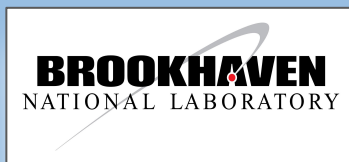


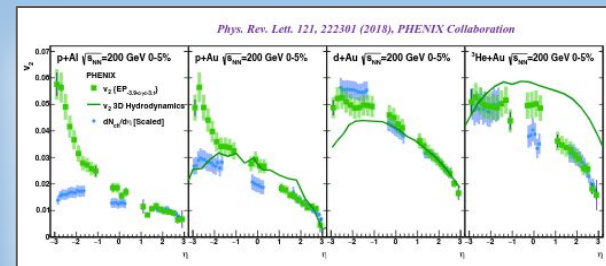
# Data & Analysis Preservation: current work items

Maxim Potekhin

*Nuclear and Particle Physics Software Group*



***PHENIX DAP Meeting***  
03/25/2021



# Overview

- REANA
- Docker
- Open Data
- HEPData
- Zenodo
- Website
- Analysis notes
- *Maxim on leave through April 11<sup>th</sup> (propose next meeting on April 22<sup>nd</sup>)*

# REANA - a quick recap



- REANA: captures the workflow, the software, the data
  - Hence reproducibility
  - If used correctly, helps present a clear description of the computational process
- Simple, intuitive description of linear workflows in YAML
  - Can be parametrized, self-documenting
- A more complex syntax is available to arbitrary DAGs
- A very basic Web UI provided by the server
- Interaction with the system is mainly via a CLI client, which can be installed on any machine where the Python environment is available
- In addition to the CLI utility there is a full-fledged Python package which allows for potentially complex applications, scripting etc

# REANA - software provisioning

- Software can be provisioned in different ways (source, images, CVMFS) depending on the needs of the calculation
  - For example, a specific version of ROOT or a Python environment can be obtained by requesting the correct image - from your chosen registry
  - Code can also be built as needed (perhaps not optimal but possible)
- All these options are available in any step of the workflow
  - e.g. the software can be provisioned differently in each step
- If the code has Docker image components, images are transparently pulled from Docker Hub or other comparable hosting service
- There is no direct mount of AFS on the worker nodes so everything needs to be staged in (CVMFS is still under discussion because of its use in EIC)
- However content from AFS can be *staged from the user's machine* if needed

# REANA - the process



- Execution takes place in a sandbox
  - A process running on one of the worker nodes in the cluster
  - Scripts, macros and other software can be staged-in (provisioning)
- Status can be checked in the Web UI or with the CLI client, or with a script utilizing the Python package
- Input data is typically staged-in to the sandbox before the execution starts
  - But data handling can also be a part of the payload job
  - CVMFS
  - XRootD?
  - This will be illustrated in an example of one of the subsequent slides
- Output is staged-out upon completion
  - Once again, possible with a variety of methods

# REANA - progress



- Tested staging AFS folders (need to be careful)
- Created an official PHENIX repository for REANA workflows
  - <https://github.com/PhenixCollaboration/reana>
- EMCAL single-cluster example (prepared for the Open Data) *ported to REANA*
  - Committed to GitHub
  - Simple and self-contained (includes the Ntuples)
  - Suitable for the PHENIX School, can be used as a template for more complex cases
- Picked the ROOT6 version of the macro for simplicity (see the next slide)
- *Anything that is based on pure ROOT macros can be committed to REANA trivially*
- Ultimately usefulness of this exercise will depend on containerization since interesting stuff is done with the PHENIX analysis libraries
  - See comments about Docker later in this presentation

# REANA - EMCAL single cluster analysis

version: 0.0.1

inputs:

files:

- ./single\_cluster.C
- data/ERTntup\_gnt.root
- data/MBntup\_gnt.root

workflow:

type: serial

specification:

steps:

- environment: 'rootproject/root'

commands:

- mkdir -p results
- root -b -q 'single\_cluster.C("position")' 'single\_cluster.C("ecore")' 'single\_cluster.C("chisq")'

outputs:

