# The Cherenkov Telescope Array Observatory workflow management system

*Luisa* Arrabito[1], *Johan* Bregeon[2], *Alice* Faure[1,*], *Orel* Gueta[3], *Natthan* Pigoux[1], and *Andrei* Tsaregorodtsev[4]

[1]Laboratoire Univers et Particules de Montpellier, CNRS/IN2P3, Montpellier, France
[2]Laboratoire de Physique Subatomique et Cosmologie, CNRS/IN2P3, Grenoble, France
[3]Deutsches Elektronen-Synchrotron (DESY), Platanenallee 6, Zeuthen, Germany
[4]Aix Marseille University, CNRS/IN2P3, CPPM, Marseille, France

**Abstract.** The Cherenkov Telescope Array Observatory (CTAO) is the next generation ground-based observatory for gamma-ray astronomy at very high energies. It is expected to produce about 2 PB of raw data each year and to manage a global data volume which will grow through the years to reach about 100 PB in 2030. In addition, CTAO will require a high computing capacity for data processing and Monte Carlo simulations, of the order of hundreds of millions of CPU HS06 hours per year. To meet these requirements, CTAO will adopt a distributed computing model using 4 academic data centers, and will use the DIRAC framework as its workload management system. In the past ten years, to optimize the instrument design and study its performances, CTAO has used the European Grid Infrastructure (EGI) to run massive Monte Carlo campaigns. In order to handle these campaigns and to automatize simulation and data processing workflows, we have developed a production system prototype based on DIRAC. Recently, we have also developed a user interface allowing for the configuration and submission of complex workflows. In this contribution we present the production system prototype, its user interface for workflow management as well as its application to CTAO workflows.

## 1 Introduction

Currently under construction, CTAO [1] will start scientific operations in the next years for about 30 years. It will consist of tens of Cherenkov telescopes, spread between two sites: one in the Northern hemisphere in La Palma (Spain), and one in the Southern hemisphere in Paranal (Chile). CTAO will have a distributed computing model using 4 academic data centers: PIC (Spain), DESY-Zeuthen (Germany), CSCS (Switzerland), INAF/INFN Frascati (Italy). Computing and storage will be distributed among the 4 data centers. In the past, CTAO had no dedicated computing resources, so those of the EGI grid have been used, with 15 sites providing CPU on a best effort basis and 7 of them providing a 7 PB global storage capacity.

To handle workload and workflows for CTAO on this distributed infrastructure, a production system prototype has been developed, based on DIRAC [2] [3]. It is an interware used

*e-mail: alice.faure@umontpellier.fr

by many experiments to exploit distributed heterogeneous computing and storage resources. It forms a layer between a community and its various computer resources, and CTAO will use it to distribute computing tasks and data.

The CTAO production system prototype is called CTA-DIRAC [4]. It has been used to run massive Monte Carlo (MC) simulation campaigns on the EGI grid to optimize the CTAO instrument design and study its performances during the conception and construction phases. During CTAO operations, it will have to process observation data and run MC simulations to derive the instrument response functions of the telescopes. For this, it will have to run complex processing and simulation workflows.

In CTA-DIRAC, there is an Application Programming Interface (API) adapted to the CTAO workflows. Recently, we have developed a user interface to configure and submit complex workflows more easily. We have also added support for workflows written in Common Workflow Language (CWL) [5].

We outline DIRAC systems for workflow management in section 2. We describe the CTA-DIRAC software and infrastructure in section 3 as well as how we use it to manage CTAO workflows. In section 4, we describe the recently developed user interface for CTAO workflow management.

## 2 Workflow management using DIRAC

DIRAC provides workflow management functionalities through two dedicated components: the *Transformation System* and the *Production System*. In the DIRAC architecture, a *System* is a combination of software components delivering high-level complex functionalities. The *Transformation* and *Production Systems* use the DIRAC *File Catalog service* (DFC) to register and access input and output files of the workflows.

### 2.1 The File Catalog service

The DFC references all the data stored on the different storage elements. Files are registered in the DFC with their low-level metadata (size, checksum, etc.). The DFC also supports user-defined metadata. For example, in CTAO we use the metadata key "site" to reference the observation site, with values being either "LaPalma" or "Paranal". Users can issue queries to the DFC and retrieve the files whose metadata match their query. Dynamic datasets can also be created, gathering all files whose metadata correspond to a particular query. These datasets are dynamically updated when new matching files are registered in the DFC.

