ASTERICS - H2020 - 653477

Technology benchmark report - D-INT

ASTERICS GA DELIVERABLE: D3.9

Document identifier:	ASTERICS-D3.9-final.docx
Date:	**30-04-2017**
Work Package:	**WP3 OBELICS**
Lead Partner:	**LAPP**
Document Status:	**Report**
Dissemination level:	**Public**
Document Link:	www.asterics2020.eu/documents/ASTERICS-D3.9.pdf

Abstract

ESFRI projects face many challenges in the processing and integration of their data. Original solutions are developed and tested by members of the ASTERICS Collaboration to overcome these challenges. In this report, tests on data format, data storage and transfer and data processing in an automated pipeline and in large databases are presented.

I. COPYRIGHT NOTICE

II. DELIVERY SLIP

	Name	Partner/WP	Date
From	Thomas Vuillaume	LAPP	10/04/2017
Author(s)	• Thomas Vuillaume (LAPP) • Nicolas Chotard (LAPP) • Stefan Geißelsöder (FAU) • Antonio Falabella (INFN) • Peter Hague (UCAM) • Tammo Jan Dijkema (ASTRON) • Tarek Hassan (IFAE)	• LAPP/WP3 • LAPP/WP3 • FAU/WP3 • INFN/WP3 • UCAM/WP3 • ASTRON/WP3 • IFAE/WP3	10/04/2017
Reviewed by	Giovanni Lamanna	LAPP/WP3	20/04/2017
Approved by	AMST		30-04-2017

III. DOCUMENT LOG

Issue	Date	Comment	Author/Partner
1	10/04/2017	First Draft	Thomas Vuillaume (LAPP)
2	15/04/2017	Integration of members sections	Thomas Vuillaume (LAPP)
3	26/04/2017	Corrections and introduction	Thomas Vuillaume (LAPP)
4	30/04/2017	Abstract	Thomas Vuillaume (LAPP)

IV. APPLICATON AREA

This document is a formal deliverable for the GA of the project, applicable to all members of the ASTERICS project, beneficiaries and third parties, as well as its collaborating projects.

V. TERMINOLOGY

ASTERICS	Astronomy ESFRI & Research Infrastructure Cluster
CTA	Cherenkov Telescope Array
ESFRI	European Strategy Forum on Research Infrastructures
KM3NeT	Cubic Kilometre Neutrino Telescope
LOFAR	The Low Frequency Array
LSST	The Large Synoptic Survey Telescope
OBELICS	Observatory E-environments LInked by common ChallengeS

SKA	The Square Kilometre Array

Table of Contents

I. COPYRIGHT NOTICE **1**

II. DELIVERY SLIP **1**

III. DOCUMENT LOG **2**

IV. APPLICATON AREA **2**

V. TERMINOLOGY **2**

1. Introduction **5**

2. Qserv tests & integration into science pipelines **6**

2.1 Goals and test case *6*

2.2 General work steps *7*

2.3 Current status *7*

2.4 Next steps *8*

3. Evaluation of lightweight virtualization using Docker in the context of ASTERICS to enhance reproducibility of analyses **9**

3.1 Considerations that led to Docker *9*

3.2 Comparison of Docker with alternative technologies *11*

4. Evaluation of low power-based storage solutions **13**

5. Development of image processing pipeline, web interface and workflow **14**

5.1 Pipeline *14*

5.2 Web Interface 14

5.3 Metadata Generation *15*

6. Data storage technologies used in the LOFAR Long Term Archive **16**

6.1 LOFAR archive overview *16*

6.2 Grid software *17*

6.3 Metadata software 18

6.4. Structure of data products *18*

7.Development of test DL3 converter **19**

7.1 DL3 data format *19*

7.2 MAGIC DL3 converter *19*

7.3 Next steps *20*

8. Conclusion **21**

9. References **22**

1. Introduction

Data Integration is an important part of the data processing life. After its generation, and in order to be correctly analysed, data have to be stored in such a way to be easily and quickly retrieved, without alteration (preservation of its integrity).

