H2020-EINFRA-2017

EINFRA-21-2017 - Platform-driven e-infrastructure innovation

DARE [777413] "Delivering Agile Research Excellence on European e-Infrastructures"

Decomposed User Stories II

Project Reference No	777413 — DARE — H2020-EINFRA-2017 / EINFRA-21-2017
Deliverable	D3.2: Decomposed User Stories II – M27
Work package	WP3: Large-scale Lineage and Process Management
Tasks involved	T 3.2: Decomposed User Stories II
Type	R: Document, report
Dissemination Level	PU = Public
Due Date	30/04/2020
Submission Date	30/04/2020
Status	Draft
Editor(s)	Rosa Filgueira (UEDIN), Alessandro Spinuso (KNMI)
Contributor(s)	Sissy Themeli (NCSRD), Malcolm Atkinson (UEDIN)
Reviewer(s)	Xenofon Tsilimparis (NCSRD)
Document description	Update on the refinement and realisation of the core DARE's features

Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
1.0	11/03/2020	Table of Contents and Outline	Rosa Filgueira, Alessandro Spinuso
1.0	19/03/2020	Sections 3 and 3.2 on DARE Capabilities and and Lineage Management	Alessandro Spinuso
1.0	23/03/2020	Section 3, Lineage exploration subsection	Alessandro Spinuso
1.0	03/04/2020	Section 3, Prepare Scientific Workflows (3.1)	Rosa Filgueira
1.0	05/04/2020	Section 3, Prepare Scientific Workflows (3.1)	Rosa Filgueira
1.0	06/04/2020	Dispel4py Information Registry (3.3)	Sissy Themeli
1.0	06/04/2020	Submit the workflow and applications (3.5)	Sissy Themeli
1.0	09/04/2020	Introduction and refinement to Section 2.2	Alessandro Spinuso
1.0	16/04/2020	wrote section 2.8	Malcolm Atkinson
1.0	16/04/2020	Addressed Comments on Sections 1, 2 and 2.7	Alessandro Spinuso
1.0	20/04/2020	Executive Summary	Alessandro Spinuso
1.0	20/04/2020	Drafting section 2.5	Rosa Filgueira
1.0	21/04/2020	Add information on API calls in section 2.5	Sissy Themeli
1.0	21/04/2020	Workflow monitoring section 2.6	Sissy Themeli
1.0	21/04/2020	Refinement of section 2.5. 2.6, conclusions and references.	Rosa Filgueira
1.0	22/04/2020	Refinements in several sections	Malcolm Atkinson

Executive Summary

DARE is an ambitious project that aims to provide novel approaches for creating and using data-powered methods at the frontiers of today's research and innovation. DARE's central goal is to support research developers – domain-expert software developers – to transparently make use of European e-infrastructures, research infrastructures and other platforms and software in order to create data- and computationally-intensive applications for their domains. DARE aims to achieve these goals by providing much needed technology and methodology aligned with EOSC developments. This document presents the progress in refining the functionalities offered by the different components of the platform, until their implementation and foreseeable improvements.

In Section 2, we start with recalling the methodology, which is inspired by the *Agile* approach of formulating value-centered stories to drive the implementation of a system. We reported some of the stories in *D3.1.18* [1]. These were specified in cooperation with the communities represented by WP6 and WP7. In this deliverable we move forward in the process, by identifying the capabilities of the DARE platform that are used by the research-developers to realise the stories. Section 2 is divided into several subsections, one for each capability, where we provide a description of the core functionalities. These include relevant details and challenges about their technical implementation.

Finally, in Section 3, we report our Conclusions. Here we highlight the challenges of identifying those technical efforts that would foster the improvement of the platform towards the achievement of sustainability goals.

Table of Contents

List of Terms and Abbreviations

Abbreviation	Definition
AAI	Authentication, Authorisation and Identity
ACS	Analysis of Climate simulations on-demand
CO	DARE Component
CWL	Common Workflow Language
DKB	DARE Knowledge Base
EGI	European Grid Infrastructure
EPOS	European Plate Observing System
ES	Ensemble Simulations
FAIR	Findable, Accessable, Interoperable and Reusable
IPCC	Intergovernmental Panel on Climate Change
KB	Knowledge Base
MVV	Monitoring and Validation Visualiser
OGC	Open Geospatial Consortium
RA	Rapid Ground Motion Assessment
SS	Seismic Source Characterisation

1 Introduction

After the formulation of the communities' user stories reported in D3.1_M18 [1], in this deliverable we illustrate details about the core capabilities of the DARE platform. We will shortly remember how the Agile artifacts have been used for the identification of the development and integration actions on DARE's components. These aim to deliver the most relevant value for the users of the DARE platform. We will cover the support for the development and the execution of different classes of workflows, their monitoring and the management of the users' methods and results, including the representation of the DARE components and conceptual elements within a comprehensive knowledge base. For each of these capabilities we will describe the core features that are available to the communities for the realisation of their use cases.

2 DARE Capabilities and Features

In this section we describe the features provided by the main DARE capabilities and the impact on the implementation of the components. These have been identified by analysing the user stories in D3.1_M18 [1]. In Figure 1 we show again a schema that summarises the methodology used. Capabilities, such as the preparation of workflow applications, management of lineage and data, and the deployment and execution of experiments, are translated into Features. These serve the Use Cases and stories of the Communities.

