This is a pre-print version of the paper Assante M, Candela L, Castelli D, et al. Virtual research environments co-creation: The D4Science experience. Concurrency Computat Pract Exper. 2022;e6925. doi: 10.1002/cpe.6925

RESEARCH ARTICLE

Virtual Research Environments Co-creation: the D4Science Experience

M. Assante[1] | L. Candela*[1] | D. Castelli[1] | R. Cirillo[1] | G. Coro[1] | A. Dell'Amico[1] | L. Frosini[1] | L. Lelii[1] | M. Lettere[2] | F. Mangiacrapa[1] | P. Pagano[1] | G. Panichi[1] | T. Piccioli[1] | F. Sinibaldi[1]

[1]Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", National Research Council of Italy, Pisa, Italy
[2]Nubisware S.r.l., Pisa, Italy

Correspondence
*L. Candela, Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", via G. Moruzzi, 1, 56122 Pisa, Italy. Email: leonardo.candela@isti.cnr.it

Summary

Virtual research environments are systems called to serve the needs of their designated communities of practice. Every community of practice is a group of people dynamically aggregated by the willingness to collaborate to address a given research question. The Virtual Research Environment provides its users with seamless access to the resources of interest (namely, data and services) no matter what and where they are. Developing a Virtual Research Environment thus to guarantee its uptake from the community of practice is a challenging task. In this paper, we advocate how the co-creation driven approach promoted by D4Science has proven to be effective. In particular, we present the co-creation options supported, discuss how diverse communities of practice have exploited these options, and give some usage indicators on the created VREs.

KEYWORDS:
Virtual research environment, Science gateway, Co-creation

1 | INTRODUCTION

When building systems that aim to provide designated communities with services to use, the motto "*build it and they will come*" must be carefully considered[1,2,3,4]. Science Gateways and Virtual Research Environments (VREs) fall under these settings; their uptake by the designated communities is paramount[5,6,7].

A series of success criteria for VREs have been proposed and discussed[8]. Buddembohm et al. highlighted that the success of a VRE depends to a great extent on integrating the user group right from the start. This involvement takes "*the form of a user requirements and market analysis, accompanied by project monitoring and is designed to ensure the biggest possible overlap between the requirements of the user group and what the VRE will later offer*"[8]. We would extend this by stressing the "velocity" dimension, a quality nowadays pervading science and communities of practice. The requirements, technologies, and general settings characterising the target scenarios are highly evolving, thus calling for agile-oriented solutions rather than highly planned, firmly, and carefully architected approaches risking no longer responding to the community needs when made available.

Co-creation is a widely used term to describe participative models where several actors together generate and develop a common "meaning"[9]. It is also known as 'participatory' or 'cooperative' design. One of the essential elements characterising the process is the continuous use of prototypes as a mechanism to bring ideas and new features to life and to generate feedback. The overall approach aims at innovation by recognising that radical and adaptive creativity can co-exist when creating new products. Moreover, groups working together tend to develop better ideas than selected lead-users.

When exploited to promote the development of VREs, the co-creation process suggests following a participatory process intertwining the activities of software developers and service providers with the activities of the VRE designated community that bring specific value to the VRE. In fact, VREs consist of two complementary parts: the *community-agnostic* part, i.e. services offering basic or advanced functionality exposing an expected behaviour when instantiated in diverse contexts; the *community-specific* part, i.e. services offering a peculiar functionality or data, sometimes implemented by combining into specific workflows the community-agnostic part with context-specific services or data. The collaborative and participatory development of these two parts makes it possible for community-agnostic software developers and service providers to incrementally develop solutions matching the needs of diverse communities by promptly testing them in actual usage scenarios. Moreover, it enacts VRE designated community members to actively contribute to the incremental development of the VRE by bringing in their peculiarity and diversity.

This paper describes how the co-creation approach to VRE development has been implemented and promoted by D4Science [10,11]. In particular, it documents the co-creation options enacting designated communities to actively contribute to the realisation of their envisaged VREs and discusses how diverse communities of practice have used these options.

The remainder of the paper is organised as follows. Section 2 describes the integration options communities of practice can use to plug into VREs applications, analytics tasks, and any worth sharing asset. Section 3 discusses how these integration options have been exploited by the diverse communities of practice served by D4Science. Section 4 presents related works. Finally, Section 5 concludes the paper by analysing the results of the proposed solutions.

2 | THE CO-CREATION OPTIONS

D4Science is an IT infrastructure specifically conceived to support the development and operation of VREs by the as-a-Service provisioning mode [10,11]. D4Science-based VREs are web-based, community-oriented, collaborative, user-friendly, open-science-enabler working environments for scientists and practitioners willing to work together to perform a specific (research) task. From the end-user perspective, each VRE manifests in a unifying web application (and a set of Application Programming Interfaces (APIs)) (*a*) comprising several components made available by portlets organised in custom pages and menu items and (*b*) running in a plain web browser. Every component aims to provide VRE users with facilities implemented by relying on one or more services possibly provisioned by diverse providers. Every VRE plays the role of a gateway giving seamless access to the datasets and services of interest for the designated community while hiding the diversities originating from various resource providers. Among the components each VRE offers, some basic ones are enacting VRE users to perform their tasks collaboratively [11], namely: (*a*) a *workspace* component to organise and share any digital artefact of interest; (*b*) a *social networking* component to communicate with coworkers by posts and replies; (*c*) a *data analytics* platform to share and execute analytics methods; (*d*) a *catalogue* component to document and publish any worth sharing digital artefact.

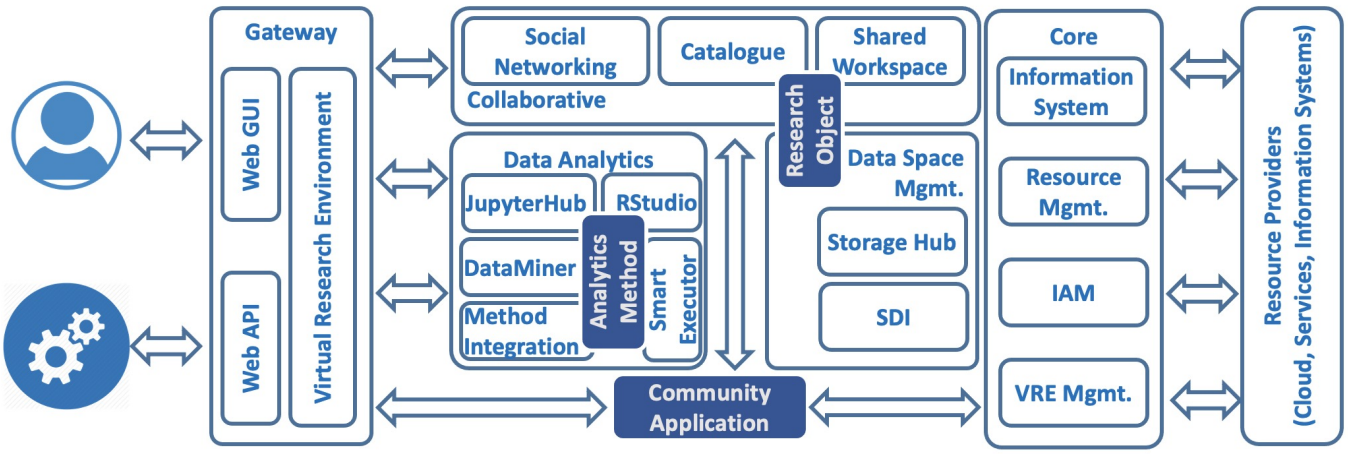


FIGURE 1 D4Science VREs: Overall architecture

Fig. 1 depicts the service-oriented view of the D4Science architecture (for details, refer to previous works [10,11]). Services are conceptually organised into three groups: *front-end components* – the D4Science part with which user interacts directly; *back-end components* – the D4Science part implementing the business logic of the system; *provided resources* – the D4Science part providing front-end components and back-end components with resources (computing, storage, data, software) to use.

