Abstraction of user storage mechanisms for heterogeneous REANA scientific pipelines.

September 2021

AUTHOR(S):
María Camila Díaz Sánchez
IT-CDA-DR

SUPERVISOR(S):
Tibor Šimko
Audrius Mečionis

ABSTRACT

REANA is an open-source reusable research data analysis platform, that allows researchers to run their analyses in remote compute clouds. The analyses use containerised environments and rely on declarative computational workflow specifications. The workflows use remote workspaces to share input/output and temporary files between workflow jobs during workflow execution.

The goal of this project is to abstract the concept of the workspace in the REANA platform, in order to allow the usage of various storage backends at the same time .This will allow the REANA administrators the flexibility to deploy clusters with several authorised workspace locations, as well as allow users to choose the desired workspace location for each particular workflow run.

The present work allows the integration of any POSIX-based filesystem storage solution and paves the way towards using object-based file storage solutions in the future.

TABLE OF CONTENTS

1. INTRODUCTION

According to a 2016 survey made by Nature[1], in recent years there has been a reproducibility crisis in most science fields. For 52% there is a significant crisis on reproducibility. In Physics and Engineering field, over 50 % of surveyed researchers failed to reproduce their own experiments, and almost 70% failed to reproduce someone else´s.

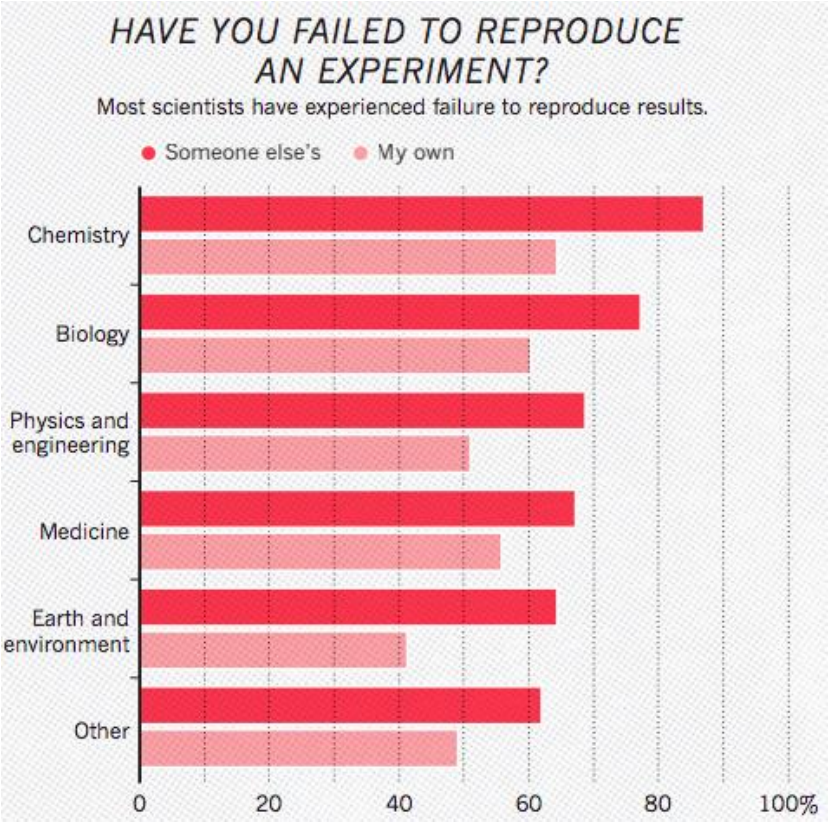

Figure 1: Reproducibility failures in science [1]

Under this survey, less than 31% of researchers think that incorrect results is the cause of the failure in the reproducibility. Some of the factors rated to contribute to irreproducible research, shown in Figure 2., include the unavailability of methods and code used, the poor experimental design and lack of the raw data used in the original lab.

In the case of High Energy Physics (HEP), the complexity and costs of the facilities used to perform the experiments make almost impossible to fully and independently replicate them. This makes it even more important to document all data taking procedures and ensure the reproducibility of data processing workflows. The reproducibility of computational experiments must then address 4 principal pillars: input data and parameters, analysis code, computational environment, analysis workflow.