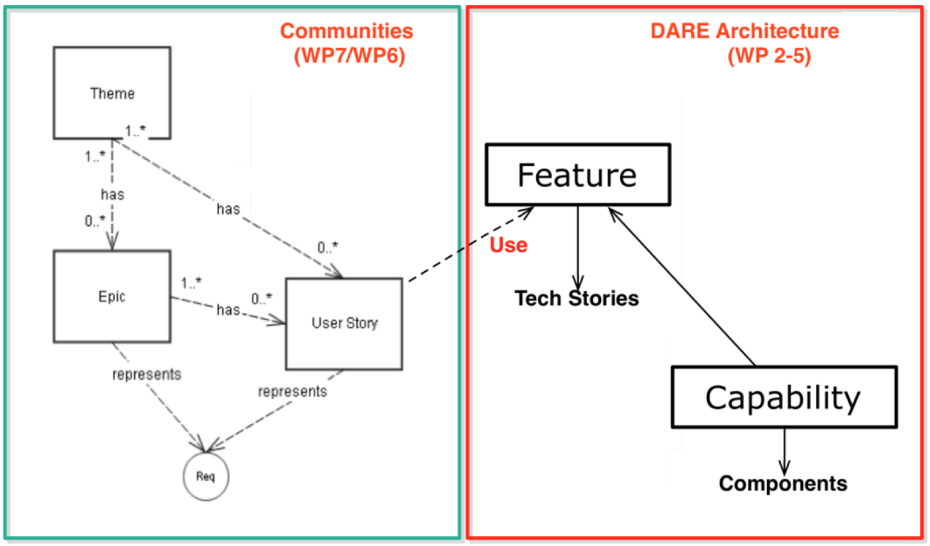

Figure 1. Mapping from Community User Stories to DARE Features and Capabilities. The implementation of the user-stories requires one or more Features. These contribute to realisation of the Capabilities that characterise the DARE platform.

2.1 Prepare Scientific Workflows (UEDIN)

Scientific workflows are an important abstraction for the composition of complex applications in a broad range of domains, such as astronomy, bioinformatics, climate science, and others. Workflows provide automation that increases the productivity of scientists when conducting computation-based studies. Automation enables adaptation to the changing application needs and resource (compute, data, network) behavior. In the past two decades, a number of Workflows Management Systems (WMSs) have been developed to automate the computational and data management tasks, the

provisioning of the needed resources, as well as different execution models required by today's scientific applications.

In order to select which WMS are going to be used/supported by DARE, we first evaluated the computational and data analysis pipelines of our user stories (e.g. Seismology and Climate user stories). These normally consist of sequential and concurrent tasks that process streams of data. Therefore, the execution models our WMS are:

- *Task-Oriented*: In this model, execution of tasks in the workflow flow from one to the other, based on the lines that connect the tasks and the logic that determine how the workflow flows from one task to the next. Essentially, in this workflow, each task waits for the preceding task to complete. This is a typical workflow model where a simulation phase is followed by a post-processing phase.

- *Streaming-Oriented*: In this model, the WMS is able to execute tasks as soon as the required resources are available, so several tasks can be running concurrently while they have data to process. This model supports workflows that process streaming data through multiple stages, for example applying different feature detection algorithms.

Based on this characterization of WMS, we have selected two WMS for our applications (one per execution model):

- *dispel4py*, a Python-based streaming-oriented framework, which describes abstract workflows for data-intensive applications, which are automatically translated to the selected enactment targets (Apache Storm, MPI, Multiprocessing, etc.) at run time.

- *The Common Workflow Language (CWL[1])*, a YAML-based specification language for describing analysis task-oriented workflows and tools in a way that makes them portable and scalable. Steps in CWL workflows are described with metadata about command line parameters, and often provided as Docker images.

To show how we have prepared our scientific workflows, we focus on one of the User Stories for the Seismological Use Case. Specifically, on the Rapid Ground Motion Assessment application (RA) introduced in [10], which its user stories have been previously addressed in D3.1_M18 [1] . RA aims to model the strong ground motion after large earthquakes, in order to make rapid assessment of the earthquake's impact, in the context of emergency response. It has five main steps (see Figure 2): (1) to select an earthquake gathering the real observed seismic wavefield, (2) to simulate synthetic seismic waveforms corresponding to the same earthquake using SPECFEM3D; (3) to pre-process both synthetic and real data; (4) to calculate the ground motion parameters for synthetic and real data; (5) to compare them with each other by creating shake maps.

[1] https://www.commonwl.org/

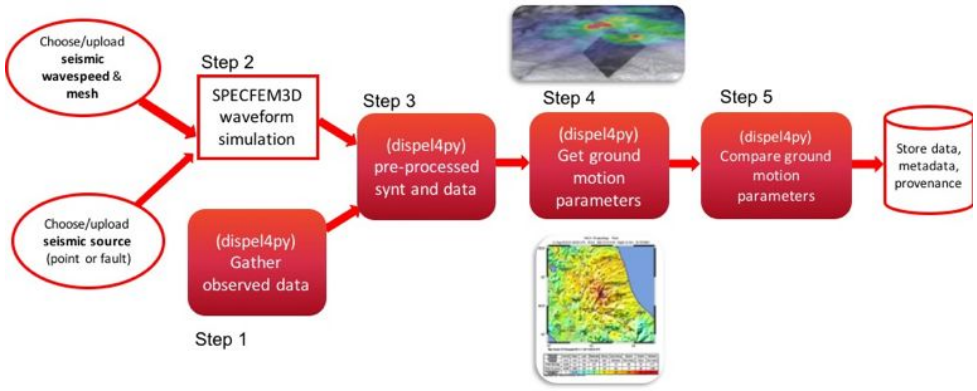

Figure 2. Rapid Ground Motion Assessment application (RA).

We built a dispel4py workflow to represent each RA step [2] as a streaming pipeline application except for the generation of the synthetic data, since SPECFEM3D is a MPI parallel application on its own. As an example, Figure 3 shows the dispel4py workflow for preprocessing synthetic and real dataset. To encode this workflow, we first created all the Processing Elements (PEs) to represent the computing activities of this step. Later we created a graph for specifying the ways in which PEs are connected and hence the paths taken by data. All PEs are concurrently executed while this workflow is running, being the only requirement that each PE has available its input data. Otherwise, PEs will wait until they have more data to process.

