

Facilitating future open data reuse via continuous integration of actionable data analysis examples

Marco Donadoni*, Audrius Mečionis*, Giuseppe Steduto*[†], Tibor Šimko*[‡]

* CERN, Geneva, Switzerland

[†] Polytechnic University of Milan, Italy

[‡] Corresponding author. Email: tibor.simko@cern.ch

Abstract—We describe a use case of facilitating open data reuse in experimental particle physics. The CERN Open Data portal disseminates over three petabytes of data published by LHC collaborations. The data is accompanied by preserved computational analysis environments and various analysis examples documenting and illustrating data access and reuse patterns. We have developed a “continuous integration” system allowing to run actionable code examples in their original computing environments using the REANA reproducible analysis workflows in order to ensure the validity of data access patterns across time. We conclude by advocating the importance of early preservation of actionable software examples alongside datasets as a key action to facilitate future data reuse.

Index Terms—data preservation, data reuse, reproducible research, computational workflows, open data, open science

I. INTRODUCTION

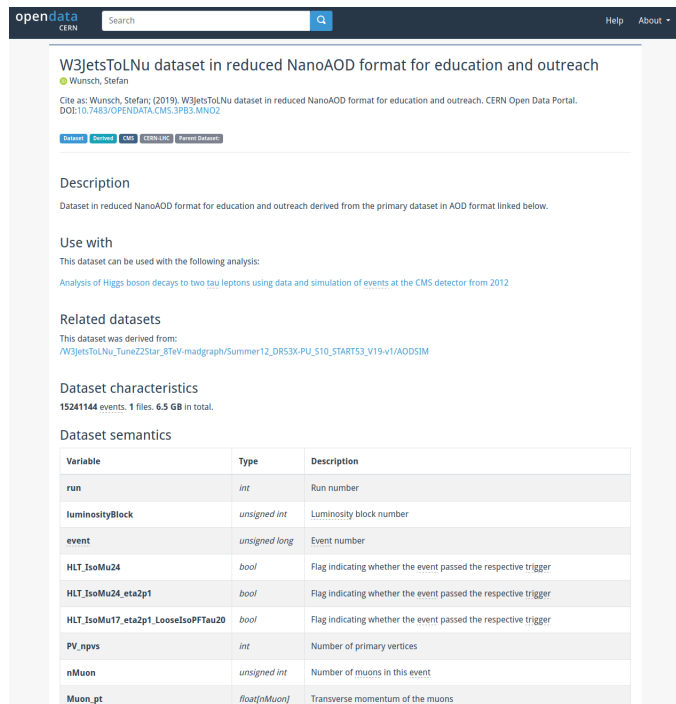
The long-term preservation of research data in particle physics poses numerous challenges. Besides the technical complexity inherent in managing petabyte-scale data volumes with datasets consisting of tens of thousands of files of gigabyte size, it is essential to capture accurate semantic description of data, their provenance, the accompanying meta-information, as well as the typical data usage patterns.

In this paper, we describe how preserving small actionable examples demonstrating typical data analysis procedures can help to ensure the completeness of preserved information. Taking advantage of the container technology, the data reuse examples are run in a continuous-integration manner to ensure the correctness of preserved data and knowledge in view of facilitating future data reuse.

II. CERN OPEN DATA

The CERN Open Data portal [1] disseminates over three petabytes of data from particle physics. The data is accompanied by virtual machines, software containers and examples of typical data analysis patterns. The experimental collaborations release data in close collaboration with data curators by means of semi-automated curation workflows [2].