### 2.2 The Transformation System

A DIRAC transformation is composed of a number of identical tasks which are applied on varying input data or which differ by just one parameter. In the case of simulation or processing workflows, tasks are jobs submitted to the DIRAC *Workload Management System* (WMS). A transformation is defined by 3 attributes: a description of the tasks that compose the transformation (*task template*), a query on metadata to select input data (*data filter*) and a rule to distribute tasks and group input files (*plugin*) (Figure 1).

The transformation tasks described in the task template are all identical except for one parameter. For example, in a data processing transformation (Figure 2), the tasks are all identical but take different files as inputs. For a MC simulation transformation which does not have input data, the tasks differ for one parameter of the simulation application.

The *Transformation System* is a data-driven system: as soon as files whose metadata match the data filter are registered in the *File Catalog*, new tasks are created to process these files.

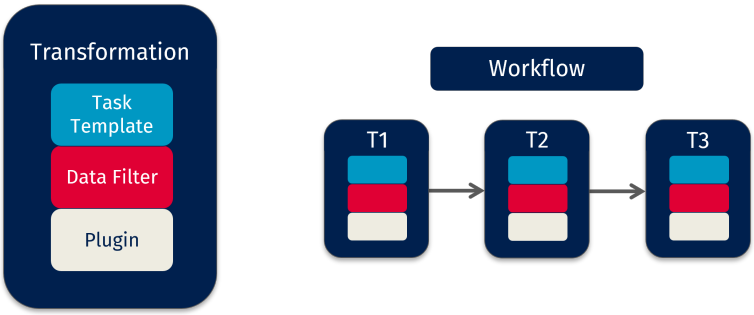


**Figure 1.** Representation of a DIRAC transformation (left) and of a workflow composed of several transformations (right).

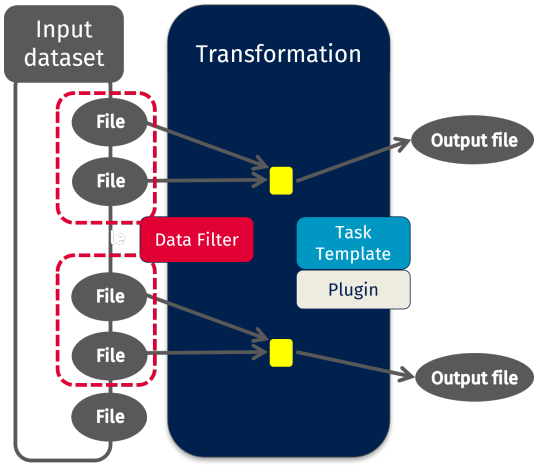


**Figure 2.** A DIRAC transformation applied to a dataset. Files whose metadata match the data filter (in red) are processed by the transformation tasks (in yellow).

### 2.3 The Production System

To submit a workflow made of several transformations, we use the DIRAC *Production System*, which is a high-level system built on top of the *Transformation System*. A DIRAC production is composed of several transformations and the information on how they are linked together. The data filter of each transformation acts as an input metadata query. If the metadata of the files produced by a transformation T1 match the metadata query on input files (i.e. the data filter) of another transformation T2, then the two transformations are connected and T2 will process the data produced by T1 (Figure 3).

Similarly to the *Transformation System*, the *Production System* is data-driven: as soon as the first transformation outputs files, the second transformation processes them. This allows to create complex data-driven workflows involving multiple transformations.

## 3 CTA-DIRAC

### 3.1 Software and infrastructure

The DIRAC architecture allows communities to develop their own extension within the common software framework [6]. For CTAO, an extension of DIRAC has been developed during the past years, called CTA-DIRAC. It mainly consists of a command line interface (CLI)

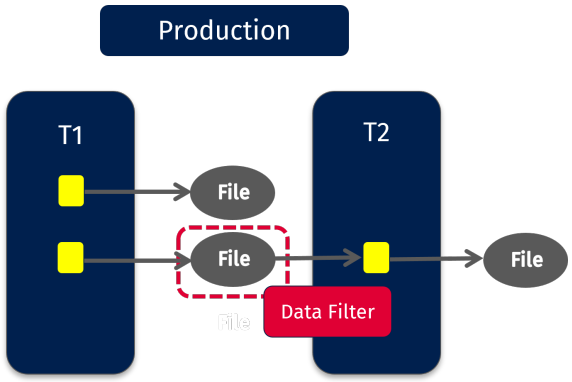


**Figure 3.** A DIRAC production composed of two transformations. Some output files of T1 (1 in this example) have metadata corresponding to the input metadata query of T2: the transformations are connected and T2 will process the file produced by T1 matching its input query.

to configure and submit CTAO workflows as well as to perform complex data management tasks. This interface makes use of the API of the different DIRAC systems involved, i.e. the *Workload Management*, the *Data Management*, the *Transformation*, and the *Production Systems* (see section 2). The CTA-DIRAC extension also includes a service dedicated to the management of the provenance metadata produced during the execution of workflows.