The D4Science front-end manifests into some Liferay[†] portal instances and a number of REST APIs for accessing the services serving a specific VRE. These instances are either replicas of the portal or the service implementing one of the APIs. Instances are made available by a (high availability) proxy implementing load balancing policies, i.e. distributing the calls to the available service instances to balance the load. The Liferay portal is equipped with a series of portlets specifically conceived to give access to functionalities offered by one or more components of the D4Science back-end. Whenever a new VRE is created, the portal is instructed to support the specific Site (Liferay concept) implementing the VRE front-end.

The D4Science back-end components are services (often frameworks on their own) organised in four main areas [10]: (*i*) core services supporting VREs management, resources management, authentication and authorisation; (*ii*) data space management services supporting the storage and management of various typologies of data, including files stored in diverse storage systems and geospatial data managed by a Spatial Data Infrastructure (SDI) exploiting an array of orchestrated GeoNetworks, THREDDS Data Servers, and GeoServers.[‡] (*iii*) data analytics services supporting several options for data analytics, including the DataMiner proprietary platform with its integration facility (cf. Sec. 2.2), JupyterHub, and a cluster of RStudio instances; (*iv*) collaborative services implementing facilities enacting the collaboration among the members of a VRE, e.g. by supporting communication and sharing.

The D4Science provided resources include the D4Science distributed computing infrastructure. This distributed computing infrastructure is spread across four main sites, geographically distributed, and managed across different administrative domains.

The Pisa site is the core element of the D4Science computing infrastructure. It realises a cloud infrastructure based on open source technologies aiming at guaranteeing the dynamic allocation of the hardware resources and high availability of the services. In particular, the site is based on OpenStack[§], while Ceph[¶] manages the storage. Service provisioning is entirely automatised by relying on a set of Ansible Playbooks[#]. Service provisioning is focused not only on D4Science services but also makes it possible to deploy (R)DBMS, NoSQL solutions, and others. The site is equipped with a Docker cluster orchestrated via Docker Swarm[‖]. This site provides 1650 CPU Cores, 0.5 PB persistent storage, and 10 TB RAM.

Three sites are operated on GARR premises, i.e. the Italian National Research and Education Network. In particular, these sites (actually OpenStack regions) are based in Catania, Naples, and Palermo. The GARR sites' computing capacity contributes to D4Science is 1950 CPU Cores, 3.6 TB RAM, and 30 TB persistent storage.

The D4Science computing infrastructure also exploits resources from the EGI Federation [12] and resources operated by the Copernicus DIAS service named WEkEO[**], thanks to the agreements established by specific communities using D4Science. WEkEO resources (48 CPUs, 12 GPUs, 192 GB RAM (+117 GB GPU node), 1 TB on SSD disk, 11 TB persistent storage, 20 TB object storage) enacted the provisioning of the analytics service instances on WEkEO premises, thus reducing the time required to access the Copernicus data needed for the computations of geospatial products.

Three integration patterns are supported (besides implementing completely new services) to complement this offering and bring community resources (the boxes with white names in Fig. 1) into VREs: (*a*) integrating existing applications (cf. Sec. 2.1); (*b*) integrating analytics methods and workflows (cf. Sec. 2.2); (*c*) integrating datasets and other resources for discovery and access (cf. Sec. 2.3). These patterns usually come into play after a VRE is created by relying on D4Science services only. The co-creation mechanism they enact counts on the presence of a working version of the VRE where community resources are "hot-plugged" without stopping or shutting down the environment.

2.1 | Integrating applications

D4Science offers four options (cf. Tab. 1) for integrating existing *applications* (i.e. stand-alone systems implementing one or more functionality either via web-based User Interfaces or via APIs) into VREs: (*i*) *Adaption level integration* where there is the

[†] Liferay platform webpage https://www.liferay.com
[‡] GeoNetwork webpage geonetwork-opensource.org; THREDDS Data Server (TDS) https://doi.org/10.5065/D6N014KG; GeoServer webpage geoserver.org/
[§] OpenStack website www.openstack.org
[¶] Ceph website ceph.com
[#] Ansible website www.ansible.com
[‖] Docker Swarm documentation website https://docs.docker.com/engine/swarm/
[**] WekEO https://www.wekeo.eu/

willingness to integrate an existing application into a VRE fully, (*ii*) *Adoption level integration* where there is the willingness to integrate an existing application only for what regards its operation and management, (*iii*) *Entry-level integration* where there is the willingness to reach a basic level of integration between the application and the VRE, (*iv*) *Client integration* where the willingness to integrate the application is unidirectional, meaning that the application will not be part of the VRE yet using the application it is possible to have access to the VRE resource space and contribute to it. Each integration option is characterised by an effort required for the implementation and a resulting benefit (discussed below). When integrated, the application becomes part of the D4Science infrastructure (to a degree depending on the specific pattern). Each application or service joined to the D4Science infrastructure can be exploited to build as many VREs as suitable. Thus the D4Science infrastructure becomes a "marketplace" where communities of practices can both (*a*) advertise their assets for "others" to use and (*b*) find a rich and ever-growing set of facilities to be easily assembled to form specific VREs.

TABLE 1 D4Science integration options by characteristic. *Hosting*, i.e. whether the integration pattern implies that the D4Science computing infrastructure hosts the application or not; *AuthN & AuthZ*, i.e. whether the integration pattern implies that the application is interacting with the D4Science Identity and Access Management Service or not; *Monitoring*, i.e. whether the integration pattern implies that the application is integrated with the monitoring system or not; *VRE Services*, i.e. whether the adoption of the pattern implies that the application is integrated with the VRE back-end services or not.

	Hosting	AuthN & AuthZ	Monitoring	VRE Services
Adaption	Yes/No	Yes	Yes	Yes
Adoption	Yes	Yes	Yes	No
Entry	Yes/No	Yes	Yes/No	No
Client	No	Yes	No	Yes/No

Before analysing each pattern, it is worth quickly discussing the security settings for D4Science, and its VREs: (*i*) the communication between services hosted on different sites is counting on the Transport Layer Security protocol; (*ii*) the use of any service is regulated by authentication and authorisation for both human users and applications; (*iii*) authorisation is realised by a token-based approach where the token is associated with every interaction and used to verify whether the owner of the token is authorised or not to execute the action on the target resource; (*iv*) the authorisation service exposes OAuth2[13] protocol APIs to enact any third-party application to interact with D4Science services, possibly on a user's behalf.