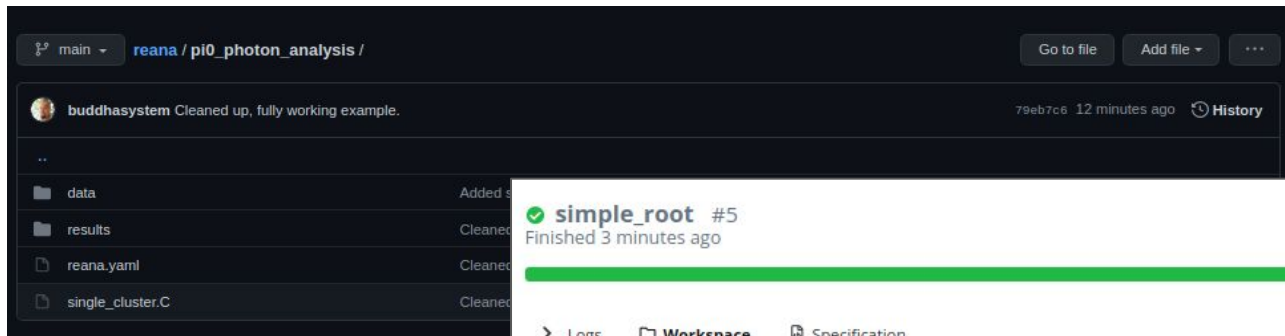
files:

- results/test.txt



# REANA - GitHub repo, test runs

[https://github.com/PhenixCollaboration/reana/tree/main/pi0\\_photon\\_analysis](https://github.com/PhenixCollaboration/reana/tree/main/pi0_photon_analysis)



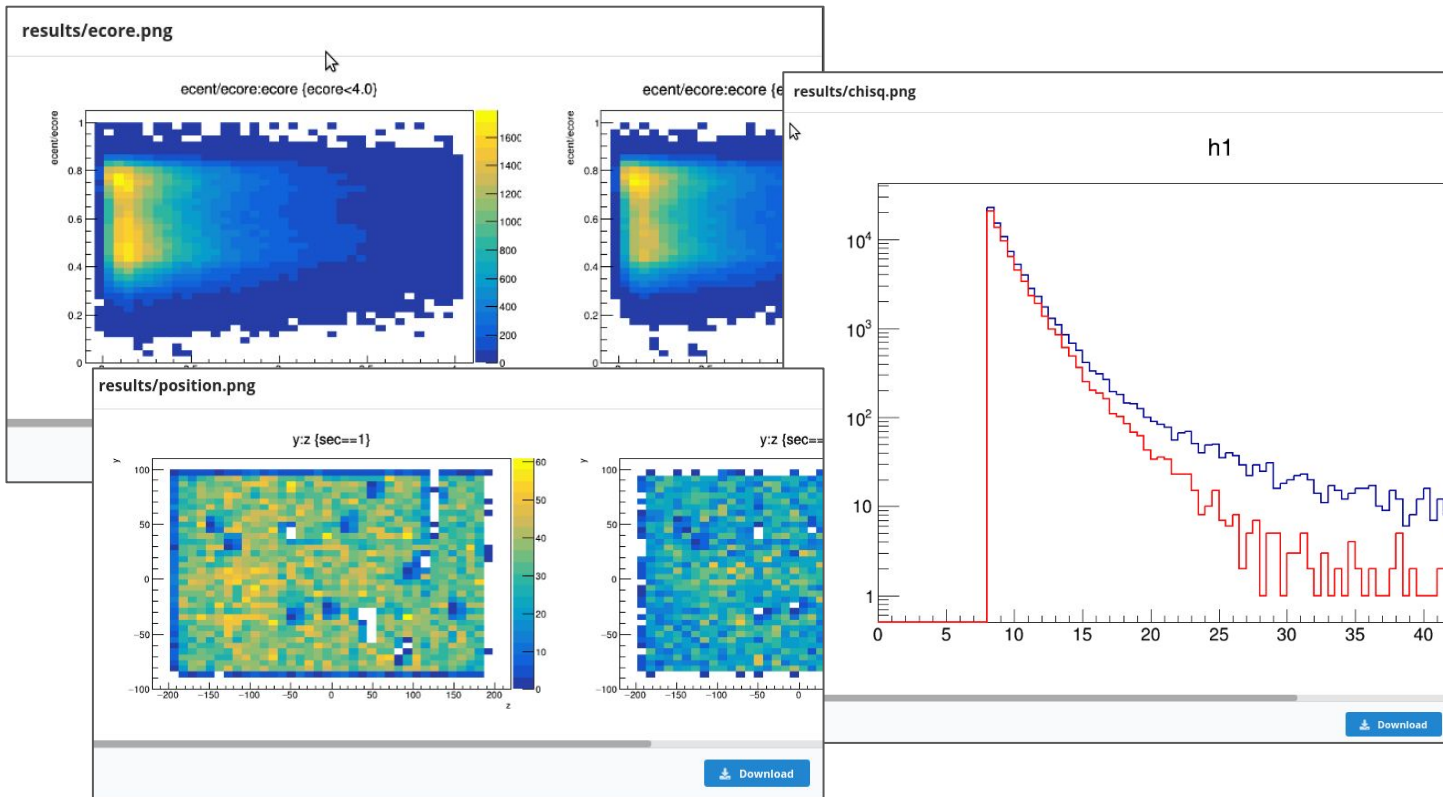
✓ **simple\_root** #5 finished in 36 seconds  
step 2/2  
Finished 3 minutes ago