The DIRAC software is designed so that each community extension such as CTA-DIRAC can be deployed together with the native DIRAC components. As a prototype for the CTAO production system (which refers to the global software system for the management of CTAO workload and workflows, not to be confused with the aforementioned DIRAC *Production System*), we have deployed an instance running CTA-DIRAC together with all DIRAC components (services, agents and databases). The different services and agents are deployed on 6 servers distributed among 3 data centers. These data centers also provide database and monitoring services that are needed for CTA-DIRAC.

We have been operating this instance, together with additional services (CVMFS[1], VOMS, FTS[2]) for 10 years to run 5 massive simulation campaigns on several distributed data centers [7]. These campaigns aimed at optimizing the telescope design and studying its performance. On average over these last 10 years, these campaigns represented 100 million HS06 CPU hours per year in 2 million jobs. For the workflow management of these campaigns, we have used the *Transformation* and *Production Systems*, and for the last campaign, also the *Production System Interface* (see section 4.1).

### 3.2 Management of CTAO workflows

In the following we give an example of how we use CTA-DIRAC to execute a workflow to produce instrument response functions (IRFs) by processing simulation data.

Simulation data (data level 0, DL0) are themselves produced in dedicated workflows composed of a single transformation for each primary particle used in the simulation, i.e. gamma, gamma diffuse, electron and proton. For a given MC campaign, we thus produce 4 datasets (DL0 gamma, DL0 gamma-diffuse, DL0 electron, DL0 proton), each one containing tens of thousands of files for a global volume of hundreds of TB. Two of these datasets (DL0 gamma-diffuse and DL0 protons) are then further split into subsets that are used by machine learning algorithms either for training or for performance evaluation. In total 7 DL0 datasets are used as input of the IRF workflow.

[1] https://cernvm.cern.ch/fs/
[2] https://fts.web.cern.ch/fts/

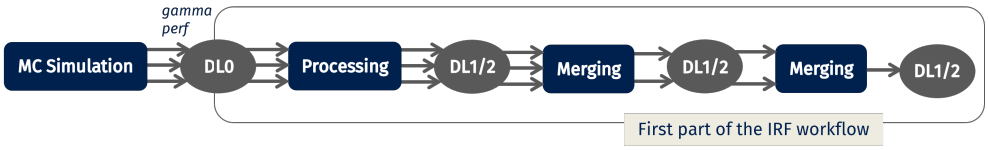


**Figure 4.** Example of workflow to produce CTAO IRFs (only the first part is shown here). The input dataset has been previously produced from Monte Carlo simulations. It consists of DL0 files. It is used in 1 processing transformation and 2 successive merging transformations to create a single output file of data level 1 and 2. These 3 transformations have to be applied to 7 different datasets.

Each dataset is first processed using the `ctapipe` application [8]. The resulting files are then merged in a two-step procedure to produce a single output file (DL1/DL2), as illustrated in Figure 4. This first part of the workflow is made of 3 transformations (1 for processing and 2 for merging) for each dataset, so that in total 21 transformations are needed. The 7 DL1/DL2 merged files we obtain undergo 8 additional transformations which produce 4 DL2 performance files (Figure 5). In total, we use 29 transformations connected in a complex workflow as shown in Figures 4-5. Finally, few further steps, not shown here, are needed to complete the workflow and produce the IRFs.