As presented in the previous report (D3.5), there are huge differences in the data integration challenges between ESFRI projects. Members of ASTERICS involved in these experiments are currently developing and testing specific solutions to the challenges they are facing. This process includes benchmarks of existing solutions as well as the ones under development to test their viability. This report presents their current status.

The developments and benchmarks realised concern all the stages in the life of the data - data format, data storage and transfer, integration into large databases - and processes to help its integration - automated pipelines and virtualization environments.

Most of the tests and benchmarks are done in the frameworks of the ESFRI projects but could be extended to a more general context and be re-used broadly.

2. Qserv tests & integration into science pipelines

With its large optical étendue (319 $m^2.deg^2$), the Large Synoptic Survey Telescope (LSST) will revolutionize wide field optical astronomy. Equipped with the largest camera ever designed, the telescope will allow detailed observation of the universe on an unprecedented scale. The instrument will conduct research from asteroid identification in our own solar system to the understanding of the nature of dark matter and dark energy in the universe. Operating from 2022 onward, the program will run for at least a decade. Celestial objects and their physical properties will be identified and cataloged in a database distributed over hundreds of nodes which will eventually include trillions of entries. With a volume in the order of several tens of petabytes, this catalog will play a major role in the scientific exploitation of data produced by the telescope. To meet these needs, a specific software called Qserv [QServ], is being developed by a team of engineers, the majority based at the US SLAC National Accelerator Laboratory.

2.1 Goals and test case

The two main goals of the project developed within the Asterics framework are to test the Qserv capabilities and performances on real data sets processed through the LSST data processing pipeline (a.k.a the "stack"), and to integrate its use into the science pipelines currently developed by the Dark Energy Science Collaboration (DESC).

- The tests that will be performed on Qserv will consist in testing:
- Several, simple and more complex, set of queries (magnitudes, position, galaxy shapes, etc.)
- Different configurations of the Qserv database (tables partitioning)
- Different catalogs from the LSST stack (based either on individual sources or on co-added ones), or external data-sets

The selected test case is a science pipeline currently developed by the LSST team at LAPP on galaxy cluster analysis, which allows the mass estimate of these objects and will ultimately lead us to cosmological parameter measurements. This science pipeline currently uses a few different catalogs produced by the LSST stack. These data come from public images taken by the Megacam camera at the Canada-France-Hawaii-Telescope in 5 filters (u, g, r, i, z) in several areas of the sky.

2.2 General work steps

Using the already produced data set described above as a starting point, the general steps needed to achieve the goals previously presented are as follow:

- Create a Qserv instance that can be used for test purposes, as well as explore ways to create a stable version of the database that can be used in the longer term. A stable version of the database will ultimately allow to automatically load the data produced by the LSST stack, and then perform the analyses.

- Automatically load stack-processed data into such Qserv instances. To do so, a set of tools needs to be built from scratch, as there is currently no existing software that allows to go from the LSST stack output format to the Qserv input format.

- Create a set of queries to test the system and, in the long term, use direct queries issued directly from science pipelines.

- Implement the use of Qserv into the Clusters science pipeline for test purposes and, in the longer term, into any other science analyses that could make use of it.

These steps will allow us to test many different properties of the database, to understand the various catalogues produced by the LSST software and their relationships, and ultimately to allow scientists to build their own science pipelines in coherence with the Qserv database development.

2.3 Current status

Tests have recently started at LAPP with the help of Qserv developer Fabrice Jammes (LPC Clermont-Ferrand, France). Qserv and the LSST stack are currently installed in Docker containers on the National Center for Super-Computing Applications (NCSA) cloud, which is dedicated to prototyping and development. While this server is suitable for these tests, the long-term use of Qserv in the context of our analysis will need a permanent access to a stable and running Qserv instance at CNRS/IN2P3 computing center (CC-IN2P3) in Lyon. It will indeed give us access to a larger amount of disk space and allow us to load more data on more Qserv workers.

As there is currently no automatic way to convert the output of the LSST stack to a data format that can be ingested by Qserv, we have started to write a Python library to perform the automatic conversion of the LSST catalogs into the appropriate format. There scripts are for now in constant evolution as we are still investigating several important aspects of both the database schema and the Qserv input format requirements.