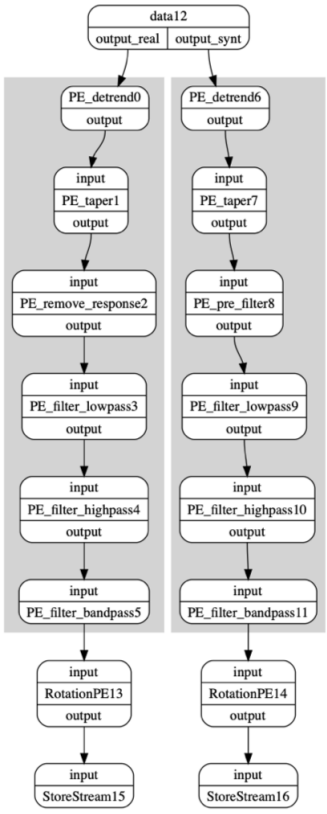

Figure 3. dispel4py pre-processing workflow, which corresponds to the third step of the RA.

CWL was used in the RA use case for describing and managing the execution of SPECFEM3D, which follows a task-oriented pattern, as we can see in Figure 4. Each SPECFEM3D task has to wait until the previous task has completed in order to start. Therefore, we could not use dispel4py for encoding this application, and CWL was selected instead.

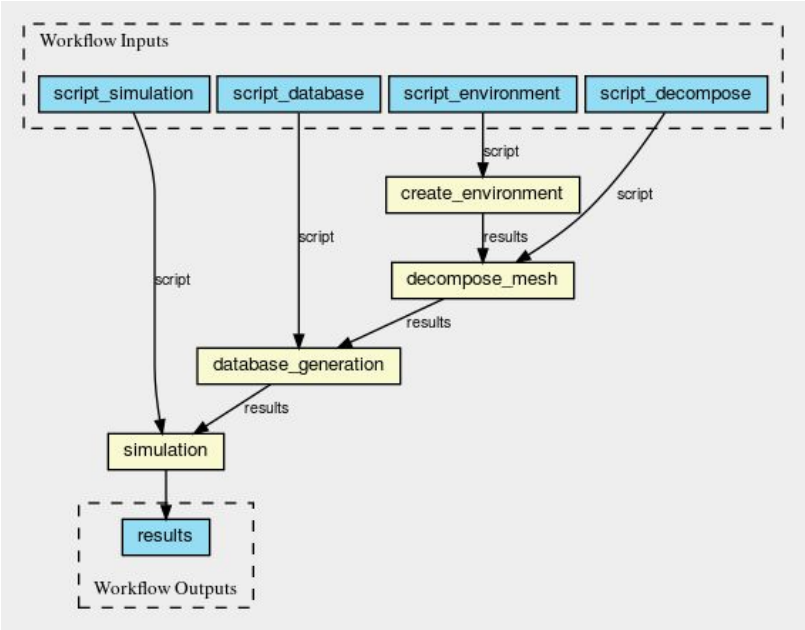

Figure 4. CWL workflow for managing the execution of SPECFEM3D .

2.2 Lineage Management (KNMI) (CO-Provenance)

This capability can be further decomposed into **generation** and **acquisition** of lineage data.

The **generation** had largely impacted over the dispel4py processing API, which has been extended with additional features in support of configuration of the lineage output, aiming at customisable metadata injection and precision. Deliverable D3.5 [29] describes this action with more technical details. These have been further explained in literature [13]. Furthermore, DARE is progressing with the adoption of CWL for the implementation of those use cases which benefit from a *task* oriented workflow. CWL already supports the automated generation of lineage as an optional configuration. It adopts CWLProv [14], which is a particular specialisation of PROV. Thus, DARE applications implemented in CWL (*i.e.* the SPECFEM3D workflow described in the RA context above [10]) and exposed through the DARE API have to be adapted. The container of the workflow has to be updated with parameters that enable the activation of the provenance generation and its upload to S-ProvFlow.

The lineage **acquisition** capability of DARE has also been improved in terms of its performance and resilience. In Figure 5 we show the deployment architecture and the communications between the S-ProvFlow components. Message and failover queues (S-Prov Queue) have been implemented to detach the workflow execution from the direct access to the provenance database services. This has the advantage of delegating to the queue those mechanisms that can recover from a temporary

unavailability of the provenance API (S-Prov API), preventing messages loss. Moreover, such architecture concentrates in the queue the overhead brought by authenticating and storing the provenance messages into the database, reducing the impact on the workflow's execution. Authentication (AAI) is integrated via the adoption of delegation tokens. These are sent from the workflow to the queue, which thereby uses them to authenticate to the API.