To solve this issue, the IT-CDA-DR section at CERN developed the open-source platform REANA, a reusable and reproducible research data analysis platform that offers

systematic approach to generalize computational practices by allowing users to structure their analyses and run them in controlled containerized compute clouds.

REANA is composed of microservices, each one responsible of a defined part in the execution of the workflows or the connection between the local server and the remote cloud. For example, REANA-Workflow-Controller[2] is responsible for executing workflow operations such as workflow creation, spawning workflow engines, checking the status of running workflows, or returning the log files of finished workflows. The REANA-Job-Controller[3] component manages all the computational jobs of the workflows. The components of REANA share workspace as a common storage for the input and output files of each step. A predefined shared directory for every user of the system was the only available workspace location at the beginning of this project.

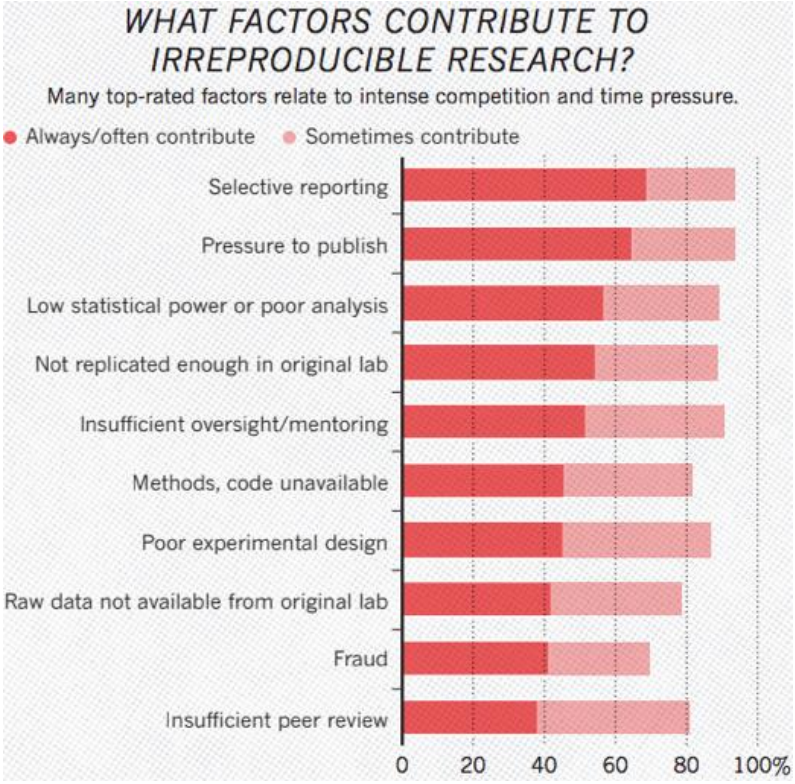

Figure 2: Factors contributing to irreproducible research [1]

The necessity of having a more flexible workspace definition, giving to the administrators the control of the available storage for the workflows and to the users the choice of where to run them, established this project. With this, we aimed to centralize the concept of workspace and make it a property of the workflows.

This new concept of the workspace allows to use alternative backend storage, such as different POSIX file systems, and makes the first step to make extension to different storages types, such as object bases storage.

2. USAGE AND REQUIREMENTS

a. REANA shared storage

As previously stated, reusable analyses consist of 4 components: input data, code, environment and dependencies, and workflow. The storage used by REANA for the first component was a shared storage folder that uses CephFS [4] as a default storage backend. This shared directory is a configurable value passed during the REANA deployment [5], having the folder `/var/reana` as the default value.