Concerning the provisioning of the applications, this can be done either (*a*) by the *as-a-Service* delivery model where the application runs on application owner premises or (*b*) by the *software package* delivery model, including containerized applications ready to be hosted on D4Science computing infrastructure. Applications provisioned with the second approach include Docker containers to be deployed into a Docker Swarm cluster and Shiny Apps deployed on request by a ShinyProxy[††].

Adaption level integration

The application owner is willing to integrate its existing application (a system on its own) with the community-agnostic services of the VRE so that there is a smooth experience for VRE users when moving from one service to another. The application owner is requested to modify the application thus to make it interoperable with (*i*) the *workspace* to transparently read content from it and store content in it from the application; (*ii*) the *social networking* thus to post messages from the application; (*iii*) the *catalogue* thus to publish or discover catalogue items from the application; (*iv*) the *data analytics* (cf. Sec. 2.2) thus to execute analytics tasks from the application. Interoperability is achieved by exploiting these services' REST APIs and their standards. In addition to that, the owner might decide to follow the Adoption level integration discussed in the next paragraph or to host and operate the application on its premises. Regarding the presentation layer, the application owner may opt to develop a dedicated "portlet" or ask for the lightweight strategy of embedding the current application GUI by an HTML iframe component.

[††] ShinyProxy Website https://www.shinyproxy.io/

Adoption level integration

The application owner is willing to integrate its application by focusing on its operation and management only rather than reconsidering its business logic to consider the infrastructure offerings and settings.

Two approaches are supported: one based on *SmartGears* and one based on *SmartProxy*.

A SmartGears[10] node is a customised Java 8 Servlet container (Apache Tomcat 8). This approach implies that the application to be integrated runs on a Java Virtual Machine. The benefits resulting from this integration pattern are that the service becomes fully interoperable with the infrastructure and can delegate the following functions: Authorisation and Authentication; Users and Roles Management; Auditing and Tracing; Monitoring. However, this pattern is limited in scope to the cases matching the described technological requirements.

The SmartProxy approach aims at overcoming the limitations of the SmartGears one. It consists of a Service Proxy, which is deployed "in front of" the integrated application. A SmartProxy transparently performs orthogonal functionality such as authentication and authorisation enforcement, monitoring, and accounting on the ingress traffic of the application before forwarding it. It may also enrich communication with meta-information such as HTTP Headers if required. Like SmartGears, SmartProxy is entirely transparent for the proxied applications and their clients. It has the significant advantage of not requiring a specific technology for the applications to be proxied besides the fact that they communicate with HTTP/s. In addition, we have decided to implement this pattern by exploiting NGINX-based components, which are already in use in the infrastructure for typical tasks such as TLS termination, routing, and load balancing.

Entry-level integration

The application owner is willing to reach a basic level of integration between its application and the VRE. Integration is achieved by passing to the application a JSON Web Token[‡‡]. The application may run by its technology server (inside or outside the infrastructure premises) but has to be known and validated a priori by the infrastructure. The application then uses this Web Token to perform service calls to D4Science services needed for its functions. The benefits resulting from this integration pattern are that the service becomes interoperable with the infrastructure and may delegate the following functions without changes in its implementation: Authorisation and Authentication; Users and Roles Management; Monitoring (only if the D4Science computing infrastructure hosts the service).

Client integration

The application owner plans to use VRE services and content rather than rely on a VRE environment made available by a D4Science based gateway. A lightweight integration characterises this scenario with few benefits. It can be used only when the application needs to contact some services but cannot perform operations on behalf of the users. In this pattern, the infrastructure provides the application owner with an Application token to be used client-side to perform service calls required for the service functions. The overall management of the application remains on the provider side.

2.2 | Integrating analytics methods

D4Science is equipped with a feature-rich platform for data analytics[14]. This platform offers a web-based method integration environment for communities willing to transform almost any existing method and implemented procedure into an executable process provided by the platform.

The integration process is relatively straightforward. It mainly consists in provisioning the source code of the method/procedure and instructing the platform on how to invoke it by precisely specifying the main component and its input and output parameters. The provisioning of the code can be done by depositing it into the workspace or referencing it via GitHub. A specific GUI has been designed and developed to facilitate this integration phase. This GUI supports method integrators by allowing: (*i*) the creation of a specific project for every method/procedure by specifying its typology (i.e. R, Python, Java, Knime, Octave, Linux executable, Windows executable); (*ii*) the specification of input and output parameters. Apart from the standard ones (e.g. integer, string), the platform caters for advanced ones (e.g. spatial parameters, time-related parameters, workspace related parameters); (*iii*) the browsing of the source code; (*iv*) publishing and updating the process.

Whenever an existing method or procedure is integrated into the platform, there are several benefits for both the provider(s) and the user(s): (*i*) the method becomes an asset of the overall infrastructure, and it can be added to every VRE (if the license

[‡‡]JSON Web Token is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object

selected by the provider allows it); (*ii*) the method is transparently executed by relying on the D4Science distributed computing infrastructure designed to scale horizontally. This guarantees that many runs of the same process can be executed concurrently as well as that, depending on the algorithm implementation, the same process can rely on multiple servers and cores at run-time; (*iii*) a per-process graphical user interface is automatically generated thus to facilitate the execution (e.g. the compilation of parameters) and monitoring of the process; (*iv*) a standard API based on the WPS protocol[14] is automatically generated, thus making it possible to invoke the process from existing clients programmatically; (*v*) a complete recording of every execution is automatically stored into the user workspace, including a provenance record enacting the repeatability of the specific computation; (*vi*) the method is published into the catalogue with rich metadata facilitating its discovery and use.

D4Science complements this by offering Jupyter and RStudio environments as a service when requested by a VRE. In practice, it provides a JupyterLab notebook development environment (spawn via JupyterHub) for documenting and recording analytics processes[15]. Users can integrate their notebooks into such an environment and share them with their coworkers thanks to the workspace area embedded into the environment and shared with all VRE members. Moreover, users can count on the catalogue sharing facility (c.f. Sec. 2.3) and publish the notebooks of interest. The benefits of this Jupyter-based integration approach lies in the easiness of sharing the notebooks with their working environment, thus facilitating their reuse. Similarly, it offers an RStudio development environment that, when made available in a VRE, provides its users with a working environment where access to the VRE workspace is seamlessly integrated into the files area.

In addition to that, D4Science offers the possibility to execute Java-based applications via SmartExecutor[§§], either on request or by a specific scheduling plan. Although this approach is primarily used to automate specific tasks for the operation of the infrastructure, it can be exploited for integrating and executing Java-based applications. Every application must be implemented via a plug-in of the SmartExecutor service, i.e. every plug-in must implement two specific Java classes providing information on the plug-in and implementation of the application. After passing a validation process, every plug-in is published in the infrastructure and provisioned via the provisioning service. SmartExecutor offers a REST API enabling authorised clients to request the execution of a registered plug-in or to schedule repetitive executions. There are several possibilities to define the scheduling plan, the most common is by cron. The scheduling plan can also specify directives regarding actions to take on certain happenings, e.g. skip the subsequent execution if the previous has not been terminated or failed. Moreover, it is possible to request SmartExecutor to notify a specific user or the entire community of a VRE about executing an application via social networking facilities.