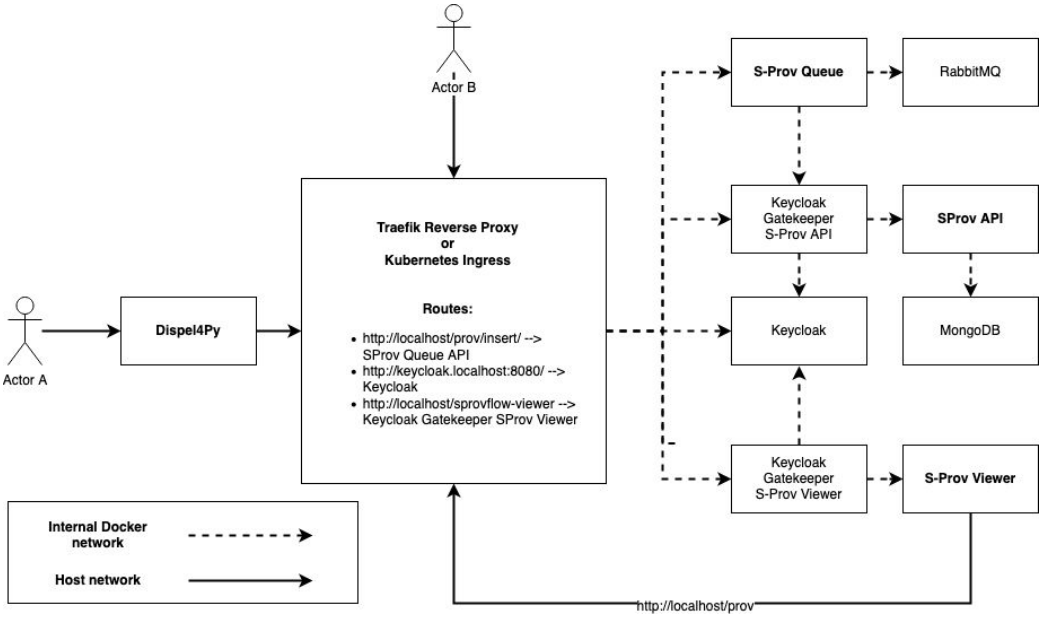

Figure 5: S-ProvFlow: microservice deployment of the S-ProvFlow system. It includes message queues and authentication. Actor A is the workflow developer interacting with dispel4py, while Actor B accesses the provenance information through the S-PROV tools.

In respect to the acquisition of CWLProv, the lineage API is extended with a new */import* feature. The method ingests CWLProv files and maps them to the JSON format used by s-ProvFlow to store lineage in MongoDB according to the S-PROV ontology. This will give the opportunity to benefit from the S-ProvFlow system's archiving, advanced and high-level queries and tooling. Given the complexity of the CWLProv format, the coverage and generic support obtained by the mapping is incrementally improved, depending on the requirements of the use cases of WP6 and WP7 [9, 10].

2.3 Register PEs and workflow in a repository (CO-Catalogue) (NCSRD)

In recent years, modern science relies more than ever on large-scale data and computing and human resources distributed in a large number of different countries. A modern scientist is required to make use of disparate computing resources (high-performance (HPC) facilities, private institutional resources, etc.), process, archive and analyze results stored in different locations as well as collaborate with other scientists. Therefore, they need to store their methods, datasets and results in order to reuse or share them with their colleagues.

DARE platform provides an execution environment for each application domain's research developers in order to execute their workflows, written in the dispel4py workflow language. In order to enable collaboration and reusability of the scientists' methods it is necessary to provide workflow storage and versioning to the users of the platform. This is achieved through the dispel4py Information Registry. It is one of the main platform components, used prior to a workflow execution. Its

functionality is wrapped and provided to the users through a RESTful Web Service. dispel4py's main concepts are stored in the Registry, e.g. PEs and PEs Implementations.

Before executing a workflow, users should make use of the Registry Service. First, they should create their own workspace where their methods / workflows will be stored. Once users own a workspace, they are able to register their dispel4py PEs and workflows. The Registry component allows the users to refer to their registered workflows by name and thus enabling the reusability of the code and the collaboration between scientists.

2.4 Submit the workflow and applications (CO-API) (SCAI) (NCSRD)

The DARE platform uses the Keycloak Service as AAI provider. Users should first be authenticated in order to use the platform and execute their workflows. The Keycloak component gives the possibility to use third-party identity providers, such as EGI or B2ACCESS. Once users are identified in the system, users need first to create or reuse a workspace and inside it register the necessary PEs in the Registry. PEs that are stored in the Registry can be reused in future experiments/executions by providing the PE name. The DARE platform provides a testing environment, named Playground, in order to execute workflows with immediate diagnostic information and direct control accurately simulating a dispel4py workflow execution. This accelerates development and improves research developers' powers to investigate issues.

After having fixed all possible errors in their workflows, users can proceed with the workflow execution via the official endpoint of the DARE Execution API. The DARE platform gives the possibility to the users to execute MPI jobs and generates multiple new containers to handle the specific request. The execution can be monitored via the Execution API and once the execution is finished, the users can request the execution logs and the produced results from the respective endpoints of the Execution API. Figure 6 depicts a dispel4py workflow execution in the DARE platform using the official Execution API.