The workflows are created and a workspace inside the shared folder is created as `/users/{reana_user_id}/workflows/{workflow_id}` inside the shared directory. These workspaces are then used for all operations during workflow execution.

b. Workspace requirements

This fixed definition of workspaces did not allow researchers to use different or additional storage locations for their workflows. The solution to integrate this new workspaces needs to :

- Allow administrators to choose the available storage inside the cluster and to set a default storage different from the standard storage used by REANA.
- Make the workspace a property that can be chosen by the users while creating the workflow
- Validate the desired workspace against the allowed directories previously set by the administrator

3. IMPLEMENTATION

The first step to allow the usage of new workspaces is to abstract and centralize the workspace property present in the REANA´s cluster database.

a. Workspace abstraction from the database

REANA-Client provides a CLI with various commands to control the workflows. The commands used to manage individual workflows rely on the workspace, meaning they use the database and the storage associated, as can be seen in Figure 4, with the workflow to access the files used in the different steps of the workflows or to be retrieved to the users.

REANA-Database [6] is the component of REANA platform responsible to create the database persistence for REANA system, by creating the models and handing the utilities needed to manage this models. The Workflows table is part of models, used to store the characteristics and specifications of a workflow, such as the name, the owner user id, the

specification of the workflow, the status of the workflow, and relevant to this project, the workspace associated with the workflow.

During the creation of a workflow, the util's function `build_workspace_path()` built the workspace stored as previously stated, taking the user id and the workflow id. In order to fully centralize this workspace, the workflow must now store the full path associated. As an example, if the workspace is created inside the shared folder `/var/reana`:

```
/users/{user_id}/workflows/{workflow_id}/
```

⬇

```
/var/reana/users/{user_id}/workflows/{workflow_id}/
```

Figure 3: Change in the workspace store by REANA.

By doing this, all the information associated with the workspace is centrally stored in the Workspace model as an attribute. Inside the REANA cluster architecture (Figure 5), the entry point is REANA-Server [7] which manages and authenticates the request, passing them to REANA-Workflow-Controller. This component instantiate and manage the workflows, manipulating the workflow workspaces.

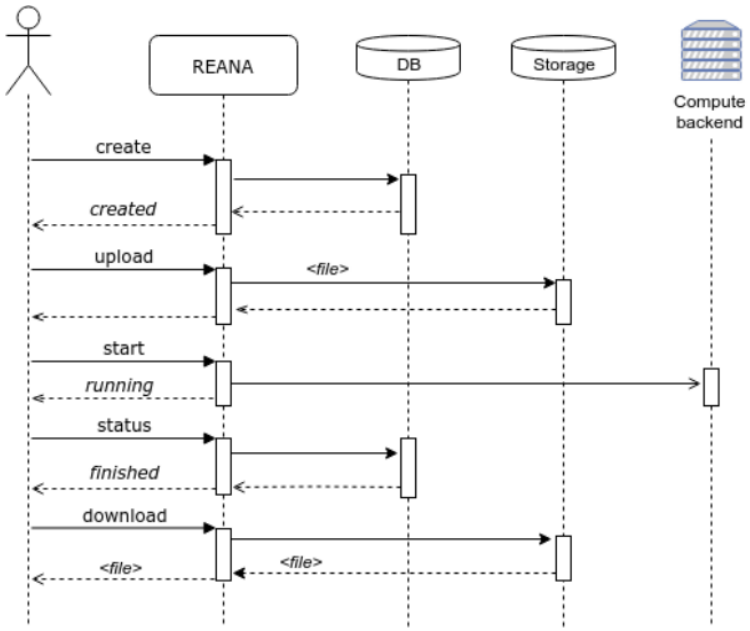

Figure 4: Sequence diagram of the backend usage for some client commands [8]

Following the centralization of the workspace in the database, all the functions inside REANA-Server and REANA-Workflow-Controller that manipulate or read the workspace from the database were refactored to directly take the full path stored in the database, instead of building it from the initial shared directory.

b. Workspace Selection by Administrators

All REANA components are built using Docker images [9] as jobs from one of several computing backends (Kubernetes, CERN HTCondor, CERN Slurm). The REANA cluster is orchestrated and deployed using a Helm chart [10]. In order to allow the administrators to choose the available storage a new value **.workspaces.paths** was added to the `values.yaml` file of the REANA Helm package, as a list of the workspaces.

The value of the workspaces is composed by a host path, as the path that will be mounted from the kubernetes nodes, and a mount path, as the path inside the pods where the host path is mounted, as a `hostPath:mountPath` string. The value **.workspaces.paths** has as a default value the shared volume `/var/reana`. All POSIX file systems are supported as possible workspaces, as can be an external storage or EOS at CERN.