2.3 | Integrating any worth sharing research object

D4Science provides its users with a highly customisable catalogue offering its content via a GUI, a RESTful API (gCat) and some standards (e.g. DCAT, OAI-PMH)[11]. This service is a crucial component of almost every VRE because it makes it possible for each community to organise a shared and searchable "research objects" space. Such a space is expected to be populated with descriptive records of any worth-sharing artefact that may or may not pre-exist the VRE. Research objects might represent datasets, software, services, processes, and the like.

The integration process, per VRE, consists of (*i*) defining the typologies of objects to be managed by the catalogue, and (*ii*) identifying the sources for existing objects to be gathered and implementing the gathering procedure.

Typologies of objects can be defined by specifying the name of each typology of interest and the attributes characterising it by an XML file (a JSON-based specification is under development). Attributes are specified by defining: the name, the type (string, time, text, boolean, number, GeoJSON), whether the attribute is mandatory or not, whether the attribute is repeatable (by specifying the maximum number of occurrences) or not, any possible default value, any controlled vocabulary to use to compile it, any validation procedure to use to check the validity of values at item record compile-time, any directive generating tags or assigning items to groups depending on the field instantiation. These fields complement the basic fields characterising every catalogue item (i.e. title, description, tags, license, author, maintainer, creation and update date). The catalogue GUI is automatically adapted to support the entering of catalogue items that match the specific typologies.

Existing objects can be gathered and published into the catalogue via standard-based harvesters or ad-hoc external approaches. The former is the catalogue system's harvesters and rely on standard-based harvesting procedures (e.g. OAI-PMH, CSW). The latter are implementations of a specific method for collecting existing content (metadata and actual content) from any target repository and publishing it into the catalogue by its RESTful API.

[§§] SmartExecutor wiki page https://gcube.wiki.gcube-system.org/gcube/SmartExecutor

The benefits of integrating existing assets of interest for the community into a unifying catalogue customised explicitly for the VRE are immediate. It enacts the development of a shared resource space where items of interest can be seamlessly discovered and accessed. Moreover, items produced in the context of the VRE contribute to populating this organised resource space. The catalogue content represents a valuable collection of resources for every designated community member that evolves with the VRE.

3 | SELECTED COMMUNITIES OF PRACTICE EXPLOITATIONS

During its lifetime, the D4Science infrastructure has been selected from several communities of practice to be the platform to develop and operate their VREs. In all the exploitation cases discussed below, a co-creation approach has been used by promoting a strong and almost continuous collaboration between the D4Science team and the specific community of practice members. This collaboration is supported by (*a*) an almost immediate delivery of every envisaged VRE with the basic functionalities and features that are ready to use; (*b*) a dynamic and shared VRE development plan implemented by feature-oriented teams including both community members and D4Science engineers. Community members often implement the new features on their own by using the integration patterns previously discussed; (*c*) a DevOps strategy where new features are released, revised, and improved with rapid iterations[16]. The release of the new features is often done in autonomy by community members, e.g. the case of new analytics methods.

3.1 | The Agri-food Community

This community has been mobilised to deal with three classes of use cases: (*i*) *agro-climatic and economic modelling*, focusing on tasks related to crop modelling and crop phenology estimation, (*ii*) *food safety risk assessment*, focusing on tasks to support scientists in the multidisciplinary field of risk assessment and emerging risk identification, and (*iii*) *food security*, focusing on tasks related to high-throughput phenotyping to support phenomics researchers to select the most suitable plant species and varieties for specific environments. A community-specific gateway has been created to provide the community with 16 specific VREs.

During the development of these VREs, the community benefited from almost all the patterns discussed before. Details on this are discussed in a separate paper[17]. The Agrodatacube Dashboard has been developed using the adaption level integration thus to make a key community asset, i.e. AgroDataCube[18], nicely manifesting in a VRE. This portlet interacts with the AgroDataCube service for data discovery and access, the DataMiner component via its WPS to execute specific analytics methods, and the workspace to save the results of the analytics tasks. Four methods for crop modelling and three for crop phenology have been integrated and made available by the data analytics platform[19]. These methods are based on either previously existing implementations (Python scripts) or new implementations (in Java) that have been transformed into executable processes. The FSK-Lab[20], a KNIME extension supporting the production and consumption of risk assessment models, has been integrated by embedding it into specific Data Miner analytics methods. Thanks to this, risk assessors are provided with easy-to-use GUIs to publish and share their models or reuse existing ones by simply using an FSKX file (a model implementation fully manageable by FSK-Lab). Specific DataMiner methods have been developed in Python to access phenomics platforms data[21], thus making it possible to use them in workflows to consume these data via the WPS protocol. The adaption pattern has been exploited to integrate the Pensoft scholarly publishing platform[22], i.e. Pensoft platform has been extended to interface with the D4Science IAM and the catalogue via the gCat REST API. This approach makes it possible to easily create *executable papers* linking to VRE resources[23]. Finally, to collaboratively manage foodborne outbreaks data, the data catalogue service was exploited, leading to implementing a reference catalogue for the specific community[24].

3.2 | The Earth Science Community

This community has been mobilised to deal with several use cases, including: (*a*) providing scientists willing to analyse data collected by EISCAT radars with a collaborative working environment, (*b*) implementing shared, standardised, and reproducible data processing and quality control procedures for long-term eddy covariance flux datasets, (*c*) providing scientists involved

in atmospheric new particle formation event analysis with computational environments for event identification and classification with built-in analysis (derivative) data FAIRification, and (*d*) providing scientists seeking to increase our knowledge of biodiversity organisation and ecosystem functions with a working environment to test models.

During the development of these cases, each leading to a dedicated VRE, the designated community mainly uses the facilities for integrating analytics methods. In particular, the algorithms for analysing EISCAT data (implemented in Octave), eddy covariance fluxes (implemented in R and C++), particle formation events (implemented in Python), and mosquito-born diseases diffusion (implemented in R) have all been nicely integrated into VREs. Details on these have been discussed in a separate paper[25]. The community has appreciated the added value in terms of usability and reuse of the shared methods (with the small integration cost) and their integration into augmented working environments.