From the full catalogs that we currently have, only a small part of the sky has been selected, in one filter (g), and for two distinct tables. This small amount of data allows us to quickly run the full process of conversion from the LSST stack format to the Qserv one, and load them into the Qserv instance accessible on the cloud, while modifying the codes if needed. The few first basic queries that we have constructed give identical and consistent results on a regular MySQL database and on the Qserv database.

2.4 Next steps

These first tests will soon be completed as our understanding of the Qserv tool increases, and as we make progress in describing the LSST data schema. After defining the tables of interest, and the relation between them, we should be able to load all the data for one galaxy cluster (all available filters and tables) and start to test more complex queries.

In a longer term, Qserv will have to be installed at CC-IN2P3, where the data storage will suit our needs, and an automatic ingestion of new cluster data in this database instance will thus be possible. While building the necessary Python tools to query these data, a full set of tests will be written, and these tools will be implemented in the Clusters (and possibly other) science pipeline(s). The use of Qserv in the framework of a real science analysis will allow to test the Qserv capabilities and performances in a realistic and controlled environment.

3. Evaluation of lightweight virtualization using Docker in the context of ASTERICS to enhance reproducibility of analyses

As analyses in astronomy, astrophysics and particle astrophysics tend to increase in complexity with the abilities available on modern computer systems, the basic requirement of (short and long term) reproducibility is becoming harder to achieve.

The high number of dependencies on other software packages, which have implicitly been used to obtain a result, are non-trivial to be reproduced exactly and sometimes not all dependencies are recognized explicitly.

Techniques like Docker can help not only to achieve this reproducibility, but also in situations where checks or changes are desired after the know-how is not fully available anymore, e.g. when the original creator has left a collaboration, or when a new person is to become involved in a workflow.

3.1 Considerations that led to Docker

Below we present a few alternative technologies for ASTERICS experiments that were considered along with Docker.

a. Plain scripts

Plainly executing custom scripts on a machine in its currently available state risks not to be able to reproduce the computation and results with updated hard/software. This does not constitute an acceptable solution.

b. Virtual machines

In principle, virtual machines (VMs) are a reasonable idea to keep analyses reproducible. While there are different competing standards, most of the technologies are compatible with each other. Without additional care, VMs can decrease the efficiency of computations [RM15], but as the effect is not too big for our application (and can be reduced by certain VM techniques [xen17, RM15], this is not considered a major drawback here.

The main problem with VMs for the envisioned application is that the memory requirement does not scale well enough for a large number of analyses, since a whole virtual machine has to be stored for each analysis.

c. Docker

Docker [doc17] is based on layered containerization. It allows direct access to the underlying kernel while abstracting the software dependencies. Since only the changes to a common base system required by an analysis have to be stored for an analysis, the main drawback of virtual machines in the envisioned application scenario is dealt with, namely the large memory requirements to store analyses.

A drawback of the current status of Docker is the danger of *privilege escalation*, which potentially allows users to perform actions on a machine they otherwise wouldn't have the rights to. This is no relevant concern for the application to physics analyses, but it prevents several clusters from supporting Docker directly. Desktop and cloud systems on the other hand support Docker.

d. Singularity

Singularity [sin17] is another recent option that could help with reproducibility similar to Docker. While it even deals with privilege escalations, so far it is not available on Microsoft Windows operating systems which are used by some members of some collaborations.

Furthermore the already established user basis is not as large as Dockers and therefore the risk is higher that it might not be supported in the long run .

e. Nix

Nix [nix17] also constitutes a development that could be suited to help keeping analyses reproducible. However, it focuses on an application and its dependencies. While it offers an elegant way to deal with these dependencies, it does not (and is not intended to) deal with e.g. input files. Furthermore, it doesn't support Microsoft Windows and, judging from the current size of the Nix community, is also less widespread than Docker.

f. Others

There are many other, similar approaches (e.g. rkt [cor17], Flockport [flo17]), but on the one hand, there is no essential feature lacking in Docker that would be required for our use-case. On the other hand, to achieve a long-term reproducibility, which is key to us, a solution should be supported by a large community and as many other co-operations or companies as possible. The current status is that this is fulfilled best by Docker.

Furthermore, many alternatives also support Docker images or Dockerfiles, while their extensions usually are incompatible to others.