The list of mounted paths are the allowed workspaces of the cluster. The first value listed is deemed as the default workspace used. Any workflow where the workspace is not specified, will have this default path as the root directory of its workspace. The paths are mounted into REANA-Server and REANA-Workflow-Controller while the cluster is being deployed.

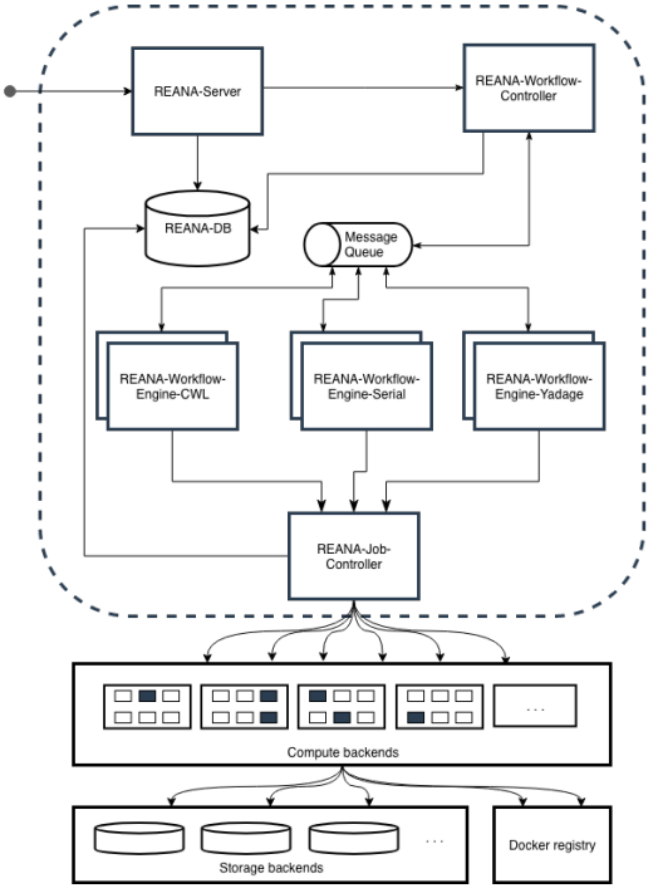

Figure 5: REANA cluster architecture and its components [8]

As shown in Figure 5., REANA-Workflow-Controller triggers the creation of jobs, used to run every step of the workflows. Therefore, the workflow controller the also mounts inside the jobs while being created.

c. Workspace exposure to REANA users

Once the workspaces are available inside the cluster, the users can choose the desired workspace location for their workflows. The workspace location can be specified as a new top section on the workflow specification file `reana.yaml`. An example of a specification file of the REANA-Demo-Helloworld [11] with a declared workspace can be seen in Figure 6.

```
version: 0.3.0
inputs:
  files:
    - code/helloworld.py
    - data/names.txt

  parameters:
    helloworld: code/helloworld.py
    inputfile: data/names.txt
    outputfile: results/greetings.txt
    sleeptime: 0

workflow:
  type:serial
  specification:
    steps:
      - environments: 'python:2.7-slim'
        commands:
          -python "${helloworld}"
              --inputfile "${inputfile}"
              --outputfile "${outputfile}"
              --sleeptime ${sleeptime}

outputs:
  files:
   -results/greetings.txt

workspace:
  root_path: /myexternaldrive/workspace
```

Figure 6: Example of a `reana.yaml` specification file with the section workspace.

In case the workspace root path is not declared inside the specification file, the workspace given to the workflow is built with the user id and the workflow id, as seen in Figure 3 , replacing the shared directory with the default workspace defined by the administrator.

d. Workspace validation

The last step needed to fully integrate the workspaces to REANA usability is to integrate the validation of a workspace solicited by the user against the workspaces authorized by the administrator.