> Logs   **Workspace**   Specification

Name	Modified	Size
single_cluster.C	2021-03-22T21:01:02	2431
results/chisq.png	2021-03-22T21:01:41	14301
results/ecore.png	2021-03-22T21:01:41	23544
results/position.png	2021-03-22T21:01:40	19529
data/ERTntup_gnt.root	2021-03-22T21:01:04	4286641
data/MBntup_gnt.root	2021-03-22T21:01:12	15122023



# REANA - results of the single cluster job



# Docker: current status

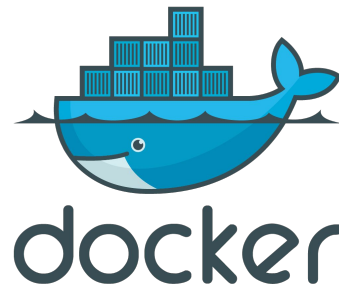


- We built a functional 64-bit ROOT5/SL7 image created from source
- Hosting on Docker Hub works fine, adding any 64-bit library should be easy
- It works with C++ macros, just not the PHENIX analysis libraries at this point
- Caveat: the PHENIX software stack is built for 32-bit for legacy reasons
- Both STAR and PHENIX build their software for 32-bit
  - Both solutions are tied to the respective experiment's setup

# Docker: the 32-bit problem

- Possible because of “multilib” feature of SL, effectively installation a parallel system of core libraries (i386 in addition to x8664) and a fairly complex setup for building ROOT via cross-compilation
  - ROOT5 available on the rcas nodes is 32-bit
- No portable procedure exists for the PHENIX stack (i.e. on someone’s workstation)
- SDCC does BNL-internal image builds but images are considered private (next slides)
- In summary, there are a lots of libraries and dependencies

# Docker: the outlook



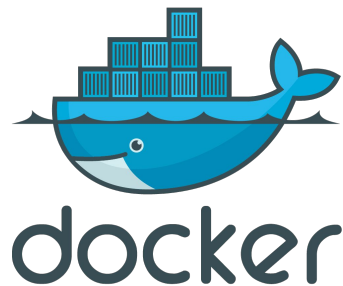
- Having a containerized PHENIX environment is a worthy goal
  - both for general DAP and REANA activities
- ***Copying libraries to a Docker image*** (as discussed before) is more involved than anticipated due to the 32-bit dependencies explained above
  - Brings in system dependencies i.e. i386 libraries that need to be identified and collected
  - *An experimental setup has been created with all the binaries managed in one place*
  - *Creating an image is the next step*
- Building this Docker image will be the focus of the DAP work for a while
- As the last option, we can try using private PHENIX/SL7 images provided by SDCC which we are not allowed to widely share for security reasons
- The issue of DB access - this only works within the BNL perimeter
  - Which analyses would be impacted if no access would be possible?