3.3 Comparison of Docker with alternative technologies

Table 1 is an overview of the comparison with perceived most relevant disadvantages in red. While it should be clear at this point that there is no single perfect solution, Docker seems to be the best choice. It does solve the problem of reproducibility of physics analyses, is relatively easy to use [Gei17b] and doesn't have major drawbacks as far as the investigated use-case is concerned.

As this lightweight virtualization promises significant benefits, first efforts have already begun to employ Docker for KM3NeT [Gei17a]. The same concepts for packaging an analysis with Docker that are being worked out with KM3NeT (e.g. a common base image, not using the common tag ``latest'' or a default input/output interface) can also be applied for other experiments.

Table 1 - Overview on Solutions and Classification

	Plain	VMs	Docker	Nix	Singularity
Reproducible	Maybe not	Yes	Yes	Yes	Yes
All OS	Yes	Yes	Yes	No	No
Widespread[1]	Yes	Yes	Yes	No	No
Efficient	Yes	Possible	Yes	Yes	Yes
Storage required	Low	High	Moderate	Low	Moderate

[1] community for different lightweight techniques on github.com, state as of 2017-03-07

Table 2 - Comparison of Solutions and activity in development

	Commits	Contributors
Docker	31219	1627
Nix	2048	34
Singularity	5075	110
rkt	5167	177

4. Evaluation of low power-based storage solutions

Another activity concerns the deployment of parallel filesystems on storage bricks created with platforms powered by low power SoCs (ARMv7, ARMv8 and x86). Several CPU have been considered and tested, such as the Intel N3700 (Braswell), the Intel Xeon-D, the NVIDIA Tegra x1 and as soon as they will become available the next generation Intel low power SoCs, namely Apollo Lake.

So far, the most promising are the Xeon-D. We installed a test based on XeonD-1540 and on the BeeGFS filesystem developed by the Fraunhofer (Fraunhofer-Gesellschaft application-oriented research organization engaged in several fields)

Concerning data movements, INFN-CNAF production services are based on StoRM (http://italiangrid.github.io/storm/index.html) that implements the SRM protocol and on gridftp servers to realize the actual data transfers. To simplify the usage of these services and to reduce the learning curve for new users CNAF developed a data transfer application called dataclient which is currently under performance evaluation. It is a wrapper around the gridftp clients that hides the complexity of the auth/authZ infrastructure based on X509 proxy certificates. We are benchmarking in different testbeds as well as production environments such as the computing models for the Km3, Cuore and Cupid experiments. It has proved to be scalable for a throughput of a few TB per day.

5. Development of image processing pipeline, web interface and workflow

We have developed a software pipeline that permits automatic processing of large sets of observations, and quick inspection of edge cases. It features a command line interface and a web interface, built on a common set of functions. For the test case, we are using the ALMA archive as a target, performing source detection and matching up the results to established source lists.

5.1 Pipeline

The pipeline consists of several programs which operate on various stages of the data. The main parts of the pipeline are

- *match* Obtains the required products/data via astroquery
- *products* Constructs an XML metadata file that identifies the individual observations, associates files with them, and attempts to determine the types of those files (primary beam corrected images, beam flux profiles etc.)
- *image* Runs SExtractor, Aegean and a new MCMC process (described in the D-ANA report) on all products, with a configuration determined by the metadata generated above.

In addition, there is ongoing work on a feature to perform standardised reprocessing of raw data, which will be integrated into the pipeline when hardware permits. All these processes operate by default across the entire archive (or archive subset). Individual parts of the archive are automatically flagged for later review, or can be manually flagged.

5.2 Web Interface

The entire pipeline can be operated through the web interface, which accepts SAMP (Simple Application Message Protocol) signals used by software such as TOPCAT (http://www.star.bris.ac.uk/~mbt/topcat/), thus allowing a cyclic workflow of batch processing, analysis, flagging and reanalysis. A bridging script converts SAMP signals to the more widely used WebSockets protocol, and allows items selected in SAMP applications to then be displayed in the browser window for flagging. These facilities detailed work on remote data.