3.3 | The Marine Science Community

Marine science is undoubtely one of the communities making the most prominent use of the D4Science facilities. A total of 20 VREs serving more than 750 users have been developed in the context of the iMarine project.[26] A total of 66 VREs serving more than 3000 users spread across 32 countries and 124 different organisations have been developed in the BlueBRIDGE project.[27] In the context of the ongoing Blue Cloud project[¶¶] VREs for the following cases are under development: (*i*) support the production and publishing of phytoplankton, zooplankton, and nutrients Essential Ocean Variables products by collating and processing diverse data scattered across several data sources; (*ii*) support a deep assessment of plankton distributions, dynamics, and fine-grained diversity to molecular resolution by focusing on species and functions discovery and exploration of genetic and morphological markers of plankton diversity and abundance; (*iii*) support the calculation and distribution of information and indicators on the environmental quality of the Mediterranean Sea (and North-East Atlantic later); (*iv*) support the development of the Fisheries Atlas (to manage public fisheries statistical data from ingestion, through harmonisation, to publication and to support state-of-the-art assessment models) and the extension of the Global Record of Stocks and Fisheries (GRSF)[28]; (*v*) support a robust and replicable environment to monitor aquaculture in marine cages and in coastal areas.

During the development of these cases, almost all the integration patterns have been exploited. In particular, extensive use of the facilities for integrating analytics methods has been pursued and highly appreciated for the ease of use. GRSF is a case primarily exploiting the catalogue service for data publishing and curation. The Marine Environment Indicators dashboard[##] is a community-specific GUI for executing a series of user-defined analytics methods via DataMiner.

3.4 | The Social Sciences and the Humanities Communities

The social science community has been mainly served by supporting the SoBigData community[29,30]. This community is willing to develop and share domain-specific analytics methods and datasets. Apart from using the catalogue to publish the so developed resources, the development of the VREs for this community essentially counted on the options to integrate existing applications by using the app integration patterns (c.f. 2.1) and the analytics methods integration (c.f. 2.2). Applications like TagMe[31], WAT[32] and SWAT[32] are now joined to D4Science by the adoption pattern. TagMe is a Java application offering its facilities by a REST API. It has been integrated by SmartGears and made available via a dedicated Virtual Research Environment hosting its GUI and the accompanying documentation.[|||] A similar approach was used for WAT and SWAT. QuickRank[33] and GATECloud[34] have been integrated by developing specific analytics methods. In particular, in the case of GATECloud, the analytics methods are simple wrappers taking care of invoking the homologous method on that platform. Twitter Monitor, an application to collect relevant messages from Twitter and share these with the SoBigData community, uses Data Miner and SmartExecutor facilities to run on D4Science[35].

Several use cases originating from the Humanities community have been supported. In particular, the PARTHENOS research infrastructure willing to strengthen the cohesion of research across several related fields associated with the humanities has been developed by relying on D4Science support. Within PARTHENOS, a complete technical framework is produced for the federation of Digital Human Infrastructures (DHI), enabling transparent access to resources managed by different DHIs and enabling the creation and operation of Virtual Research Environments[36]. The resulting PARTHENOS VRE supports researchers to perform multidisciplinary research with an online collaborative environment that integrates: built-in tools and services with

[¶¶] Blue Cloud project webpage https://www.blue-cloud.org/
[##] MEI VLab https://blue-cloud.d4science.org/web/marineenvironmentalindicators
[|||] Entity Linking toolset https://sobigdata.d4science.org/web/tagme

access to other widely-used external tools, a cloud service for storing data and documents, a powerful search mechanism for finding the various items provided (tools, data, etc.), persistent identifiers for datasets, social networking for communication.

4 | RELATED WORKS

Science gateways (SGs)[37], Virtual Research Environments (VREs)[5] and Virtual Laboratories (VLabs) are all terms used to indicate solutions aiming at providing a designated community with online research platform catering for an integrated access to *resources* (e.g. computing, software, data) of interest for the community[6,7]. However, the scope of systems falling under this definition is ample and varied. Unfortunately, there is no globally shared definition of these terms to precisely understand whether they should be considered synonyms or not. For the sake of this work, we will consider them synonyms and equally representative of the overall application domain where D4Science is. Nonetheless, differences exist in the actual implementations of the various solutions and systems. Fon instance, every D4Science-based VRE is equipped with several services facilitating communication and collaboration among the members, a feature that is not a must for all the systems. D4Science is an infrastructure supporting the development of Virtual Research Environments by using the as-a-Service paradigm, while many solutions require development and deployment activities.

In a recent study, 168 primary studies (including one on D4Science and one on its enabling system) on frameworks, models, methodologies, processes, and good practices to manage IT resources and services to realise Science Gateways have been analysed.[38] This study concludes by recommending the exploitation of cloud technologies to guarantee an "adequate management of the set of IT resources and services used to support Science Gateway environments" that is precisely the approach promoted by D4Science for its VREs as a Service.

Very often, in studies and solutions for SG and VRE technologies, there is a decoupling of the approaches focusing on the front-end part from the approaches focusing on the back-end part of these solutions. The formers use and extend gateway and portal solutions to support the development of the domain-specific User Interfaces. The latter develop frameworks enabling the composition, deployment, and orchestration of the constituents of the application. One of the results of this decoupling is that among the solutions for the development of SGs and VREs, there are heterogeneous items and diverse customisation/extension patterns.

WS-PGRADE/gUSE[39] is a framework consisting of a workflow-oriented framework that enables the development, execution, and monitoring of scientific workflows (gUSE) and a Liferay technology-based web portal of the workflow framework (WS-PGRADE). Overall, this framework has four customisation options: (*i*) developing new workflow applications and making them available for others to use by the WS-PGRADE UI; (*ii*) developing custom portlets for executing workflow applications via gUSE; (*iii*) developing custom clients for executing workflow applications via HTTP protocol; (*iv*) developing custom clients for directly submitting jobs to a distributed computing infrastructure.

The Globus platform[40] provides several common services to build SGs or web applications to support research. These common services comprise (*i*) identity management, single sign-on, and authorisation capabilities, including user profile management; (*ii*) data sharing and data discovery; (*iii*) file transfer; and (*iv*) service to service automation capabilities. SGs developers can access Globus capabilities directly through its various interfaces, offer customised branded sites exposing access to Globus functionality, and use Web interfaces via redirection from a gateway.

Apache Airavata[41] is widely used to manage application and workflow executions on a range of back-end resources by providing component abstractions for common tasks. Because of this, it is frequently cited as a solution enacting the development of SGs and VREs, although it is a platform requiring gateway administrators to configure, enable, or construct various applications and workflows to allow users to execute them. Moreover, it has to be complemented by a front-end component. For instance, the EasyGateway solution[42] was proposed to facilitate the development of SGs relying on Airavata for the back-end. This solution is comparable with the Data Miner part of D4Science VREs (cf. Sec. 2.2). The resulting gateway aims to offer facilities for experiments execution where an Airavata-based workflow implements an experiment. The Airavata team proposed its Django-based framework[43]. This framework is paired with Airavata, i.e. it counts on Airavata capabilities for authentication and authorisation, computing and storage resources configuration, registration of applications and workflows. The framework adds customisations regarding how users provide values for the inputs of an experiment, the provisioning of custom views of the output data generated, and creating wholly custom user interfaces (by a Django app).