# Docker - testing “phpythia” in a self-contained env

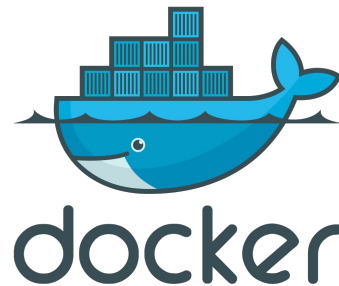
```
97 gamma      1      22      78      0.079      -0.000      -0.189      0.205      0.000
98 gamma      1      22      81      -0.030      -0.051      -0.131      0.144      0.000
99 gamma      1      22      81      -0.057      0.030      -0.6      0.144      0.000
100 pi+       1      211     84      -0.370      0.185      -19.1     0.185      0.000
101 pi-       1      211     84      -0.306      0.088      -5.5      0.185      0.000
102 pi+       1      211     93      -0.203      -0.043      -0.6      0.185      0.000
103 pi-       1      211     93      0.194      0.038      -0.1      0.185      0.000
=====
sum: 2.00      -0.000      0.000      -0.000
***** PYSTAT: Statistics on Number of Events and Cross Sections *****
=====
I Subprocess I Number of points I
I-----I-----I
I N:o Type I Generated I
I-----I-----I
I 0 All included subprocesses I 1000 I 11392 I 2.463D+01 I
I 11 f + f' -> f + f' (QCD) I 65 I 776 I 1.648D+00 I
I 12 f + fbar -> f' + fbar' I 0 I 5 I 1.781D-02 I
I 13 f + fbar -> g + g I 2 I 10 I 2.521D-02 I
I 28 f + g -> f + g I 432 I 5577 I 1.046D+01 I
I 53 g + g -> f + fbar I 22 I 89 I 4.204D-01 I
I 68 g + g -> g + g I 479 I 4935 I 1.206D+01 I
=====
***** Total number of errors, excluding junctions = 0 *****
***** Total number of errors, including junctions = 0 *****
***** Total number of warnings = 0 *****
***** Fraction of events that fail fragmentation cuts = 0.00000 *****

maxim@ferocity:~/phpythia$ ls -ltr
total 16776
-rwxr-xr-x 1 maxim maxim 3834564 Mar 17 20:24 libfun4all.so
-rwxr-xr-x 1 maxim maxim 292560 Mar 17 20:24 libPHPythiaEventGen.so
-rwxr-xr-x 1 maxim maxim 1494308 Mar 17 20:24 libPHPythia.so
-rwxr-xr-x 1 maxim maxim 2575604 Mar 17 20:24 libPHPyTrigger.so
-rwxr-xr-x 1 maxim maxim 826576 Mar 17 20:24 libPHPyParticleSelect.so
-rwxr-xr-x 1 maxim maxim 3846252 Mar 17 20:24 libsimreco.so
-rw-r--r-- 1 maxim maxim 2658 Mar 17 20:24 phpythia.C
-rw-r--r-- 1 maxim maxim 452 Mar 17 20:24 pythia.cfg
-rw-r--r-- 1 maxim maxim 6219 Mar 24 12:54 phpy_xsec.root
-rw-r--r-- 1 maxim maxim 4194561 Mar 24 12:54 phpythia.root
```

Prep work for the  
PHENIX image, running  
from a self-contained  
AFS area



# Docker: action items



- Complete the experimental 32-bit image setup
  - i.e. the one obtained with the library collection
  - Test and understand limitations
  - If functionality adequate commit to Docker Hub
- Discuss private hosting of images at BNL which would solve the problem of the image availability and DB access
  - SDCC managing images is by far the optimal solution
  - Private registry is possible, there is a test service
  - Access mode is similar to Docker Hub i.e. images are pulled from the service
- SDCC - at some point will need to organize a meeting with a Docker focus
- Collect use cases amenable to REANization - expert help needed - Gabor, Dillon?

# Open Data - the “ $\pi^0$ and $\gamma$ analysis” entry

- The companion document updated to reflect the latest round of changes
- ROOT5 and 6 versions of the macros created and tested
- Uploaded to CERN, admins notified, awaiting response
- Have enough experience at this point to create another entry, how do we get volunteers?
  - If there is a close-to-final Ntuple and some macros for event/track/cluster selection that would be a low-hanging fruit
  - Can create images complete with data if compact enough

# HEPData

- A new script - to aid in formatting the errors - has been developed based on the original script developed in STAR
- Sharing OK'd by Rongrong Ma (STAR), will be credited
- Corrections by Takahito and others
- Added to the repository, will make further improvements if necessary
  - <https://github.com/PhenixCollaboration/hepdata/tree/master/scripts>



# Zenodo

- More uploads of conference presentations (thanks Gabor!)
  - The “new” workflow in action, works nicely
- Correction and management of the keywords attached to Zenodo items
- A total of 340+ PHENIX items as of this week, a substantial achievement

# Website

- Added/corrected conference info on the site
- 23 conferences now listed and linked on the site: webpages and Zenodo entries
- Keywords: total of 185 now
- Cleanup/improvement of links and references in a few places (DAP, Zenodo)
- Built up the REANA page (next slide) - for concise how-to information
- Pulled to production URL: <https://www.phenix.bnl.gov/>