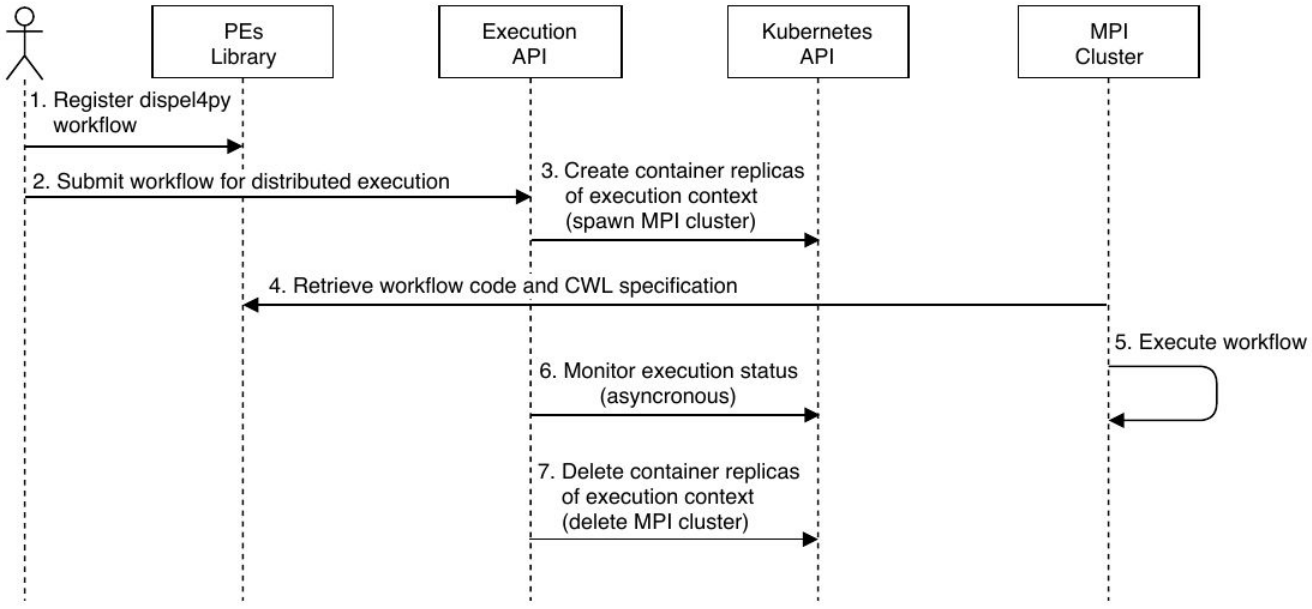

Figure 6. The executions of a dispel4py workflow using the DARE API

2.5 Platform's data storage (CO-DataStore) (SCAI) (NCSRD)

As we introduced in the previous subsection, the DARE API acts as a gateway between the cloud deployed DARE platform components and the interactive user interfaces. Among the different calls, which are defined in ID2.2 [22], we would highlight the ones that allow us to upload/download data to the DARE platform and specify the input parameters necessary for running a dispel4py workflow (see Table 1).

`create_workspace(clone, name, desc, creds)`	Create a workspace using dispel4py registry API
`submit_d4p(impl_id, pckg, workspace_id, pe_name, n_nodes, token, creds, reqs=None, **kw)`	Spawn MPI cluster and run dispel4py workflow
`upload(token, path, local_path, creds)`	Upload data into a working environment
`submit_specfem(n_nodes, data_url, token, creds)`	Spawn an MPI cluster and run a specfem workflow
`myfiles(token, creds)` and `files_pretty_print(_json)`	List the uploaded files
`pod_pretty_print(_json)`	Monitoring container status
`download(path, creds, local_path)`	Downloads a file using exec-api filesystem reference.
`__list()`	Lists all the directories of a user

Table 1: DARE API calls for the specifying of input parameters, uploading and download files and creating workspaces.

The upload API endpoint allows platform users to upload and store in the DARE platform's Shared File System necessary files for their workflow execution. The users should provide the local path of the file, the base url of the API (e.g. https://testbed.project-dare.eu) and optionally a directory name (i.e. the path parameter of the upload function). Finally, users should be authenticated in the platform and provide their token in all API calls.

Users in the platform have their own directory, named after their username (the one they use in the platform for authentication). Inside their files are stored under uploads and runs directories. All uploaded files will be stored under the uploads directory. As mentioned above, during the upload the users can specify a specific directory to store their files (this directory will be generated under the uploads directory).

The submit_d4p function makes use of the run-d4p API call to execute a dispel4py workflow. Users should provide Dispel4py Information Registry related data such as the PE name, the workspace id

and the package, dispel4py parameters (number of processes, number of nodes, target, input_data etc) and optionally a requirements.txt file.

In case that users provide a requirements file, the system executes their workflow using a virtual environment where only the requested libraries (with the requested version) are stored. In this way, the users can execute their workflows in an environment similar to their local where they have initially tested the workflow.

The DARE platform API provides endpoints to list folders and files in the Shared File System. The _list function lists all the directories of a user, i.e. the execution and upload directories. On the other hand the myfiles function lists the files in a specific folder.

For example, after a workflow execution, users should list their directories. From the list returned by the API, users should select the most recent execution directory (directory names have a timestamp as to be easily sorted based on the execution time) and provide the path in the myfiles function. In this way, users can check the output files inside an execution directory. Finally, users can download files from the platform by specifying the path to the file (retrieved from the _list and myfiles functions). To download files, users should provide a local path to store the file, the file path in the DARE platform and of course the authentication token.

2.6 Monitoring and Lineage exploration (CO-Processing) (CO-Provenance) (KNMI)(NCSRD)

After every call, we can monitor the status of the job and see and download the output. The DARE API provides a monitoring function to the platform users (see Table 2).

`monitor(creds): Monitor`	`Monitor a dispel4py workflow run`

Table 2: DARE API for monitoring a dispel4py workflow.

Once a workflow execution starts, users can monitor the docker containers spawned for the requested job and check when the job is finished. The DARE platform is deployed using Kubernetes and therefore all docker containers are orchestrated and managed by Kubernetes. By using the Kubernetes API, the DARE platform can retrieve information on the containers, spawn new docker containers etc. Thus, when users require information on their execution, the DARE API makes use of Kubernetes API to find the docker containers associated with a specific execution.

2.6.1 Lineage exploration

The S-ProvFlow system offers a visual tool (Monitoring and Validation Visualiser - MVV), that enables different sorts of operations through the interactive access and manipulation of the provenance information. These include monitoring of the progress of the execution, discovery of data and runs, filtering, data preview, download and staging. The tool displays the activity of the workflow while being executed. The view can be updated at runtime and its content is dynamically fetched from the API. Most of the features have been already introduced in D3.7 [15]. However, several improvements

have been implemented, especially addressing the expressiveness and usability of the search capabilities of the API and the tools. The API interface of the search and discovery methods has been completely re-designed in order to support a simple syntax that allows users to formulate queries over multiple terms' using single values, ranges or lists. This is reflected in the tools, as shown in Figure 7.