5.3 Metadata Generation

In order to deal with inhomogeneous or incomplete archives, metadata is needed to guide the process in each particular case. In some finished ALMA products for example, metadata provided in .fits headers is often insufficient to allow an automated script to perform source detection correctly. File names are assigned haphazardly (although this improves notably in the most recent part of the archive) and different projects choose to archive different products.

One of the issues that arises is primary beam correction. The primary beam sensitivity varies strongly with viewing angle, thus the data towards the edge of the image show reduced flux. Correcting for this introduces an increasing amount of error moving towards the edge of the image which can confuse source extraction software. Some projects store the image after primary beam correction (in some cases including the flux profile of the primary beam) whilst some projects also store the uncorrected image. The uncorrected image is easier for simpler source detection algorithms to work on due to consistent noise, but in order to find the correct flux the source must be mapped on the corrected image.

Part of the pipeline process detects primary beam flux files by using a 2D fast fourier transform. Because the primary beam has a single uniform peak it looks starkly different after FFT than an image, having a far sharper peak which is easy to detect automatically. If the pipeline then finds a product with only a flux profile and one other image, it assumes that the remaining image is primary beam corrected and recreates the uncorrected image. If there is a beam profile and two other images, it compares the beam to these images to find out which is corrected and which is uncorrected.

Files in product folders are grouped into observations and product type and this metadata is stored in an xml file for later use. Calibrators have much more standardised file names and can be reliably discarded on this basis.

6. Data storage technologies used in the LOFAR Long Term Archive

The main function of the LOFAR Long Term Archive (LTA) is to store raw and processed LOFAR data as the main interface for users to access their data as part of their project, and to have a legacy archive for the LOFAR telescope. The LTA also offers further processing or reprocessing. The LOFAR LTA currently contains about 26 PetaByte of data, distributed over several grid sites.

The LTA has been designed to store all data only on tape, with only one copy of each data product. This brings the risk of some data loss, which in our radio astronomy use case is acceptable. This application of tape storage as primary data storage is somewhat non-typical.

The main operational problems of the LTA are in maintaining a stable high throughput in the 10 Gb/s links between all sites. In this report however, we will focus on the applications used in the LTA. It should be clear that the size of the archive places extreme loads on the various applications.

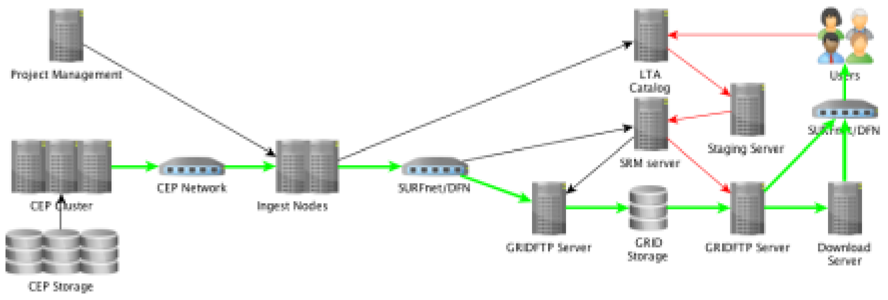
6.1 LOFAR archive overview

The raw data is collected from LOFAR stations at LOFAR Central Processing (CEP), where initial processing is also performed. A recent update of CEP introduced Docker as the main framework under which pipelines are run. The data is then transferred to one or more of the LTA locations for long term storage and further processing.

There are four LTA sites:

- Forschungszentrum Jülich: a large computing centre in Germany.
- SURFsara, a Dutch foundation that among others provides supercomputers, colocation and networking services. SURFsara is mainly located in Amsterdam.
- Target: a public-private project in the area of data management of large data volumes. The Target project is coordinated by the University of Groningen, which also hosts a cluster of several petabytes. This project has now ended, and the Target cluster is no longer actively used in the LTA.
- Poznan Supercomputing and Networking Centre: a large computing centre in Poland. Each of the LTA sites has one or more processing clusters attached to it.

An overview of the LTA layout is below:



A request is then sent to a GRID SRM server to store the data. The next step is the actual transfer of the data to a GRIDFTP server attached to a storage location. If this transfer is successful, the metadata is stored in the LTA Catalog.