Figure 1 shows one reduced dataset for education and outreach published by the CMS experiment. The dataset was derived from the bigger CMS simulated Monte Carlo dataset in order to reduce data complexity (from the original dataset consisting of 1715 files of 5.4 TiB size to the published reduced dataset consisting of 1 file of 6.1 GiB size; note that the original dataset is also available). The dataset variables



The screenshot shows the CERN Open Data portal interface. At the top, there is a search bar and navigation links. The main content area displays the following information:

- Title:** W3jetsToLNu dataset in reduced NanoAOD format for education and outreach
- Author:** Wumsch, Stefan
- Cite as:** Wumsch, Stefan; (2019). W3jetsToLNu dataset in reduced NanoAOD format for education and outreach. CERN Open Data Portal. DOI:10.7483/OPENDATA.CMS.3FES.MNCG
- Description:** Dataset in reduced NanoAOD format for education and outreach derived from the primary dataset in AOD format linked below.
- Use with:** This dataset can be used with the following analysis: Analysis of Higgs boson decays to two tau leptons using data and simulation of events at the CMS detector from 2012
- Related datasets:** This dataset was derived from: /W3jetsToLNu_TuneZ2Star_8TeV-madgraph/Summer12_DR53X-PU_S10_START53_V19-v1/AODSIM
- Dataset characteristics:** 1524144 events. 1 files. 6.5 GB in total.
- Dataset semantics:** A table with 3 columns: Variable, Type, and Description.

Variable	Type	Description
run	int	Run number
luminosityBlock	unsigned int	Luminosity block number
event	unsigned long	Event number
HLT_IsoMu24	bool	Flag indicating whether the event passed the respective trigger
HLT_IsoMu24_eta2p1	bool	Flag indicating whether the event passed the respective trigger
HLT_IsoMu17_eta2p1_LooseIsoPFTau20	bool	Flag indicating whether the event passed the respective trigger
PV_npv	int	Number of primary vertices
nMuon	unsigned int	Number of muons in this event
Muon_pt	float[nMuon]	Transverse momentum of the muons

Fig. 1. An example of a CMS dataset available on the CERN Open Data portal digital repository [3]. The data format is custom NanoAOD suitable for educational physics analyses. The screenshot shows the data semantics section describing the variables.

are documented in detail in order to facilitate understanding how to work with the dataset. However, the best way to ensure that the data are “actionable” is via a directly runnable analysis example showing how to read the physics variables from the dataset and use them in an example data analysis.

Figure 2 shows such an actionable analysis example studying Higgs to tau lepton decays. The example uses the ROOT analysis framework [4] and studies nine reduced datasets similar to the one presented in Figure 1. The analysis consists of several steps running C++ and Python code and produces several physics results. The analysis computational environment, the source code and the detailed procedures are published so that any open data digital repository user can study and reproduce this data reuse example.

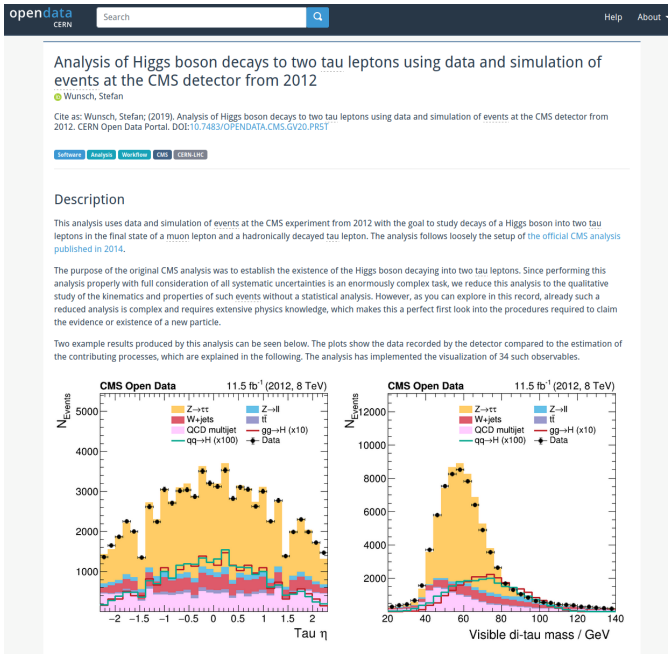


Fig. 2. An example of physics analysis using dataset from Figure 1. The analysis uses ROOT physics analysis framework with C++ and Python code and illustrates the reading of the dataset for the study of Higgs boson decays to two tau leptons [5].