The Agave Platform[44] offers science-oriented facilities by microservices deployed as Docker containers. Agave is designed for multi-tenancy where a tenant is a group of users, usually representing a community or organisation, who share a configuration of

the platform regarding IAM, data, services, documentation, and global settings. The platform services handle everything associated with deploying, testing, configuring, and administering a tenant, its APIs, and its users. It is possible to add their serverless component Abaco to a tenant deployment, thus extending the facilities available by implementing new functions as Docker images and registering them. Regarding the science-oriented microservices, Agave's offerings range from file-oriented storage to URLs shorteners, metadata management, data transformation, and entire "systems", i.e. physical, virtual, digital resources that stores data and run code. Systems are particularly relevant because they allow multiple users to configure and use a single shared resource, such as an HPC cluster, in the way(s) that best meet their individual needs. The platform is expected to be exploited by its command-line interface, via the development kits existing in multiple programming languages (wrappers around Agave's REST API), or via Jupyter and RStudio. The Agave team has also developed ToGo, an open-source, static web application serving as the reference science gateway for the Agave platform. Communities can clone this reference implementation to start implementing their gateways.

KubeNow[45] represents another initiative leveraging microservices to support the deployment of VREs. The emphasis here is on simplifying the deployment of a VRE in terms of several interacting microservices. At the same time, the science-oriented business logic of the VRE is left to the implementation of microservices. The KubeNow was mainly exploited to develop community-specific solutions[46].

HUBZero[47] is a platform for building the front-end part of a Science Gateway aiming at providing analytical tools and facilities to publish data, share resources, collaborate and build communities in a single web-based ecosystem. A key component is a content management system to create and share the specific "hub" content ranging from blog entries to datasets and computational tools. This platform focuses more on supporting the development of "hubs" than on promoting collaboration and sharing across hubs. It needs to be extended with other technologies[48]. DEEDS[49] is an example of a platform for supporting the entire research investigation process with data and computing tasks built on top of HUBZero.

Compared with these other efforts and initiatives, D4Science aims at providing an end-to-end solution for SGs and VREs development ad operation. In particular, the deployment of a VRE is entirely automatic thanks to the use of cloud technologies and a managed set of interacting services and technologies. Every VRE is a complete solution, including the services implementing the back-end and the front-end components. The integration patterns discussed in this paper share commonalities with some of the approaches discussed. Similarities regard the exploitation of APIs, the development of ad-hoc GUIs, the integration of entire "applications". However, the D4Science patterns aim at promoting the growth of the overall infrastructure offering and cross-fertilisation among the supported communities and cases. Whenever an application, a method, or a research object is integrated into a VRE, it is also onboarded into the infrastructure becoming an asset that can be used (under the sharing policy defined by the owner) when creating a new Virtual Research Environment.

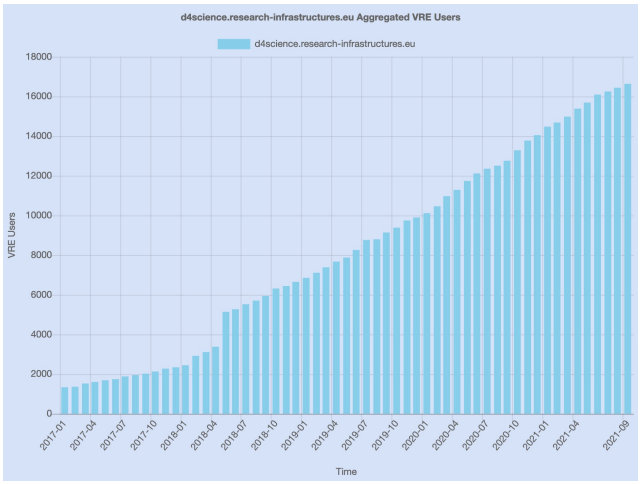
5 | DISCUSSION AND CONCLUSIONS

This paper described the VREs development approach promoted by D4Science. In particular, it advocated how a co-development strategy is fundamental to guarantee that the resulting VREs better match the expectations of their designated communities. The patterns for integrating existing applications, analytics methods, and research objects into VREs have been described by discussing the available options, the costs of alternative approaches, and the resulting benefits.

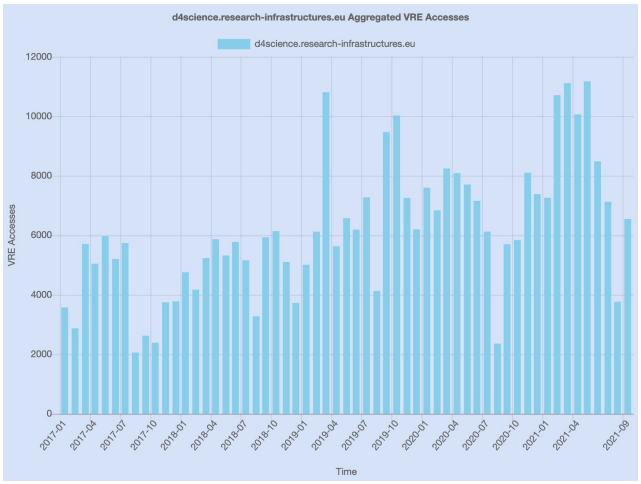
The D4Science co-creation approach provides a VRE as a service that is ready to be extended and customised with community-specific resources.

At the time of writing this paper (September 2021), a total of 167 VREs are operational, with others to come. Figure 2 reports several VREs exploitation and uptake indicators. The user base counts more than 16,600 users (Fig. 2 (a)) with a growth rate of circa 1140% since January 2017. More than 351,300 working sessions have been initiated in Jan. 2017-May 2021 (Fig. 2 (b)) with an average of more than 6100 sessions per month and a peak of more than 11,100 sessions per month. More than 1,137,148,500 analytics tasks have been initiated in the same period (Fig. 2 (c)) with an average of more than 15,974,450 tasks per month and a peak of more than 224,253,400 tasks per month. On average, every user performed more than 2850 tasks per month. More than 79,950 workspace sessions have been executed (Fig. 2 (d)) with an average of more than 1400 sessions per month. These figures give a rough picture of the uptake of the environments.

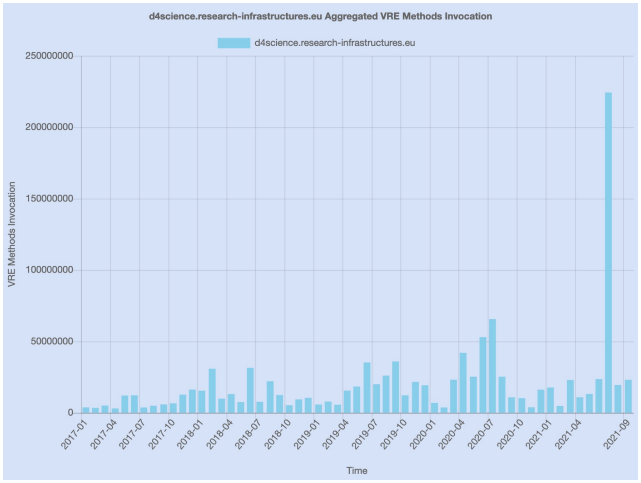
Future research directions include extending the D4Science back-end services thus to better responding to user needs. Planned forthcoming facilities include integrating open source solutions for the collaborative editing of documents such as OnlyOffice