A user can query the metadata through the catalog, and request datasets to be retrieved. This request is then sent to a Staging Server that makes sure the data is made available through GRIDFTP again. Then the user can either retrieve the data directly through GRIDFTP, or through HTTP (see below).

6.2 Grid software

The LTA mainly uses GRID software as the middleware. There are various implementations. Currently we have good experiences with dCache (SURFsara, Jülich), which is relatively robust. We also have good experience in getting support on dCache when we encounter bugs.

TARGET used the StoRM & GEMMS software as that has been developed to cooperate with the IBM HSM tape system used at TARGET. The communication with the outer world using SRM and GridFTP is taken care of by StoRM. Since the development of those tools focuses on specific versions of GPFS, using these tools makes it hard to pick a version of GPFS to use. TARGET has used development versions of GPFS, which did not support backups of metadata. This has proven to be a major issue. Furthermore, the GEMMS system can only trigger on percentage of disk filling. Ideally, in our use case, all data should be migrated to tape, irrespective of how full or empty the disks are. We have been able to get some support to make changes to the StoRM software, especially concerning incompatibilities between dCache and their software.

To simplify the usage of the LTA and to reduce the learning curve regarding grid certificates for new users, we developed an application which stages data through the SRM protocols and makes it available through HTTP on a download server. The performance of the HTTP download server is limited, as the bandwidth to the download server is typically much lower than that to the GridFTP-servers.

6.3 Metadata software

The astronomical metadata is stored in the LTA Catalog, which is a LOFAR-specific information system. The LTA Catalog was developed using ASTROWISE technology. AstroWise (Astronomical Wide-field Imaging System for Europe) which is an astronomy specific information system. In the LTA Catalog, SRM URLs are kept of all dataproducts. Currently, the LOFAR part of ASTROWISE contains of order 3 million dataproducts, and is about 300 GB in size. ASTROWISE runs on an Oracle database.

Finally, there are options within the LTA to do processing of the data. This mostly concerns the GRID clusters. To use these, currently the users need to have a GRID voms account, and then interact with the SRM directly to get their data onto the right processing servers. They then have to submit jobs to the GRID CREAM servers which will then do the requested processing on the GRID Cluster. The results then need to be stored again though the GRIDFTP servers into the GRID Storage, from where they can be retrieved through the means described earlier.

6.4. Structure of data products

The data products of LOFAR are stored as Measurement Sets, a structured collection of tables holding the interferometric visibility data and the metadata. The tables are stored in the CasaCore Table Data System.

7.Development of test DL3 converter

The VHE gamma-ray astronomy is evolving with CTA away from the old model of collaboration-led experiments towards that of a public observatory, where guest observers will submit observation proposals and have access to the corresponding data, software for scientific analysis and support services. We believe the open high-level data format (DL3) currently being developed for CTA (see [open-dl3]) could be extended to be used by all Imaging Atmospheric Cherenkov Telescopes (IACTs) and possibly other high energy observatories (e.g. water cherenkov detectors or even neutrino telescopes).

Following similar initiatives within other IACTs, we developed a test pipeline to convert MAGIC data products into the DL3 format as a testbed. These tools are currently being used to test and extend the proposed open DL3 format and future CTA science tools, such as *ctools* (see [ctools]) or *gammaPy* (see [gammapy]).

7.1 DL3 data format

As described in [open-dl3], data level 3 (DL3) data files are stored using the FITS file format. These files are made up of a binary table containing all event-wise parameters required by the scientific analysis of the data (e.g. event energy, time of arrival, direction, etc...) together with the instrument response function (IRF) components describing the capabilities of the detector that measured those events. By using such a format, any event-based telescope could potentially store their high-level data products, allowing the use of similar (or even identical) analysis tools.

7.2 MAGIC DL3 converter

An experimental DL3 converter was developed by extending the available MAGIC reconstruction software (MARS). This converter consists of the following methods:

- FITS exporters: MARS uses as standard data format ROOT analysis framework files (developed by CERN). Within the DL3 converter we used FITS file exporters that allowed to store event lists and IRFs following the open specifications. *CFitsIO* (see [cfitsio]) and *FlexIRF* (see [flexirf]) C libraries were used for the data format and FITS i/o.
- IRF generators: by default, MAGIC IRF components are calculated for a given region of the field of view. As the DL3 format requires these to be calculated as a function of the camera offset, dedicated tools were developed to calculate these directly from reconstructed Monte Carlo (MC) event lists.