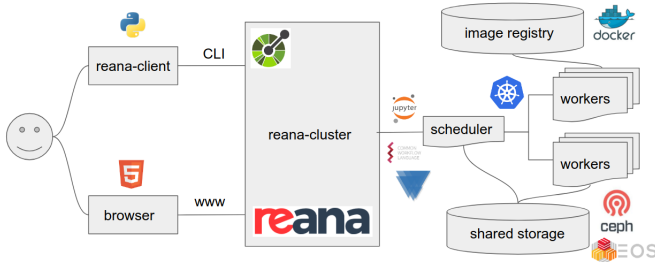


Fig. 3. The architecture of the REANA reproducible analysis platform allowing users to submit containerised computational workflows using command-line and web interfaces [7]. The REANA platform instantiates the submitted analysis runs on the compute cloud using container technologies. The workflow jobs use a shared workspace storage to exchange computation results amongst multiple cluster nodes.

III. REANA REPRODUCIBLE ANALYSES

REANA [6] is a reusable and reproducible data analysis platform that offers tools to particle physicists to structure their input data, analysis code, compute environments and computational workflow steps so that data analyses can be instantiated and run on remote containerised compute clouds. The platform uses container technologies (Docker, Singularity) and runs user jobs on supported compute backends (Kubernetes, HTCondor, Slurm) using several supported declarative workflow description languages (CWL, Snakemake, Yadaage) [6].

Figure 3 shows the overall architecture of REANA platform. A user can use command-line or web interfaces to submit containerised analysis workflows which are then instantiated and ran on supported computed backends using containerised

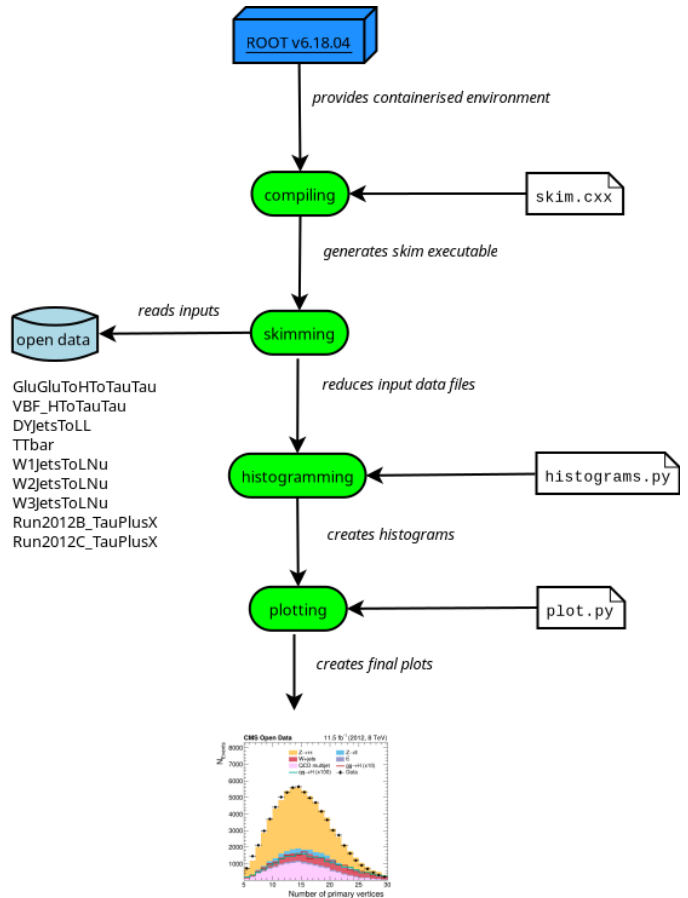


Fig. 4. The computational workflow associated to the $H \rightarrow \tau\tau$ data use example presented in Figure 2. The workflow consists of four simple sequential steps that run inside the ROOT analysis environment. Firstly, the dataset skimming code is compiled for the targeted compute backend. Secondly, the skimming job is run that reads nine different datasets from the digital repository. Thirdly, histogramming is run on the skimmed data. Finally, the histograms are plotted.

images [6].