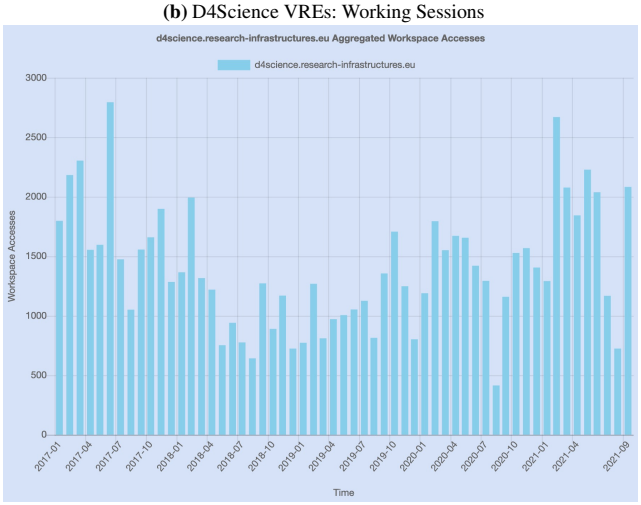
(a) D4Science VREs: User Base



(b) D4Science VREs: Working Sessions



(c) D4Science VREs: Analytics Tasks



(d) D4Science VREs: Workspace Accesses

FIGURE 2 D4Science VREs Usage Indicators from Jan. 2017 to September 2021

and Overleaf, integrating communication-oriented solutions like video conferences and chats, integrating recommender systems, reinforcing catalogue based publishing processes by implementing review and approval practices. Regarding integration patterns, micro-frontends seem a worth investigating solution[50].

ACKNOWLEDGEMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under AGINFRA PLUS project (grant agreement No. 731001), Blue Cloud project (grant agreement No. 862409), EOSC-Pillar project (grant agreement No. 857650), and SoBigData-PlusPlus (grant agreement No. 871042).

References

1. Markus ML, Keil M. If We Build It, They Will Come: Designing Information Systems that People Want to Use. *Sloan Management Review* 1994; 35(4).

2. Nelson WA, Bueno KA, Huffstutler S. If You Build It, They Will Come. But How Will They Use It?. *Journal of Research on Computing in Education* 1999; 32(2): 270-286. doi: 10.1080/08886504.1999.10782278

3. Finholt TA, Birnholtz JP. *If We Build It, Will They Come? The Cultural Challenges of Cyberinfrastructure Development*: 89–101; Dordrecht: Springer Netherlands . 2006

4. Cutcher-Gershenfeld J, Baker K, Berente N, et al. Build It, But Will They Come? A Geoscience Cyberinfrastructure Baseline Analysis. *Data Science Journal* 2016; 15. doi: 10.5334/dsj-2016-008

5. Candela L, Castelli D, Pagano P. Virtual Research Environments: an Overview and a Research Agenda. *Data Science Journal* 2013; 12: GRDI75-GRDI81. doi: 10.2481/dsj.GRDI-013

6. Barker M, Olabarriaga SD, Wilkins-Diehr N, et al. The global impact of science gateways, virtual research environments and virtual laboratories. *Future Generation Computer Systems* 2019; 95: 240 - 248. doi: https://doi.org/10.1016/j.future.2018.12.026

7. Calyam P, Wilkins-Diehr N, Miller M, et al. Measuring success for a future vision: Defining impact in science gateways/virtual research environments. *Concurrency and Computation: Practice and Experience* 2021; 33(19). doi: https://doi.org/10.1002/cpe.6099

8. Buddenbohm S, Enke H, Hofmann M, Klar J, Neuroth H, Schwiegelshohn U. Success Criteria for the Development and Sustainable Operation of Virtual Research Environments. *D-Lib Magazine* 2015. doi: 10.1045/september2015-buddenbohm

9. Ind N, Coates N. The meanings of co-creation. *European Business Review* 2013; 25(1): 86-95. doi: https://doi.org/10.1108/09555341311287754

10. Assante M, Candela L, Castelli D, et al. The gCube system: Delivering Virtual Research Environments as-a-Service. *Future Generation Computer Systems* 2019; 95(n.a.): 445-453. doi: 10.1016/j.future.2018.10.035

11. Assante M, Candela L, Castelli D, et al. Enacting open science by D4Science. *Future Generation Computer Systems* 2019; 101: 555 - 563. doi: https://doi.org/10.1016/j.future.2019.05.063

12. Wallom DCH, Turilli M, Drescher M, Scardaci D, Newhouse S. Federating Infrastructure as a Service Cloud Computing Systems to Create a Uniform E-infrastructure for Research. In: IEEE. 2015 (pp. 155-164)

13. Hardt D. The OAuth 2.0 Authorization Framework. RFC 6749, Internet Engineering Task Force; 2012.

14. Coro G, Panichi G, Scarponi P, Pagano P. Cloud computing in a distributed e-infrastructure using the web processing service standard. *Concurrency and Computation: Practice and Experience* 2017; 29(18): e4219. doi: 10.1002/cpe.4219

15. Perez F, Granger BE. IPython: A System for Interactive Scientific Computing. *Computing in Science & Engineering* 2007; 9(3): 21-29. doi: 10.1109/MCSE.2007.53

16. Wiedemann A, Forsgren N, Wiesche M, Gewald H, Krcmar H. The DevOps Phenomenon. *Queue* 2019; 17(2): 93–112. doi: 10.1145/3329781.3338532

17. Assante M, Boizet A, Candela L, et al. Realizing virtual research environments for the agri-food community: The AGINFRA PLUS experience. *Concurrency and Computation: Practice and Experience* 2020; 33(19): e6087. doi: https://doi.org/10.1002/cpe.6087

18. Janssen H, Janssen S, Knapen M, et al. AgroDataCube: A Big Open Data collection for Agri-Food Applications. agrodatacube.wur.nl; 2018

19. Knapen MJR, Lokers RM, Candela L, Janssen S. AGINFRA PLUS: Running Crop Simulations on the D4Science Distributed e-Infrastructure. In: Athanasiadis IN, Frysinger SP, Schimak G, Knibbe WJ., eds. *Environmental Software Systems. Data Science in Action - 13th IFIP WG 5.11 International Symposium, ISESS 2020, Wageningen, The Netherlands, February 5-7, 2020, Proceedings*. 554 of *IFIP Advances in Information and Communication Technology*. Springer; 2020: 81–89

20. Alba Aparicio dM, Buschhardt T, Swaid A, et al. FSK-Lab – An open source food safety model integration tool. *Microbial Risk Analysis* 2018; 10: 13 - 19. Special issue on 10th International Conference on Predictive Modelling in Food: Interdisciplinary Approaches and Decision-Making Tools in Microbial Risk Analysisdoi: https://doi.org/10.1016/j.mran.2018.09.001

21. Neveu P, Tireau A, Hilgert N, et al. Dealing with multi-source and multi-scale information in plant phenomics: the ontology-driven Phenotyping Hybrid Information System. *New Phytologist* 2019; 221(1): 588-601. doi: 10.1111/nph.15385

22. Penev L. From Open Access to Open Science from the viewpoint of a scholarly publisher. *Research Ideas and Outcomes* 2017; 3: e12265. doi: 10.3897/rio.3.e12265

23. Filter M, Candela L, Guillier L, et al. Open Science meets Food Modelling: Introducing the Food Modelling Journal (FMJ).. *Food Modelling Journal* 2019; 1: e46561. doi: 10.3897/fmj.1.46561

24. Ribeiro Duarte AS, Nielsen CL, Candela L, Valentin L, Aarestrup FM, Vigre H. Global Food-source Identifier (GFI): Collaborative virtual research environment and shared data catalogue for the foodborne outbreak investigation international community. *Food Control* 2021; 121: 107623. doi: https://doi.org/10.1016/j.foodcont.2020.107623

25. Candela L, Stocker M, Häggström I, et al. Case study: ENVRI science demonstrators with D4Science. In: Springer. 2020.