7.3 Next steps

MAGIC DL3 converter is currently being reviewed for validation within the collaboration, and MAGIC DL3 products are being used to test open science tools. A task-force is currently being devoted to produce the first cross-IACT Crab Nebula spectrum by combining data from Fermi-LAT, H.E.S.S., MAGIC and VERITAS experiments. In a longer timescale, the MAGIC DL3 converter will be used to process all MAGIC data, producing the first MAGIC DL3 database.

8. Conclusion

As we see, the current challenges of data integration faced by the ESFRI experiments are quite different in nature. Therefore, specific solutions are being developed and tested with already some achievements:

- The LSST database solution, Qserv, that shall store all the data directly used for scientific results, is still under development and tests are now being instigated. These tests are necessary to ensure its expected performing and reveal lacks in the developments. These will help any experiment using large database to foresee potential issues.
- The problem of reproducibility concerns every experiment supposed to be running for tens of years as history as shown that informatic systems are evolving much faster than this timescale. Virtual systems such as Docker are proving to be efficient solutions.
- An automated pipeline using standard tools (astroquery) and format (xml) with a web interface is being developed. This will be of use for the whole community.
- A first prototype of a common high-level data format (DL3) for current IACT experiments. This format can be used also by other gamma-ray experiments (satellites or water cherenkov detectors) and extended for the use of any event-based instrument (e. g. neutrinos, gravitational waves).
- Original data storage and data transfer solutions are being investigated.

Even though these developments are conducted in the frameworks of the different ESFRI projects, most of them are of general nature and can be extended to other projects.

9. References

[QServ] “Qserv: A distributed shared-nothing database for the LSST catalog” - Daniel L. Wang et al. - 2011 International Conferenece for High Performance Computing, Networking Storage and Analysis (SC) - http://ieeexplore.ieee.org/document/6114487/

[cor17] coreos.com. rkt A security-minded, standards-based container engine. https://coreos.com/rkt, 2017. 2017-03.

[doc17] docker.com. What is Docker. https://www.docker.com/what-docker, 2017. 2017-03

[flo17] flockport.com. Instant apps for everyone. https://www.flockport.com/learn, 2017. 2017-03.

[Gei17a] Stefan Geißelsööder. dockerProjects. https://github.com/ sgeisselsoeder/dockerProjects, 2017.

[Gei17b] Stefan Geißelsöder. Using Docker for your analysis in KM3NeT, 2017. https://drive.google.com/open?id=0B6l8SNtndcwaeUZtUGV1TjZPclk

[nix17] nixos.org. Nix The Purely Functional Package Manager. https://nixos.org/nix, 2017. 2017-03.

[RM15] M. Komu R. Morabito, J. Kjallman. Hypervisors vs. Lightweight Virtualization: A Performance Comparison. 2015 IEEE International Conference on Cloud Engineering (IC2E), 2015.

[sin17] singularity.lbl.gov. About Singularity. http://singularity.lbl. gov/about, 2017. 2017-0.

[xen17] xenproject.org. Why Xen Project? https://www.xenproject.org/ users/why-the-xen-project.html, 2017. 2017-03.

[open-dl3] Data formats for gamma-ray astronomy: https://github.com/open-gamma-ray-astro/gamma-astro-data-formats

[ctools] GammaLib and ctools. A software framework for the analysis of astronomical gamma-ray data. J. Knödlseder, M. Mayer, C. Deil, et al. 2016, AAP, 593, A1

[gammapy] Gammapy - A Python package for gamma-ray astronomy. A. Donath, C. Deil, M. P. Arribas, et al. 2015, arXiv:1509.07408

[cfitsio] CFITSIO, v2.0: A New Full-Featured Data Interface. Pence, W. 1999, Astronomical Data Analysis Software and Systems VIII, 172, 487

[flexirf] C++ flexible Instrument Response Function I/O. https://github.com/cta-observatory/flexIRF