms to interrogate the IoT data, it will speed up scientific discoveries.

We have demonstrated the feasibility of IoT-Hub via a real use case application, which processes sensor data from the British Geological Survey (BGS) environmental baseline programme [4] (freely available online). IoT-Hub collects, preprocesses, and stores in real-time time-series data from several distributed locations and sensors types, and makes them available to domain scientists (e.g. groundwater modelers) and data scientists, so they can use it to build their models, make predictions, and conduct analyses.

This paper is structured as follows. Section II presents background. Section III discusses IoT-Hub features. Section IV presents the use case for testing the platform. We conclude with a summary of achievements and outline future work.

II. Background and Related Work

In this section, we provide a brief overview on the state-of-the-art encompassing VREs, IoT, and middleware for data-Intensive analytics.

A. Virtual Research Environments (VREs)

VREs can be defined as community-development set of tools, applications, and data that is integrated via a portal or a suite of applications, usually in a graphical user interface, that is further customized to meet the needs of a specific community [5]. These tools sit behind the scenes and exploit a wealth of resources residing on multiple computing infrastructures

and data providers (according to their policies). Some VREs examples include:

- *VERCE* [6] is a data-intensive e-science environment to enable innovative data analysis and data modeling methods that fully exploit the increasing wealth of open data generated by the observational and monitoring systems of the global seismology community.
- *MoSGrid* [7] is a portal that offers an approach to carrying out high-quality molecular simulations on distributed compute infrastructures to scientists with all kinds of background and experience levels.
- *CyberSKA* [8] is a collaborative portal which aims to address the current and future needs of data-intensive radio astronomy. A wide variety of tools and services that have been developed and integrated with the CyberSKA portal, including a distributed data management system, a data access tool, remote visualization tools, and third party applications.
- *EFFORT* [9] is an innovative platform to promote persistent collaboration research in Rock Physics and Volcanology. It organizes data from rock physics experiments and volcano monitoring to open up opportunities for sharing and comparing data, observations and model runs, and analytical interpretation methods.
- *myExperiment* [10] is a portal for collaboration and sharing of workflows and experiments. In contrast to systems that simply make workflows available, it provides mechanisms to support the sharing of workflows within and across multiple communities via a social web approach.

Having a closer look to the technologies, tools, systems, and computing resources that are very often behind VREs' backends, we can categorize them as follows:

- *High Performance Computing solutions*: aggregated computing resources to perform high performance computations (including processors, memory, disk, and operating system) [11];
- *Distributed Computing Infrastructures*: distributed systems characterized by heterogeneous networked computers called to offer data processing facilities. This includes high-throughput computing and cloud computing;
- *Scientific workflow management systems* (SWMS): systems enacting the definition and execution of scientific workflows consisting of a list of tasks and operations, the dependencies between the interconnected tasks, control-flow structures, and the data resources to be processed [12], [13];
- *Data analytics frameworks and platforms*: platforms and workbenches enabling scientists to execute analytic tasks. Such platforms tend to provide their users with implementations of algorithms and (statistical) methods for the analytics tasks [14].

These classes of solutions and approaches are not isolated, rather they are expected to rely on each other to provide VREs end users with easy to use, efficient, and effective data processing facilities, e.g., SWMS rely on distributed computing infrastructures to actually execute their constituent tasks.

The proposed framework (`IoT-Hub`) can be used as a VRE backend, where scientists can simply inject and execute their processing analyses (via VRE fronted) without putting effort in operating the enabling technology. In order to meet this goal, we have leveraged *Docker* containers, which allows us to have an elastic computational environment based in loosely coupled services, which are immediately portable. *Docker* handles the packaging and execution of a container so that it works identically across different machines, while exposing the necessary interfaces for networking ports, volumes, and so forth, allowing other users to reconstruct an equivalent computational environment. Therefore, `IoT-Hub` can be deployed on demand (as-a-service) reducing engineering time and computational cost.