The validation of a specification file is made inside the `reana-client` package, either using the `reana-client validation` command or as the first step while creating a new workflow. As the validation is performed outside the cluster, a new API endpoint was added to REANA-Server to extract the workspaces available of the desired REANA cluster. The example of a correct and incorrect validation of the workspace in the specification file can be seen in Figure 7.

```
$ reana-client validate -f reana.yaml
==> Verifying REANA specification file... reana.yaml
  -> SUCCESS: Valid REANA specification file.
==> Verifying REANA specification parameters...
  -> SUCCESS: REANA specification parameters appear valid.
==> Verifying workflow parameters and commands...
  -> SUCCESS: Workflow parameters and commands appear valid.
==> Verifying dangerous workflow operations...
  -> SUCCESS: Workflow operations appear valid.
==> Verifying workspace in REANA specification file...
  -> SUCCESS: Workflow workspace appear valid.

$ reana-client validate -f reana.yaml
==> Verifying REANA specification file... reana.yaml
  -> SUCCESS: Valid REANA specification file.
==> Verifying REANA specification parameters...
  -> SUCCESS: REANA specification parameters appear valid.
==> Verifying workflow parameters and commands...
  -> SUCCESS: Workflow parameters and commands appear valid.
==> Verifying dangerous workflow operations...
  -> SUCCESS: Workflow operations appear valid.
==> Verifying workspace in REANA specification file...
==> ERROR: Desired workspace "/var/rean" not valid.
Please run reana-client info to see the list of allowed prefix
values.
```

Figure 7: Example of the `reana-client validation` for workspaces.

The users may want to learn about the list of available workspace locations that were allowed by the administrators. To allow this, a new command was added to `reana-client` as `info`. The new command lists will list all available workspace locations and the default workspace of the cluster. An example of the workspaces command result can be seen in the Figure 8.

```
$ reana-client info
Default workspace: /var/reana
List of available workspaces: /var/reana, /home/myworkflows, /eos
```

Figure 8: Example of the `reana-client info` command.

4. CONCLUSIONS

This report summarizes the results of the full abstraction of the workspace location concept inside the REANA platform. The REANA codebase was refactored to be able to use the centralized absolute workflow workspace stored in the database.

The REANA cluster administrators can configure a list of allowed workspace locations while deploying the cluster as well as and choose the default workspace storage option. The platform currently supports any POSIX compliant file system as workspace storage.

The choice of workspace location is exposed to users who can select their desired workspaces by means of the standard workflow specification file. The REANA command-line client has been modified to validate the desired values as part of the workflow submission and to give the available workspaces information.

5. FUTURE WORK

The possible future work to improve the current implementation can address the following points:

- Extend integration with the EOS system, to allow direct access to personal EOS spaces for users at CERN.
- Extend storage types supported as storage beyond POSIX layers, to object-based files systems, such as AWS/S3.

REFERENCES

[1] Baker, M. 1,500 scientists lift the lid on reproducibility. *Nature* **533,** 452–454 (2016). https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970

[2] Reproducible and reusable research data analysis, REANA-Workflow-Controller. https://github.com/reanahub/reana-workflow-controller.

[3] Reproducible and reusable research data analysis, REANA-Job-Controller. https://github.com/reanahub/reana-job-controller.

[4] Ceph File System. https://docs.ceph.com/en/pacific/cephfs/

[5] Reproducible and reusable research data analysis, Deploy values. https://github.com/reanahub/reana/blob/master/helm/reana/README.md

[6] Reproducible and reusable research data analysis, REANA-Database. https://github.com/reanahub/reana-db

[7] Reproducible and reusable research data analysis, REANA-Server. https://github.com/reanahub/reana-server

[8] Šimko, T.; Heinrich, L; Hirvonsalo, H; Kousidis, D; Rodríguez, D . REANA: A System for Reusable Research Data Analyses EPJ Web Conf. 214 06034 (2019) http://cdsweb.cern.ch/record/2652340/files/CERN-IT-2018-003.pdf

[9] Docker Platform. https://docs.docker.com/get-started/overview/

[10] Helm Charts. https://helm.sh/

[11] Reproducible and reusable research data analysis, REANA-Demo-Helloworld. https://github.com/reanahub/reana-demo-helloworld