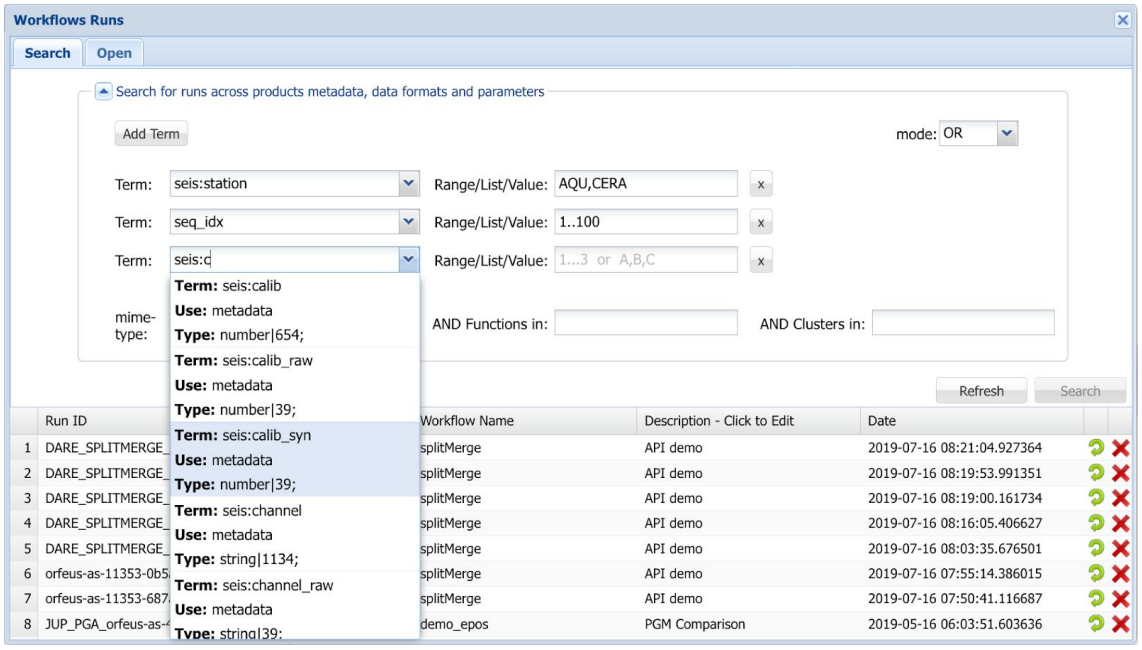

Figure 7: Workflow Execution search panel. This panel is used to discover runs of interest based on parameters and metadata values. Users can enter expressions indicating lists or ranges of values. For instance, the image shows comma separated values to express a list in the case of the *seis:station*, and three dots to express a range, as for the *seq_index* term.

In addition to search functionalities, we also started the implementation of a new solution to identify users in the lineage. A common approach had to be agreed by the partners developing the different DARE services, taking also into account the GDPR regulations of separating "by design" the users' anonymous Ids from their personal details. This solution required DARE to integrate login and userId resolution within the MVV and to enable authentication from this to the s-ProvFlow API. The implementation makes sure that logged-in users would be automatically directed to their lineage data.

2.7 Domain Context Representation (UEDIN)

The successful conduct of research is a complex combination of well-established knowledge with new knowledge representing insights, new approaches and new methods. Paul Edwards [16] refers to the established knowledge on which research depends and against which it is validated as the "global knowledge infrastructure". Education, training and long-term collaboration develop an understanding and appreciation of a relevant view of it as an underpinning culture. To bring it into play in formal models and methods requires its codification through international collaboration, standardisation and adoption, as undertaken by the OGC, IPCC and EPOS [17-20], etc. This global knowledge infrastructure inevitably changes very slowly and achieving adoption of changes takes substantial effort. Consequently, Venki Ramakrishnan [21] writing about his long Nobel-Prize winning campaign to understand the ribosome, asserts that it is not possible to explore new science while embedded in organisations complying with the global knowledge infrastructure.

DARE sets out to support research in such deeply embedded contexts because our communities are in global and multinational consortia and they deliver evidence and production services that depend on such global knowledge infrastructures. At the same time DARE promises to deliver research agility, so that exploration and exploitation of new ideas can be very rapid, responding to urgent needs or rapidly exploiting the potential of new data and computational power. To do this DARE proposes a new structuring methodology for knowledge bases (KBs) that it presented at the 2019 eScience conference [12]. That was further developed in ID2.2 [22] and is now being implemented as the DKB [25, 26]. This introduces a Context as a means of denoting a zone of interest in the KB as shown in Figure 8.