Figure 4 shows the computational workflow associated to the $H \rightarrow \tau\tau$ data analysis usage example that was presented in Figure 2. The data is read from the CERN Open Data portal digital repository in the skimming step and the reduced data is subsequently analysed and physics histograms plotted. Note that the analysis uses several datasets. The overall dataset reuse example is sequential in this case, but the computational workflows can be parallelised and can consist of many hundreds of analysis steps, as required by the physics content of the data reuse example.

Figure 5 presents results of one such computational workflow run on the REANA reusable analysis platform. The user sees detailed logs of the workflow, the produced files, and can use the web interface to visualise final plots.

The automation of various open data reuse examples such as the $H \rightarrow \tau\tau$ example workflow from Figure 4 and their periodic execution helps to ensure that the released data is still usable as a function of time just as it was when the data and the reuse examples were first published.

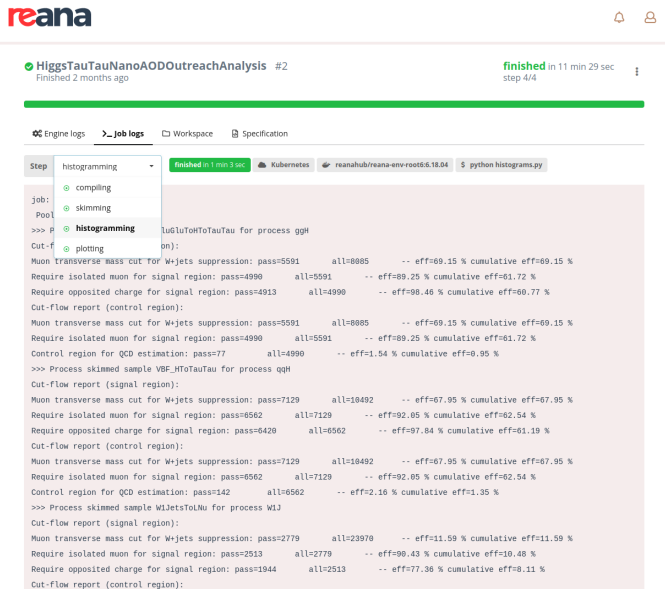


Fig. 5. Results of one computational run of the $H \rightarrow \tau\tau$ example workflow from Figure 4 on the REANA platform. The screenshot shows the logs of the histogramming stage. A parallel tab allows users to peruse the workspace and visualise workflow outputs such as produced physics plots.

IV. “CONTINUOUS REUSE”

We have developed a “continuous integration” system allowing to run the open data usage examples by means of containerised computational workflows in a periodic manner in a fully automated way.

Given the description of the input data, the analysis example code, the necessary computing environments as well as the precise computational workflow steps to analyse the data, the data reuse examples are instantiated on the REANA reproducible analysis platform. The periodic “continuous reuse” allows to ensure the accessibility of data, verify the accuracy and completeness of preserved information as well as the correctness of the recipes for data reuse procedures.

The continuous periodical reuse helps to discover problems with not only access to data, but also access to any remote resources such as condition database fetched from a remote site, troubles induced by changing versions of software protocols, etc. This provides an early-warning system for a plethora of possible data reuse blockers.

A. Architecture overview

The overall architecture of the “continuous reuse” platform is shown in Figure 6. The two core components of the system are the digital repository that is hosting the data (i.e. the CERN Open Data portal) and the containerised computational workflow running system (i.e. REANA). The “continuous reuse” service joins and links these two core components and ensures periodical execution and testing of the reuse examples.