B. Internet of Things (IoT): Big Data challenges

The explosive increase in the number of devices connected to the IoT and the exponential increase in data consumption only reflect how the growth of big data perfectly overlaps with that of IoT [15]. And therefore, many architectural design challenges have arisen for the delivery of big data services based on the IoT. These challenges have been described in detail in [16]. In this work, we have mainly focused in the following ones:

- *The number of IoT devices*: With growth forecasted in the number of connected "things" and expected to reach billions world-wide, there will be masses of devices which may be a data source, and which may be subject to third party control;
- *Risk of IoT device malfunction*: With a great number of IoT devices and manufacturers it is reasonable to assume there will be many occasions where IoT devices malfunction in various ways;
- *Update frequency*: Though some devices will produce data reports at a low frequency there may be substantial quantities of data streaming from more sophisticated Internet connected things.

Therefore, `IoT-Hub` has been designed to collect data from a wide range of different IoT devices, geographically distributed that stream data at different frequency ratios, and could yield malfunction behaviors intermittently.

Our proposed solution provides an IoT data-platform, which support an ecosystem of third party application developers (e.g.domain scientists and data scientists) to explore data given the described challenges. `IoT-Hub` offers a degree of flexibility making use of *Docker* and *Docker-compose* tools for deploying services on demand in virtualized environments, such as cloud systems. Previous works have targeted similar environments, such as [17], which is a cloud-based autonomic information system for delivering Agriculture-as-a-Service (AaaS) through the use of cloud and big data technologies.

C. Microservices & Middleware

With recent advances in cloud computing, virtualization, containerization, continuous integration, and the DevOps movement, deploying software solutions today is very different from even just a few years ago. Today's distributed applications are built as a set of independently deployable microservices distributed over clusters of commodity hardware. The microservice term – also known as the microservice architecture – refers to a new architectural style that structures an application as a collection of loosely coupled services, which implement business capabilities. So, we have built `IoT-Hub` following microservice principles [18].

The question that arises now is, which components should be used in `IoT-Hub` to develop a high performance platform to efficiently analyze IoT big data. For answering this question, we developed a prototype platform in an elastic cloud environment, and *Falcon*[1], *Apache Kafka*[2], *Apache Spark*[3], *Elasticsearch*[4], *Kibana*[5] and *Docker*[6] have been initially selected (see Table I). This selection could be easily extended in the future for including additional data-frameworks, such as *Cassandra*, *Apache Flink*, and *Jupyter Notebooks*.

TABLE I Overview of software that conforms `IoT-Hub`.

Technology	***Description***	***Version***
Falcon	Reliable, high-performance Python web framework for building large-scale app backends and microservices. It encourages the REST architectural style with minimal external dependencies, while remaining highly effective.	2.0
Apache Kafka	Distributed streaming platform that allows for publishing and subscribing to streams of records (topics) in a fault-tolerant way and process streams of records as they occur.	0.10.2.0
Apache Spark	Fast and general engine for large-scale data processing. Among other features, it allows writing streaming jobs the same way as writing batch jobs. It supports Java, Scala and Python.	2.2.0
Elasticsearch	A distributed, RESTful search and analytics engine for performing and combining many types of searches structured, unstructured, geo or metric.	6.2.2 (oss)
Kibana	An open source data visualization plugin for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. It also supports remote I/O	6.2.2 (oss)
Docker	A lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.	1.13.1

[1]https://falconframework.org/
[2]https://kafka.apache.org/
[3]https://spark.apache.org/
[4]https://www.elastic.co/
[5]https://www.elastic.co/products/kibana
[6]https://www.docker.com/