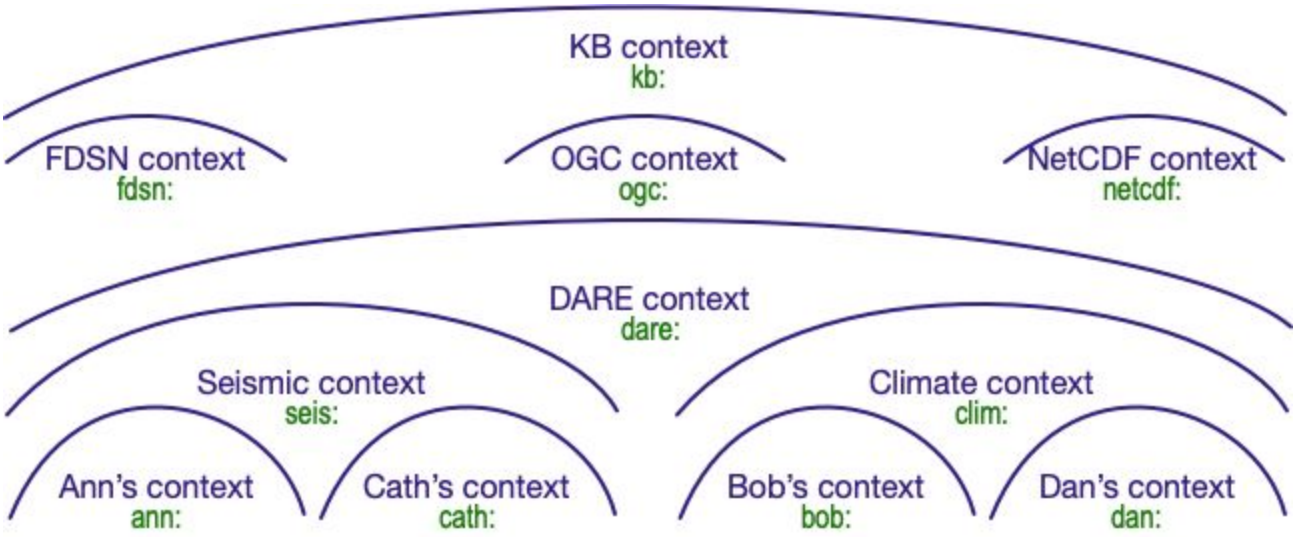

Figure 8: Showing the partitioning of a knowledge base into contexts. Those higher in the figure denote widely shared authoritative knowledge predominantly imported from the global knowledge infrastructure. Those lower contexts denote more localised knowledge agreed upon by groups to support collaboration. At the lowest level are individual contexts. These may be stable to support routine tasks and methods or they may be evolving rapidly as their users explore new ideas.

The established widely used contexts are relatively stable and their maintenance is amortised over large communities, but with potential requirements for local tailoring as each version is imported. Group contexts develop as fast as a group wishes and their research developers can manage. Again, they will import as far as possible, e.g., relevant tools and software libraries, to limit local maintenance costs, but their tailoring and packaging may be much more substantial and depend on the group's insights and ingenuity. The two forms of specific contexts (at the lowest level in Figure 8) have substantially different behaviour. Those tailored for routine procedures, e.g., archiving evidential data. Normally, a community requires to keep these stable over long-running campaigns, while the digital context surrounding them evolves rapidly. They will be anchored to very specific terms and encodings, which their research developers may need to take action to preserve. The research-led contexts will exploit new opportunities as they emerge or as they are created. To avoid having to do more than necessary in any context, they each have a user-controlled search path about where else to look. This greatly accelerates induction of new practitioners and context building. Such requirements are needed much more widely than just the DARE communities, e.g. in [25] the communities running museums need a context for their museum's procedures and local information as well as more global shared authoritative contexts, such as botanic anatomy and taxonomy, to classify their curated natural objects.

Today in DARE the KB is a loosely coupled assembly of the registry, the data catalogues and the provenance collection as reported in [26]. It delivers the mix of required elements of the global knowledge infrastructure and the required local and transitive information, such as the parameters needed, the methods to run and their mapping to the currently available computational and data infrastructure. This is inevitably complex and requires informatics experts to help in its assembly and use. We believe that by pioneering a DKB with harmonised information and contextual structuring our application-domain experts and developers will be able to be much more self-sufficient and productive. They should be able to formulate their own conceptual structures as they develop their ideas, with corresponding methods and collections directly under their control. As explained in [24] this is essential, both for agility, enabling them to respond to emergencies and lead in their research, and for sustainability, enabling them to continue and develop with affordable amounts of support. Co-design and co-development of use cases with DARE's communities will demonstrate this potential.

When researchers supported by the DARE platform have produced results they wish to share, they will need to present them as part of the global information infrastructure, which is currently adopting FAIR principles [27] often presented as digital research objects [28]. Thus the DKB has to support mapping from these surrounding infrastructures and mappings to and participations in those infrastructures to support the work of a mature research community.

3 Conclusions (KNMI)(UEDIN)

In this deliverable we have identified the capabilities of the DARE platform that are being used by the research-developers to realise the stories consistently across our application domains. For each of the capabilities, we have provided a description of the core functionalities, including the relevant details and challenges about their technical implementation. They integrate technologies in ways that enable new types of interaction with the underlying systems and e-infrastructures, making it easier for research developers and scientists to describe and execute data-driven computational experiments. With these new capabilities, we are providing researchers, engineers and decision makers with a methodology, research environment and tools to:

1) collaborate across boundaries,
2) exploit data effectively but compliantly,
3) use and adapt the increasing computational power, and
4) combine stability with agility.

The core motivation for the implementation of generic functionalities is to scale the adoption of DARE to more scientific communities. This would foster further improvement of the platform and thereby its future sustainability. This is also bound to the choice of building the system on a selection of technologies that are becoming a *de-facto* standard and that will be well sustained by their development communities and commercial partnerships. This would guarantee DARE to be running on constantly improved versions of cloud and containerised middleware. Finally, the collection of features presented have been built on components that heavily rely on containerised software, workflow systems and lineage information. This opens a concrete path towards an integrated reproducible environment, that will be further refined by new use cases aiming at the automated generation and dissemination of FAIR scientific results.

References

[1] Rosa Filgueira et al. *DARE Deliverable D3.1.18, Decomposed User Stories and Tooling.* http://project-dare.eu/wp-content/uploads/2019/08/DARE_D3.1-Decomposed-User-Stories-and-Tooling_final.v1.0.pdf

[2] Rosa Filgueira. "*DARE Platform: Enabling Easy Data-Intensive Workflow Composition and Deployment*."RO-12 at *Workshop on Research Objects (RO 2019), 24 Sept 2019, San Diego, CA, USA.* https://doi.org/10.5281/zenodo.3357806

[3] *Manifesto for agile software development*, 2001, http://agilemanifesto.org

[4] Sutherland J, Schwaber K, *The Scrum Guide,* 2018, http://scrumguides.org

[5] Brenner P, *A Technical Tutorial on the IEEE 802.11 Protocol, Breezecom Wireless Communications edition,* 1997.