The core of the “continuous reuse” service consists of a central database listing the reuse examples that are to be periodically run. The “reuse launcher” component takes care

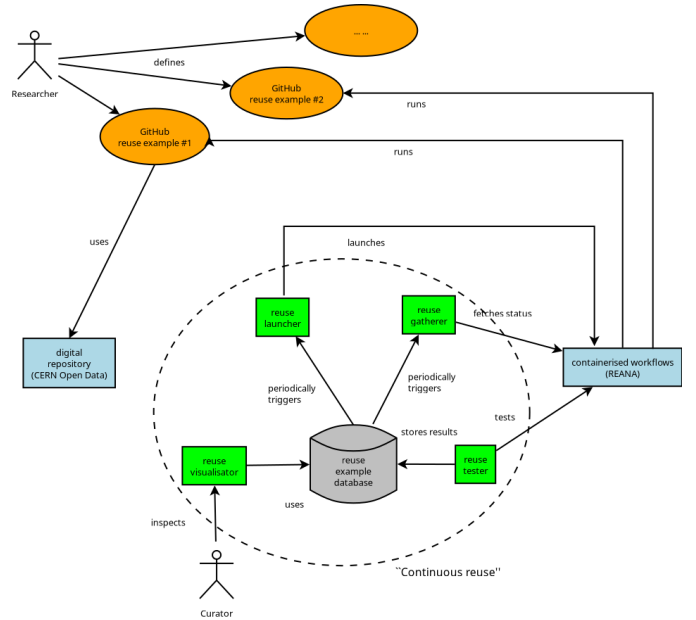


Fig. 6. Architecture of the “continuous reuse” platform. The platform allows to specify the reuse examples using the data together with tests to check whether the reuse runs were successful. The launcher, gatherer and tester are periodically triggered by the platform with desired frequency. The results are gathered in a database and visualised for the data curators allowing to assess whether the reuse examples keep working normally.

of launching these examples on the REANA reproducible analysis platform with the desired periodicity (daily, weekly, monthly). The “reuse gatherer” component then polls the REANA system to enquire about the status of submitted runs and collects information about their statuses (finished, failed, *etc*) and logs them in the central database. Finally, the “reuse tester” component is triggered to verify the output of finished workflow runs against a desired list of behavioural tests comparing analysis workspace files, job logs, and job duration times.

The gathered data is then exposed via the “reuse visualiser” component that allows the end users such as data curators to have an easy-to-peruse web dashboard showing the history of reuse runs as a function of time, allowing to either confirm the validity of the data reuse example or detecting any broken reuse example for further inspection and fixing.

B. Defining reuse examples

The reuse examples are defined in the “continuous reuse” database. The entries are added into the database by means of YAML snippets that describe the reuse example. Figure 7 shows one such snippet for the $H \rightarrow \tau\tau$ reuse example described above in Figures 1, 2, 4. The reuse example name is specified together with the location of the source code files (i.e. on GitHub), the location of the desired behavioural tests (see section IV-C), as well as the desired testing frequency (“daily”). The optional category field (“cms”) allows for grouping several reuse examples for a joint dashboard presentation to data curators (see section IV-D).

```

- name: cms-htautau-nanoaod
  category: cms
  frequency: daily
  source:
    url: "https://github.com/cms-opendata-analyses/HiggsTauTauNanoAODOutreachAnalysis"
  testfiles:
    - cms-htautau-nanoaod.feature