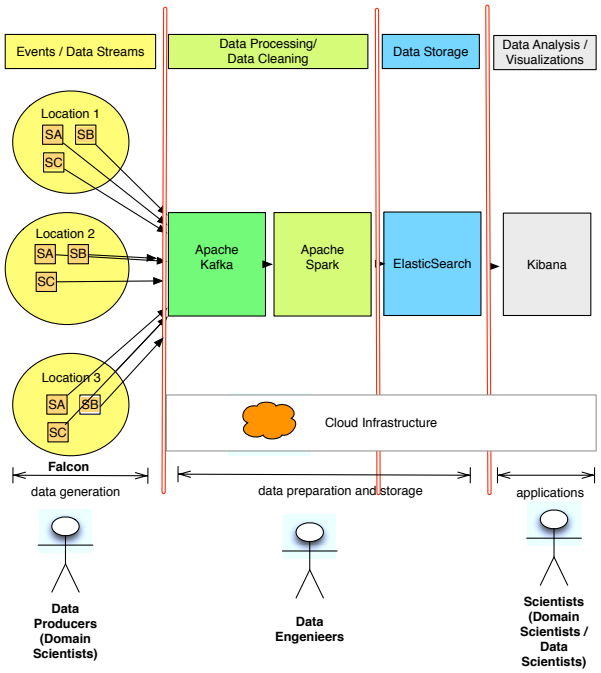

Fig. 1: `IoT-Hub`: Data-platform for gathering, quality checking, storing, and visualizing environmental sensors streams.

III. `IoT-Hub` FEATURES

`IoT-Hub` integrates several middleware based on the microservices architecture. *Apache Kafka* provides the mechanism for ingesting real-time data streams and making them available to downstream consumers in a parallel and fault-tolerant manner. Data in *Apache Kafka* is organized into *topics* that are split into partitions for parallelism. A *topic* can be viewed as an infinite stream where data is retained for a configurable amount of time. *Producers* are applications that publish stream of records to one or more *topics*. In our case, *Apache Kafka* streams events out to *Apache Spark consumers* which are subscribed to one or more *topics* for parsing their content, all of which is done in near real-time.

Spark Streaming API enables scalable, high-throughput, fault-tolerant stream processing of live data streams. Data can be ingested from many sources (e.g. *Apache Kafka, Flume, Twitter*), but in this current version of `IoT-Hub` we limited it to *Apache Kafka*. *Spark Streaming API* can be used for processing the ingested data using complex algorithms composed of high-level functions like `map`, `reduce`, `join`, and `window`. The processed data can be published to yet another *Kafka topic* for further consumption or it can be stored as results in HDFS, databases, or dashboards. In this work, we have selected *Elasticsearch* as temporary storage system. One of the reasons for this choice is *elasticsearch-hadoop* provides native integration between *Elasticsearch* and *Apache Spark*, in the form of an *RDD* (Resilient Distributed Dataset).

Kibana offers interactive visualizations (e.g. histograms, line graphs, pie charts, sunbursts, etc.) and advanced time

series analysis on *Elasticsearch* data, by leveraging the full aggregation capabilities of *Elasticsearch*.

In order to have a portable, scalable, and elastic data-platform, we created a *Docker cluster* for each of the previous middleware and connected them via *docker-compose*, since it allows us to setup and run multi-container environments. Figure 1 shows a visual description of the `IoT-Hub` components, and the interactions from different roles (e.g. data producers, data architects, data scientists, domain scientists) that we anticipate. All `Dockerfiles` and `docker-compose file` used to generate the `IoT-Hub` are available freely online in a GitHub repository[7] for allowing reproducibility and share our approach among the scientific community.

For our experiments, we have used the NSF-Chameleon cloud[8], using a CentOS7 image with 42 CPUS for deploying our hub. Note that the proposed framework could be deployed to any other Cloud system.

IV. Caste study: Environmental Baseline Monitoring Programme

To demonstrate the feasibility of `IoT-Hub`, we have used the *Environmental Baseline Monitoring programme* [4], which provides the perfect scenario for testing our platform with IoT sensor data. The British Geological Survey (BGS), along with partners from the Universities of Manchester, York, Birmingham, Bristol, and Public Health England (PHE), is carrying out a science-based environmental monitoring programme in the areas of *Lancashire* [19] and *Vale of Pickering* [20]. This programme represents the first independent, integrated monitoring study to characterize the environmental baseline in areas subjected to close scrutiny in anticipation of the development of a nascent UK shale-gas industry. The monitoring involves ways of managing high volume, highly varied data, generated by a range of IoT sensor data, including:

- *Groundwater quality*: The sensors installed in the boreholes provide real-time measurements of water-quality parameters: water level, temperature, pH, conductivity, and dissolved gases (O2, CH4, CO2, Rn).
- *Seismicity*: The monitoring of background seismicity has involved installation of a network of seismic stations in the vicinity of the proposed shale gas wells. Real time seismic data are being collected from the array of stations to help characterize current levels of seismic activity. The information captured in near real-time includes: station code, station name, seismic data (for a single channel), and timestamp.
- *Air composition*: The monitoring equipment measures concentrations of ozone (O3), particulate matter (PM1, PM2.5, PM4, and PM10), nitrogen oxides (NO, NO2 and NOx), methane (CH4), non-methane hydrocarbons (NHMCs), hydrogen sulphide (H2S) and carbon dioxide (CO2) as well as capturing meteorological information.

[7]https://github.com/rosafilgueira/EMB_datastreaming
[8]https://www.chameleoncloud.org/

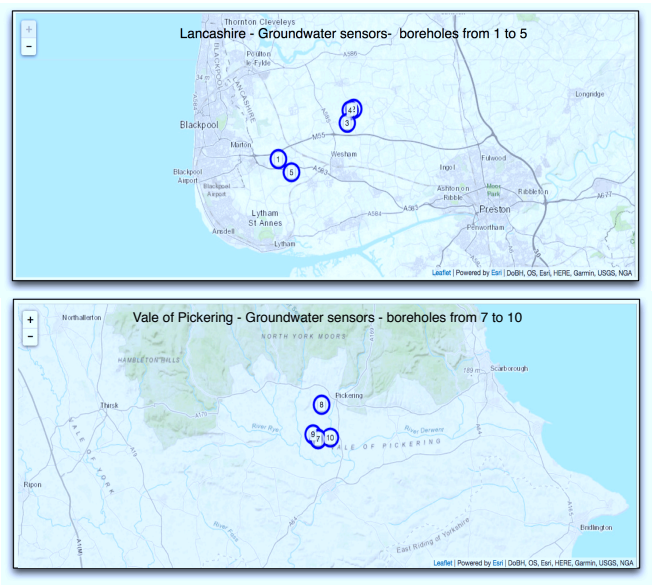

Fig. 2: Groundwater sensors from the areas of *Lancashire* and *Vale of Pickering*. In total, we have 9 boreholes (sensors are attached to boreholes) placed at geographically distributed locations.

We have initially focused on *Groundwater quality* sensor data. However, very little work has to be done in `IoT-Hub` to enable support to other sensors. This is discussed in Section V. These groundwater sensors are attached to boreholes, which are called `emb1`, `emb2`, `... emb10`. Figure2 shows the locations of these boreholes. For simplicity, we have selected `emb2`, `emb3`, and `emb4` boreholes, but `IoT-Hub` supports any number of boreholes and sensors.

`IoT-Hub` collects in simulated real-time the water-quality parameters described before, from sensors attached to the selected boreholes (marked as **2**, **3**, and **4** in Figure 2). Since we did not have direct access to these sensors, but access to yearly compressed files instead, monthly datasets were downloaded locally [19]. These datasets are originally in comma separated values (CSV) format and sensors provide a single reading for every hour (every line corresponds to an hour reading) per each day. In this work, we simulate the setup where sensor data corresponding to one hour arrives every 10 seconds to test the ability of `IoT-Hub` to deal with high frequency data transmission. To achieve this, a feeder script was implemented to send POST requests messages (with a sensor reading) to *Falcon* web services every 10 seconds. Then, *Falcon* was configured to act as a *producer*, publishing streams to *Apache Kafka* by using the *emb* topic. *Apache Kafka* ingests those streams in real-time and makes them available to a *Spark-Streaming* application which acts as a *consumer*. This application is subscribed to the *emb* topic, and stores the data in *Elasticsearch* (see Listing 1) after performing a quality check over the values received (e.g. if the data is within the range specified by the sensor manufacturers). If

the values are within the predefined range, all the data is stored into the corresponding fields and the `normal` label is stored in the *qc* field. Otherwise, the data is stored and annotated with the `anomaly` label instead. Note that *Apache Kafka* offers a decoupling and buffering of the input streams. Therefore *producer-consumer* need not to know about each other. From multiple sources *producers* can write data to any topic in kafka, and several *consumers* can be subscribed to the same topic making each of them different processing analysis. Since kafka persists data to disc, *consumers* can be shut down for performing changes and when they are restarted, they will retrieve all data from the time they were offline.