[6] Malcolm Atkinson, et al. *DARE Deliverable D2.1-M12 Dare Architecture and Technical Positioning,* 2018, http://project-dare.eu/wp-content/uploads/2019/03/D2.1-DARE-Architecture-and-Technical-Positioning-I_final_draft.pdf

[7] Christian Pagé, et al. *DARE Deliverable D7.1 Requirements and Test Cases I*, 2018, http://project-dare.eu/wp-content/uploads/2019/03/D7.1-Requirements-and-Test-Cases-I_Final_draft.doc.pdf

[8] Andreas Rietbrock, et al. *DARE Deliverable D6.1 Requirements and Test Cases I*, 2018, http://project-dare.eu/wp-content/uploads/2019/03/D6.1-Requirements-and-Test-Cases-I_final_draft.pdf

[9] Christian Pagé, et al. *DARE Deliverable D7.3 Pilot Tools and Services, Execution and Evaluation Report*

[10] Federica Magnoni, et al. DARE *Deliverable D6.3 Pilot Tools and Services, Execution and Evaluation Report,http://project-dare.eu/wp-content/uploads/2019/08/DARE_D6.3-Pilot-Tools-and-Services-Execution-and-Evaluation-Report-I_final.v1.0.pdf*

[11] Malcolm Atkinson, et al. *Comprehensible Control for Researchers and Developers facing Data Challenges has been accepted for e-Science 2019*. IEEE eScience 2019, September, San Diego, US https://ieeexplore.ieee.org/document/9041709

[12] Alessandro Spinuso. DARE *Deliverable D3.3 Data Lineage Services (Software Report)* *http://project-dare.eu/wp-content/uploads/2019/08/DARE_D3.3-Data-Lineage-Services-I_final.v1.0.pdf*

[13] Alessandro Spinuso, Malcolm Atkinson and Federica Magnoni, *Active provenance for Data-Intensive workflows: engaging users and developers*, Proceedings of the BC2DC workshop IEEE eScience conf. 2019. https://ieeexplore.ieee.org/document/9041815

[14] Soiland-Reyes, Stian, et al. "Capturing interoperable reproducible workflows." *Workshop on Research Objects: Workshop at IEEE eScience 2018*. https://zenodo.org/record/1312623#.Xp7dN5rTWjQ

[15] Alessandro Spinuso et al. *D3.7 - Integrated Monitoring and Management Tools I,* http://project-dare.eu/wp-content/uploads/2019/03/D3.7-Integrated-Monitoring-and-Management-Tools-I_final_draft.pdf

[16] Paul N Edwards, *A vast machine: computer models, climate data and the politics of global warming*, MIT press 2013, Introduction page xiv. https://mitpress.mit.edu/books/vast-machine

[17] Luca Trani . *A methodology to sustain common information spaces for research collaborations*, PhD thesis, University of Edinburgh, 2019. https://era.ed.ac.uk/handle/1842/36139

[18] Luca. Trani, Rosanna. Paciello, etc. D. Ulbricht and the EPOS IT Team, *Representing Core Concepts for solid-Earth sciences with DCAT – the EPOS-DCAT Application Profile*, Geophysical Research Abstracts 2018. https://www.epos-eu.org/representing-cross-disciplinary-knowledge-solid-earth-sciences-epos-dcat-ap

[19] Luca Trani., Malcom Atkison, etc. *Establishing core concepts for information-powered collaborations*, Future Generation Computer Systems 89, 421–437, 2018.

[20] Luca Trani, etc. (2018). *EPOS-DCAT-AP: a DCAT Application Profile for solid-Earth sciences*. In 2018 Fall Meeting AGU. Abstract IN31B-33. https://ui.adsabs.harvard.edu/abs/2018AGUFMIN31B..33P/abstract

[21] Venki Ramakrishnan, *Gene Machine*, Oneworld Publications, 2018 page 263. https://oneworld-publications.com/gene-machine.html

[22] Malcolm Atkinson; Rosa Filgueira, Andre Gemünd, etc. *DARE Architecture and Technology*, Technical report ID2.2, March 2020, DOI https://doi.org/10.5281/zenodo.3697898

[23] Malcolm Atkinson, Amélie Levray and Rui Zhao, *DKB design*, in progress https://docs.google.com/document/d/1hCyoqeB8R0Ov5ZcBZVxyCOAiOST7QEP90n37PnghxGs/edit?usp=sharing

[24] Amélie Levray and Rui Zhao, *DKB specification*, in progress
https://docs.google.com/document/d/1bh2CzZJOUOYL_1nPJI8f5hcokHYxB1m9NHr9owWq_F0/edit?usp=sharing
[25] Larry Lannom, Dimitris Koureas, Alex R. Hardisty . *FAIR data and services in biodiversity science and geoscience*. Data Intelligence 2 (2020), 122–130. http://www.data-intelligence-journal.org/p/40/
[26] Iraklis Angelos Klampanos, Athanasios Davvetas,etc. *DARE: A Reflective Platform Designed to Enable Agile Data-Driven Research on the Cloud*, in BC2DC workshop proc. https://ieeexplore.ieee.org/document/9041753
[27] P. Groth, H. Cousijn, T. Clark & C. Goble. *FAIR data reuse – the path through data citation*. Data Intelligence 2(2020), 78–86.http://www.data-intelligence-journal.org/p/37/
[28] P. Wittenburg, *et al.*, *Digital objects as drivers towards convergence in data infrastructures*, Tech. rep., GO FAIR Office, Leiden (2018)
https://www.rd-alliance.org/sites/default/files/Digital_Objects_as_Drivers_towards_Convergence_in_Data.pdf
[29] A. Tsili, I. Klampanos, A. Spinuso, *D3.5-DARE-API-I*
http://project-dare.eu/wp-content/uploads/2019/03/D3.5-DARE-API-I_final_draft.pdf