```

Fig. 7. An example of a YAML file snippet defining the testable continuous reuse example for the $H \rightarrow \tau\tau$ analysis. The location of the source code of the reuse workflow is specified, together with the desired running frequency and the desired tests to be performed on the finished run. An example of a test file content is listed in Figure 8.

```

Feature: cms-htautau-nanoaod

  Scenario: Workspace content
    When the workflow is finished
    Then the workspace should contain "njets.png"
    And the workspace should contain "phi_1.png"
    And the workspace should contain "phi_2.png"
    And the workspace should contain "pt_1.png"
    And the workspace should contain "jpt_2.png"
    And the workspace should contain "jdetta.png"

  Scenario: Workspace size
    When the workflow is finished
    Then the workspace size should be less than 75 MiB

  Scenario: Log content
    When the workflow is finished
    Then the job logs of the step "skimming" should contain "Event has good muons: pass=36921"
    And the job logs of the step "skimming" should contain "Event has good taus: pass=38041"
    And the job logs of the step "histogramming" should contain
      "Muon transverse mass cut for W+jets suppression: pass=5063"
    And the job logs of the step "plotting" should contain "pdf file m_1.pdf has been created"
    And the job logs of the step "plotting" should contain "pdf file m_2.pdf has been created"

  Scenario: Run duration
    When the workflow is finished
    Then the workflow run duration should be less than 25 minutes
    And the duration of the step "compiling" should be less than 1 minutes
    And the duration of the step "skimming" should be less than 20 minutes
    And the duration of the step "histogramming" should be less than 3 minutes
    And the duration of the step "plotting" should be less than 3 minutes

```

Fig. 8. Example of a Gherkin file defining expected outcomes of one reuse run of the $H \rightarrow \tau\tau$ analysis. The behavioural tests are using predefined “business-friendly” language that allows to specify the desiderata such as the presence of certain output files in the workspace or the presence of certain phrases in the job logs. The duration of various steps can also be tested to alert about any possible performance deterioration.

C. Defining behavioural tests

The behavioural tests are using the Gherkin feature files [8] that allow data curators to express the desired tests using a natural language syntax. This offers several advantages over other formal approaches used for writing system tests by being close to the problem-domain language [9].

The Gherkin feature file lists several testing scenarios that group together certain individual tests and include a trigger condition specifying when the individual tests are to be run:

```

Feature: <reuse example>
  Scenario: <scenario name>
    When <trigger condition>
    Then <test to be run>
    And <another test to be run>
    And <another test to be run>
  Scenario: <another scenario>
    When ...

```

We have developed a syntax parser allowing data curators to express the individual tests in a workflow-oriented natural language. The selection of available tests include testing of the absence or presence of certain files in the workspace (“workspace should contain”), testing of the expected workspace and file sizes (“size should be”), testing the absence or presence of certain phrases in the job log messages (“job logs of the step X should contain”), as well as testing the temporal duration of individual steps in the workflow (“duration of the step X should be”).

Figure 8 shows the concrete example of how the desired tests were defined for the $H \rightarrow \tau\tau$ reuse example. The developed language is problem-domain oriented and does not require the knowledge of any programming language. It is usable for data curators that are not necessarily acquainted with the inner workings of the “continuous reuse” system.

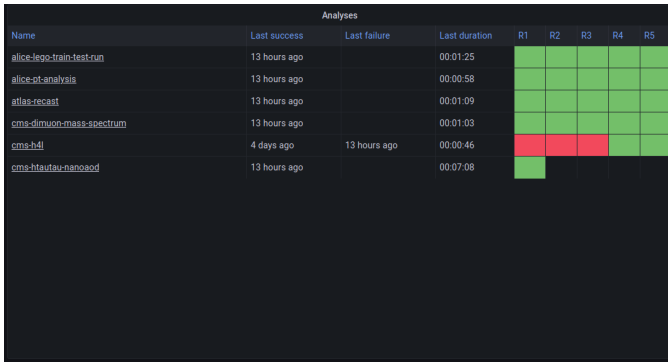


Fig. 9. The dashboard home page showing the overall status of reuse examples defined by the user. The dashboard allows to quickly check the last success and the last failure time stamps as well as to inspect the results of last five runs.