# Website - the updated REANA page

The screenshot shows the REANA website with a blue header containing navigation links: Experiment, Results, Detectors, Software, Analysis, and About. The main content area is titled "REANA" and includes an "Overview" section. The Overview text states that REANA is a reproducible research data analysis platform developed at CERN, used for Analysis Preservation in PHENIX. It lists two features: unambiguous description of workflows in YAML and capture/preservation of software and environment using containerization. It explains that REANA workflows are represented as Directed Acyclic Graphs (DAGs) in a YAML schema based on the Common Workflow Language (CWL), where each component may require a separate Docker container. The text also mentions that REANA workflows require a properly configured REANA cluster, with one available to CERN users and a test instance at BNL. A "Getting Started" section follows, explaining that users need an access token and that the REANA client is a Python-based tool installed on the user's machine. It provides two code blocks: one for creating a virtual environment and installing the client, and another for setting environment variables, cloning the REANA demo repository, and running a workflow. Below the code blocks, it explains that the client looks for workflow definitions in a file named reana.yaml and that the -w option defines the workflow handle. It also mentions that progress can be tracked via the Web-based GUI or the CLI, and that outputs are available for download. A final code block shows the command to delete a workflow. The page ends with a "Useful Options" link.

REANA

## Overview

REANA is a reproducible research data analysis platform developed at CERN. It is considered for Analysis Preservation in PHENIX due to the following features:

- Unambiguous description of workflows (encoded in YAML)
- Capture and preservation of both the software and the software environment by using containerization

REANA workflows can be represented as *Directed Acyclic Graphs* which is reflected in the YAML schema based on the [Common Workflow Language \(CWL\)](#). Each computational component of a workflow may require a separate and distinct *Docker container*, although individual steps can be as simple as a shell command writing a comment to a log file, in which case containers would be redundant.

Execution of workflows in REANA requires a properly configured **REANA cluster**. One such cluster is available to CERN users, and there are instances at other institutions. There is also a test instance currently being evaluated at BNL and it is available on the internal BNL network only. Access to REANA clusters is controlled by their administrators granting access tokens to qualified users. The user interacts with a REANA cluster via its network interface (HTTPS), either via the Web GUI for a quick overview of workflows in various stages of execution, or the CLI client which affords the user full access to all REANA functions. The client also makes it possible to use an automated agent for interaction with the system by scripting various actions.

## Getting Started

To be able to access a REANA cluster the user must be issued an *access token* by the administrators (this may be specific to each institution hosting its REANA facility and typically involves visiting the requisite Web page). REANA client must be installed on the user's machine. It is a Python-based tool so optimally this is done via the "virtual environment" mechanism:

```
# create new virtual environment
virtualenv ~/virtualenvs/reana
source ~/virtualenvs/reana/bin/activate
# install reana-client (may need sudo)
pip install reana-client
```

The "activate" step will be necessary if a new shell/window is created for interacting with REANA. A SSH tunnel is required to access the REANA cluster at BNL. Assuming a token has been obtained and a SSH tunnel established on port 30443 a test session might look like this:

```
# set REANA environment variables for the client
export REANA_SERVER_URL=https://localhost:30443
export REANA_ACCESS_TOKEN= # user's REANA token
# clone and run a simple analysis reana
git clone https://github.com/reana/reana-demo-root6-roofit
cd reana-demo-root6-roofit
reana-client run -w root6-roofit
```

By default the client will look up the workflow definition from the file `reana.yaml` found in the current folder. The `-w` option ("workflow") simply defines the handle/name by which this workflow will be known to the system. The name can be anything. To specify a different workflow definition file and a different name one might use something like

```
reana-client run -f my_workflow_file.yaml -w my_custom_workflow_name
```

Progress of REANA workflows can be tracked in the Web-based GUI provided by each cluster or via the CLI, `reana-client`. Likewise, outputs files generated by the workflows (including the example above) are available for download both via the GUI and the CLI. If a workflow is no longer useful it can be deleted from the REANA system:

```
reana-client delete -w my_custom_workflow_name
```

## Useful Options

# Analysis notes: technology downselect

- GitHub
  - More feedback from colleagues re: longevity - looks positive
  - Can start operating now
- Gitea@BNL info
  - Hosts 8 organizations at BNL with a total of 104 repositories
  - Expected lifetime: did not receive guidance or commitment
- Looks like the final call should be as recommended by the EC i.e. GitHub

# Plans

- *Maxim on vacation March 31<sup>st</sup> - April 11<sup>th</sup>*
  - Next meeting? Propose April 22<sup>nd</sup> (or later)
  - Docker/REANA/SDCC
  - DAP participants
- Will start hosting analysis notes on GitHub
- Docker image development is the focus area, ETA is not too clear at this point
  - There are some promising results though
- More engagement in the REANA effort?
  - Can/should start before we have containers (we already do have private ones)
  - Would consist of self-contained clean packages of macros and data