Listing 1: Elasticsearch index for storing groundwater values.

```
index:emb_test
type:emb
fields:
  sensor_id -> type: text, description: Id of the sensor
  date -> type: date, description: Date (UTC)
  time ->type: date, description: Time (UTC)
  sec -> type: integer, descrip.: Microsiemens per centimer
  ph -> type: float, description: PH
  water_level-> type: float, description: Water level aOD
  water_temp-> type: float, description: Water temperature
  tdg -> type: integer, description: Total dissolved gas
  qc -> type: text, description: Quality control
```

Once sensor data becomes available in *Elasticsearch*, different applications can be run to analyze it. These applications can vary from simple scripts to check the insertion of data, to more complex machine learning analysis, such as the anomaly detection that will be explained in the following subsection.

Furthermore, `IoT-Hub` includes *Kibana* for exploring the data visually (see Figure 3 as an example).

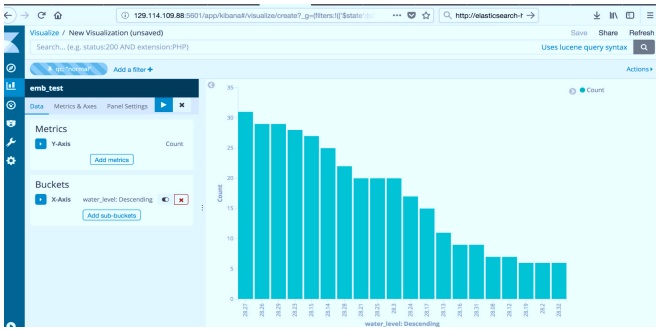

Fig. 3: *Kibana* screenshot for filtering the `normal` values, and counting the different values of the *water_level* parameter.

A. Anomaly Detection

An anomaly detection (AD) algorithm has been implemented to periodically interrogate the data that is automatically collected, processed, and stored by `IoT-Hub`. The AD algorithm is called every 15 minutes to review all the data transmitted in that period. The `IoT-Hub` presents new challenges in an AD context due to its continuous data streaming.

Therefore, we have established a three criteria that an algorithm must fulfill in `IoT-Hub`:

1) Robustness in seasonal changes (i.e., weather patterns).
2) Minimal or no parameter tuning.
3) Both globalized and localized AD in evolving time series.

The criteria address two main problems in the AD domain. First, the monitoring of the network for the detection of anomalies without a-priori knowledge. Any selected method should be able to perform well both in global and local AD. Detecting localized anomalies can be utilized as a warning system to prevent device failures. As a consequence, this delivers more consistent up-time, improved data quality, and thereby results in a more robust system. Second, the monitoring of global trends that change over time and detect any unusual seasonal patterns. This can help in improving the understanding about the nature of seasonal patterns or changes in research environments.

Due to the aforementioned reasons, the algorithm selected is *Twitter's Seasonal Hybrid Extreme Studentized Deviate* (S-H-ESD) AD algorithm [21] that uses robust statistics with a focus on analyzing long term and short trends in time series. The underlying algorithm employs time series decomposition to detect both global and local anomalies. A combination of piecewise approximation to extract the trend of a time series and an ESD test for anomalies is performed to accommodate a more localized AD. What is of particular interest is the ability to detect both local and global anomalies or seasonal changes and identify when a different pattern emerges in a continuous data flow.

The only algorithm parameter that was pre-configured was the maximum anomaly (set to 0.2), m, that controls the algorithm's upper bound of suspected anomalies. The algorithm ran for each of the different measurements. For brevity, in Figure 4 only the most interesting AD case is shown. In this case, the first (left) and the second run (right) of the algorithm after 15 minutes are shown. The first figure shows normal deviation and no anomalies are detected. In the second run, the algorithm picks up the anomalies that show a small spike before the sensor stops provides readings of value zero.