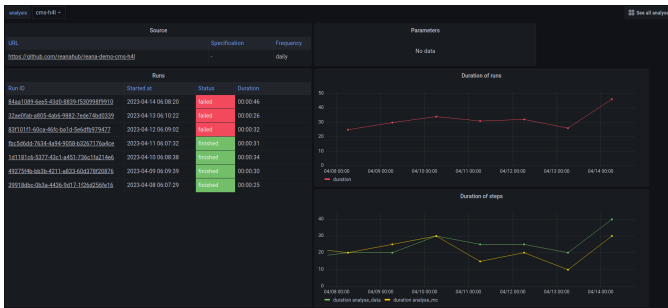


Fig. 10. The detailed page of the dashboard for one particular reuse example. Besides showing the run history, the page allows to inspect notably the different workflow steps and their duration times for several past runs. This is especially important for complex reuse examples that may contain many dozens of computational steps necessary to illustrate complex data reuse scenarios.

D. User dashboard

The collected data about the periodic runs of reuse examples and about successful and unsuccessful tests are stored in the database. A web dashboard was developed using Grafana allowing to visualise the collected data. Figure 9 shows the main dashboard page with a view on a history of various reuse examples and their success/failure statistics. The information about last success and last failure is printed together with the duration of the run and the summary of last five runs. The reuse examples can be organised in categories in order to expose joint statistics for certain groups of data reuse examples (e.g. by data curator teams).

Another dashboard view offers detailed information about one particular data reuse example as illustrated in Figure 10. In addition to presenting the overall success/failure statistics over several past runs, the workflow details such as the duration of each step of the workflow can be seen. This allows to detect any possible computational bottlenecks related to various parts of the data reuse pipelines.

E. First results

The developed “continuous reuse” system is being used to test the various CERN Open Data reuse examples in view of

detecting early any problems with access to the preserved data in the digital repository or with the applicability of published data usage examples. This allows to detect problems not only with data storage per se, but also the deprecation of data access protocols, the impact of changed software versions or changed compute environments.

The best way to understand research data is by means of small actionable examples, and the “continuous reuse” philosophy is helping to make sure that the examples keep running as a function of time regardless of any changes in data hosting or computing solutions.

V. CONCLUSIONS

The focus of the present work was to ensure mid-term data reuse by taking advantage of container technologies. We argue that preparing actionable data reuse examples adds considerable value to the preserved data and that the “continuous reuse” by automated workflows ensures both the correctness and the completeness of the preserved information. This process also allows to better understand and document all the knowledge required to be able to reuse the data, which may prove valuable in the very long-term future even though the current container technologies themselves may become technologically obsolete in longer-term horizons.

REFERENCES

- [1] CERN Open Data portal, <https://opendata.cern.ch/>
- [2] T. Šimko *et al*, “Open data provenance and reproducibility: a case study from publishing CMS open data”, EPJ Web Conf. **245** (2020) 08014, <https://doi.org/10.1051/epjconf/202024508014>
- [3] S. Wunsch, “W3JetsToLNu dataset in reduced NanoAOD format for education and outreach”, CERN Open Data portal (2019), DOI <https://doi.org/10.7483/OPENDATA.CMS.3PB3.MN02>
- [4] R. Brun, F. Rademakers, “ROOT — An object oriented data analysis framework”, Nucl. Instr. Methods in Phys. Research A **389** (1997), 81–86
- [5] S. Wunsch, “Analysis of Higgs boson decays to two tau leptons using data and simulation of events at the CMS detector from 2012”, CERN Open Data portal (2019), DOI <https://doi.org/10.7483/OPENDATA.CMS.GV20.PR5>
- [6] REANA reusable analysis platform, <https://www.reana.io/>
- [7] T. Šimko *et al*, “Scalable declarative HEP analysis workflows for containerised compute clouds”, Front. Big Data **4** (2021), <https://doi.org/10.3389/fdata.2021.661501>
- [8] D. North, “The evolution of behavior-driven development”, Better Software 3 (2006), <https://dannorth.net/introducing-bdd/>
- [9] R. K. Lenka, S. Kumar and S. Mangain, “Behavior Driven Development: Tools and Challenges”, 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2018, pp. 1032-1037, DOI <https://doi.org/10.1109/ICACCCN.2018.8748595>