26. Candela L, Ellenbroek A, Pagano P. Virtual Research Environments Activity Report. Deliverable D6.5, iMarine; 2014.

27. Assante M, Candela L, Cirillo R, Dell'Amico A, Pagano P. BlueBRIDGE VREs Operation Activity: Final Report. Deliverable D4.5, BlueBRIDGE; 2018.

28. Tzitzikas Y, Marketakis Y, Minadakis N, et al. Methods and Tools for Supporting the Integration of Stocks and Fisheries. In: Salampasis M, Bournaris T. , eds. *Information and Communication Technologies in Modern Agricultural Development*Springer International Publishing; 2019; Cham: 20–34

29. Giannotti F, Trasarti R, Bontcheva K, Grossi V. SoBigData: Social Mining & Big Data Ecosystem. In: WWW '18. International World Wide Web Conferences Steering Committee; 2018; Republic and Canton of Geneva, CHE: 437–438

30. Grossi V, Giannotti F, Pedreschi D, Manghi P, Pagano P, Assante M. Data science: a game changer for science and innovation. *International Journal of Data Science and Analytics* 2021; 11(4): 263–278. doi: 10.1007/s41060-020-00240-2

31. Ferragina P, Scaiella U. TAGME: On-the-Fly Annotation of Short Text Fragments (by Wikipedia Entities). In: CIKM '10. Association for Computing Machinery; 2010; New York, NY, USA: 1625–1628

32. Piccinno F, Ferragina P. From TagME to WAT: A New Entity Annotator. In: ERD '14. Association for Computing Machinery; 2014; New York, NY, USA: 55–62

33. Capannini G, Lucchese C, Nardini FM, Orlando S, Perego R, Tonellotto N. Quality versus efficiency in document scoring with learning-to-rank models. *Information Processing & Management* 2016; 52(6): 1161 - 1177. doi: https://doi.org/10.1016/j.ipm.2016.05.004

34. Tablan V, Roberts I, Cunningham H, Bontcheva K. GATECloud.net: a platform for large-scale, open-source text processing on the cloud. *Philosophical Transactions of the Royal Society A* 2013; 371. doi: https://doi.org/10.1098/rsta.2012.0071

35. Cresci S, Minutoli S, Nizzoli L, Tardelli S, Tesconi M. Enriching Digital Libraries with Crowdsensed Data. In: Manghi P, Candela L, Silvello G., eds. *Digital Libraries: Supporting Open Science*Springer International Publishing; 2019; Cham: 144–158.

36. Frosini L, Bardi A, Manghi P, Pagano P. An Aggregation Framework for Digital Humanities Infrastructures: The PARTHENOS Experience. *SCIRES-IT* 2018; 8(1). doi: 10.2423/i22394303v8n1p33

37. Lawrence KA, Zentner M, Wilkins-Diehr N, et al. Science gateways today and tomorrow: positive perspectives of nearly 5000 members of the research community. *Concurrency and Computation: Practice and Experience* 2015; 27(16): 4252-4268. doi: https://doi.org/10.1002/cpe.3526

38. Sepúlveda-Rodríguez LE, Garrido J, Chavarro-Porras JC, et al. Study-based Systematic Mapping Analysis of Cloud Technologies for Leveraging IT Resource and Service Management: The Case Study of the Science Gateway Approach. *Journal of Grid Computing* 2021; 19(4): 41. doi: 10.1007/s10723-021-09587-7

39. Gottdank T. *Introduction to the WS-PGRADE/gUSE Science Gateway Framework*: 19–32; Cham: Springer International Publishing . 2014

40. Ananthakrishnan R, Chard K, Foster I, Tuecke S. Globus platform-as-a-service for collaborative science applications. *Concurrency and Computation: Practice and Experience* 2015; 27(2): 290-305. doi: https://doi.org/10.1002/cpe.3262

41. Pierce ME, Marru S, Gunathilake L, et al. Apache Airavata: design and directions of a science gateway framework. *Concurrency and Computation: Practice and Experience* 2015; 27(16): 4282-4291. doi: https://doi.org/10.1002/cpe.3534

42. Galizia A, Roverelli L, Zereik G, Danovaro E, Clematis A, D'Agostino D. Using Apache Airavata and EasyGateway for the creation of complex science gateway front-end. *Future Generation Computer Systems* 2019; 94: 910-919. doi: https://doi.org/10.1016/j.future.2017.11.033

43. Christie M, Marru S, Abeysinghe E, et al. An Extensible Django-Based Web Portal for Apache Airavata. In: PEARC '20. Association for Computing Machinery; 2020; New York, NY, USA: 160–167

44. Dooley R, Brandt SR, Fonner J. The Agave Platform: An Open, Science-as-a-Service Platform for Digital Science. In: PEARC '18. Association for Computing Machinery; 2018; New York, NY, USA

45. Capuccini M, Larsson A, Carone M, et al. On-demand virtual research environments using microservices. *PeerJ Computer Science* 2019; 5: e232. doi: 10.7717/peerj-cs.232

46. Peters K, Bradbury J, Bergmann S, et al. PhenoMeNal: processing and analysis of metabolomics data in the cloud. *Gigascience* 2019. doi: https://doi.org/10.1093/gigascience/giy149

47. McLennan M, Kennell R. HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering. *Computing in Science & Engineering* 2010; 12(2): 48-53. doi: 10.1109/MCSE.2010.41

48. McLennan M, Clark S, Deelman E, et al. HUBzero and Pegasus: integrating scientific workflows into science gateways. *Concurrency and Computation: Practice and Experience* 2015; 27(2): 328-343. doi: https://doi.org/10.1002/cpe.3257

49. HewaNadungodage C, Catlin AC, Bejarano A, et al. The DEEDS platform: Support for integrated data and computing across the research lifecycle. *Future Generation Computer Systems* 2020; 111: 793-805. doi: https://doi.org/10.1016/j.future.2019.10.031

50. Peltonen S, Mezzalira L, Taibi D. Motivations, benefits, and issues for adopting Micro-Frontends: A Multivocal Literature Review. *Information and Software Technology* 2021; 136: 106571. doi: https://doi.org/10.1016/j.infsof.2021.106571