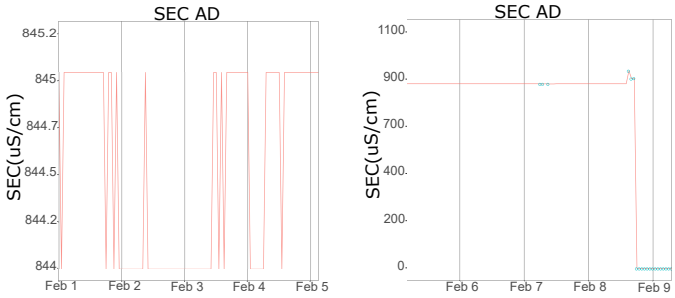

Fig. 4: `IoT-Hub`: AD results in `emb3`, Lancashire sensor for February 2017. First run (left), second run (right). Detected anomalies are shown as turquoise dots.

The performance of the method is promising in a real world scenario for AD. However, as can be seen by the high number of anomalies there is always the caveat of too many false positives, which is something to be avoided in real world

scenarios. The parameter selection has also to be evaluated periodically. Careful consideration of algorithms that fulfill the above criteria with a focus on precision rather than recall have to be chosen in order to be able to deliver robust results.

In the future, we plan to create a warning system (e.g., via a VRE fronted) that makes use of `IoT-Hub` for running the described AD analysis to interpret sensors in the field. This system will send out alerts to domain scientists subscribed to these alerts, as well as to those in charge of deploying and maintaining the sensors in the field.

V. Conclusions and Future Work

In this paper, we have presented `IoT-Hub`, which delivers specialized data-framework to exploit the IoT in a scalable, efficient, and robust manner reducing the engineering time and computational cost. We have demonstrated the feasibility of the proposed solution by using the *Environmental Baseline Monitoring programme*. Data from different sensors are collected, preprocessed, and stored in real-time using different microservices middleware. To test the capacity of `IoT-Hub` for running complex data-analytics tasks, we have implemented an *anomaly detection* algorithm, which queries data from *Elasticsearch* and detects the anomalies of each of the water-quality parameters. All the middleware that forms `IoT-Hub` has been containerized, which enables flexible and agile development, and deployment in cloud-based infrastructures.

The current version of `IoT-Hub` has been pre-configured for working with groundwater sensors. To extend this work to other sensors, it only would require to: (1) create a new *Kafka topic* to produce and consume new datasets; (2) modify the *Spark-Streaming* application to consume and check the data from this new *topic*; and (3) create a new *Elasticsearch* index.

One of the main uses of `IoT-Hub` could be to act as a backend for VREs or Scientific Gateways for running data-intensive applications and deploying cloud resources upon request.

As future work, we plan to include more middleware in `IoT-Hub`, such as *Cassandra* database (for high performance operations and handling massive datasets), an *RDF repository* (to store and harvest RDF data), *SparQL Endpoint* (to query a knowledge base via the SPARQL language), a *job submission system* (to submit applications to distributed computing infrastructures), and *Jupyter Notebook* (to offer an interactive computational environment).

Additionally, we also plan to create a warning system that makes use of `IoT-Hub` for running the described AD analysis to interpret sensors in the field, and leverage `IoT-Hub` capabilities to process near real-time logs from scientific workflow executions [22].

Acknowledgments. This work was carried out when the lead author was with the British Geological Survey. It was funded under the Scottish Informatics and Computer Science Alliance with the Postdoctoral and Early Career Researcher Exchanges fellowship, partially funded by DOE under Contract DESC0012636, " Panorama - Predictive Modeling and Diagnostic Monitoring of Extreme Science Workflows", and by " DARE -Delivering Agile Research Excellence on European e-Infrastructures", EU H2020 funded project (No 777413). We thank the NSF Chameleon Cloud for providing time grants to access their resources. This work contains British Geological Survey materials NERC [2017 and 2018 years].

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] M. Atkinson and M. Parsons, "The digital-data challenge," in *The DATA Bonanza – Improving Knowledge Discovery for Science, Engineering and Business*, M. P. Atkinson *et al.*, Eds. Wiley, 2013, ch. 1, pp. 5–13.

[3] L. Candela, D. Castelli, and P. Pagano, "Virtual research environments: An overview and a research agenda," *Data Science Journal*, vol. 12, pp. GRDI75–GRDI81, 2013.

[4] "Instrumenting the earth," http://www.bgs.ac.uk/Sensors/.

[5] C. E. Catlett, "Teragrid: A foundation for us cyberinfrastructure," in *Proceedings of the 2005 IFIP International Conference on Network and Parallel Computing (NPC'05)*, 2005, pp. 1–1.

[6] M. Atkinson, M. Carpene, E. Casarotti, S. Claus, R. Filgueira *et al.*, "Verce delivers a productive e-science environment for seismology research," in *2015 IEEE 11th International Conference on e-Science (e-Science)(E-SCIENCE)*, vol. 00, 2015, pp. 224–236.

[7] L. de la Garza, J. Krger, C. Schrfe, M. Rttig, S. Aiche, K. Reinert, and O. Kohlbacher, "From the desktop to the grid: conversion of knime workflows to guse," in *Proc. IWSG 2013*, ser. CEUR-WS, vol. 993, 2013, p. 9. [Online]. Available: http://ceur-ws.org/Vol-993/

[8] C. Kiddle *et al.*, "Cyberska: An on-line collaborative portal for data-intensive radio astronomy," in *2011 ACM Workshop on Gateway Computing Environments (GCE '11)*, 2011, pp. 65–72.

[9] R. Filgueira *et al.*, "escience gateway stimulating collaboration in rock physics and volcanology," in *2014 IEEE 10th International Conference on e-Science - Volume 01*, ser. E-SCIENCE '14, 2014, pp. 187–195.

[10] D. De Roure, C. Goble, and R. Stevens, "The design and realisation of the myExperiment Virtual Research Environment for social sharing of workflows," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 561–567, 2009.

[11] G. Hager and G. Wellein, *Introduction to High Performance Computing for Scientists and Engineers*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2010.

[12] I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, *Workflows for e-Science: scientific workflows for grids*. Springer Publishing Company, Incorporated, 2014.

[13] J. Liu, E. Pacitti, P. Valduriez, and M. Mattoso, "A survey of data-intensive scientific workflow management," *J. Grid Comput.*, vol. 13, no. 4, pp. 457–493, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1007/s10723-015-9329-8

[14] C.-W. Tsai, C.-F. Lai, H.-C. Chao, and A. V. Vasilakos, "Big data analytics: a survey," *Journal of Big Data*, vol. 2, no. 1, p. 21, 2015.

[15] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, A. I. A. Ahmed, M. Imran, and A. V. Vasilakos, "The role of big data analytics in internet of things," *Comput. Netw.*, vol. 129, no. P2, pp. 459–471, 2017.

[16] "Iot big data framework architecture," https://www.gsma.com/iot/wp-content/uploads/2016/11/CLP.25-v1.0.pdf.

[17] S. S. Gill, I. Chana, and R. Buyya, "Iot based agriculture as a cloud and big data service: The beginning of digital india," *JOEUC*, vol. 29, no. 4, pp. 1–23, 2017.

[18] I. Nadareishvili, R. Mitra, M. McLarty, and M. Amundsen, *Microservice Architecture: Aligning Principles, Practices, and Culture*, 1st ed. O'Reilly Media, Inc., 2016.

[19] "Environmental baseline monitoring in the lancashire," http://www.bgs.ac.uk/research/groundwater/shaleGas/monitoring/lancsDataSummary.html.

[20] "Environmental baseline monitoring in the vale of pickering," http://www.bgs.ac.uk/research/groundwater/shaleGas/monitoring/vopDataSummary.html.

[21] O. Vallis, J. Hochenbaum, and A. Kejariwal, "A novel technique for long-term anomaly detection in the cloud," in *6th USENIX Conference on Hot Topics in Cloud Computing (HotCloud'14)*, 2014, pp. 15–15.

[22] E. Deelman *et al.*, "PANORAMA: An approach to performance modeling and diagnosis of extreme scale workflows," *International Journal of High Performance Computing Applications*, vol. 31, no. 1, pp. 4–18, 2017.