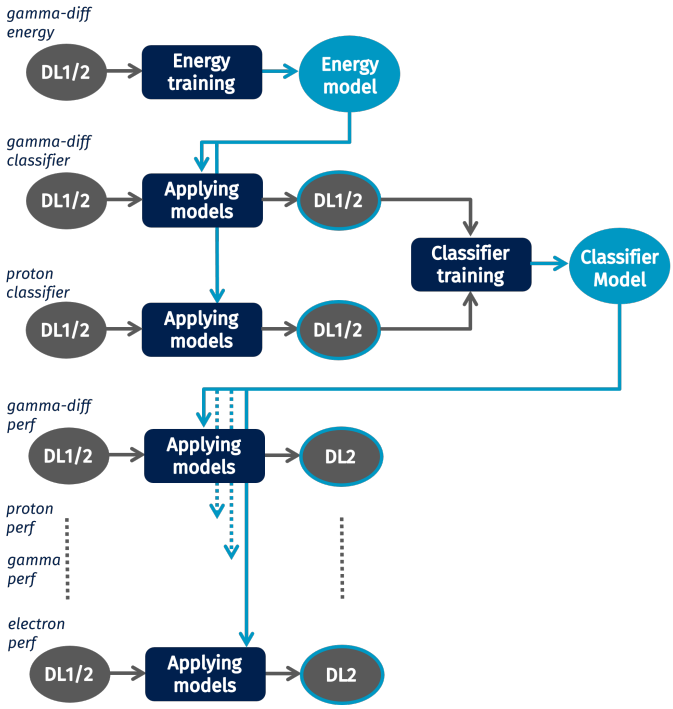


**Figure 5.** Additional processing steps of the workflow to produce the IRFs. The 7 input files correspond to the 7 output files of Figure 4. One gamma-diffuse DL1/2 file is used to train the energy model. This model is applied to another gamma-diffuse DL1/2 file and to a proton DL1/2 file. The output files are then used to train the classifier model. This model is applied for performance evaluation to the 4 DL1/2 files not yet used: 1 gamma-diffuse, 1 proton, 1 gamma and 1 electron file. This produces 4 DL2 files.

During the latest MC campaign (Prod6), which started in December 2022, we have configured and run this kind of workflow using the CTA-DIRAC *Production System* and the recently developed user interface described in section 4.1. We could have created a single production composed of several transformations and let the *Production System* take care of the execution of the whole workflow on the computing infrastructure in a fully automated way. However, since we were testing such a complex workflow for the first time, we decided to create a small number of productions and to run them one after the other. This allowed us to automatize most of the workflow while carefully monitoring some key steps.

# 4 User interface for CTAO workflow management

## 4.1 YAML-based interface

In order to submit workflows with CTA-DIRAC we used to rely directly on the *Production System API*. However, we had to write one script for each kind of workflow we wanted to submit, so it was not suitable for a non-expert user or for a complex workflow such as the one presented in section 3.2. We have thus developed a high-level interface to configure and submit productions more easily. We used this interface for the submission of the IRF workflows of the latest Monte Carlo campaign (Prod6).

The *Production System Interface* we developed uses metadata to create and link transformations. The user describes the workflow in a YAML file. Starting from this description, the interface builds a production composed of several transformations. The metadata query on input data of each transformation (i.e. the data filter) is automatically created from the step description given by the user in the YAML file. The interface allows to configure and submit productions to the *Production System*, which then runs the different transformations. The transformation jobs are submitted to the *Workload Management System* and they execute the applications requested by the user. The metadata query describing the output data is also automatically created by the interface for each transformation.

```
ProdSteps:
  - ID: 1
    input_meta_query:
    job_config:
      type: MCSimulation
      version: 2022-08-03
      site: Paranal
      particle: electron
      pointing_dir: North
      zenith_angle: 20
      n_shower: 100
      moon: dark, half
  - ID: 2
    input_meta_query:
      parentID: 1
      moon: dark
    job_config:
      type: CtapipeProcessing
      version: v0.17.0
      group_size: 5

Common:
  MCCampaign: Prod6
  configuration_id: 15
```

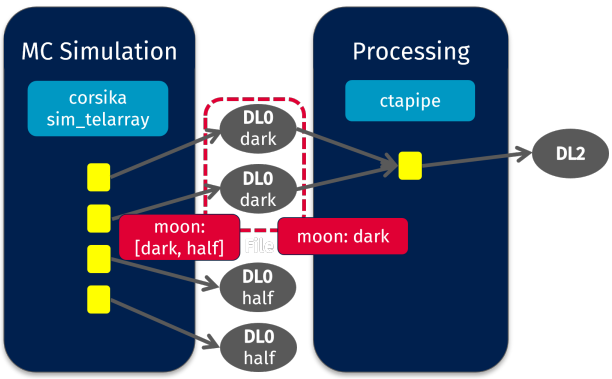


**Figure 6.** User description of a 2-step workflow (left): the first step simulates data for dark sky and sky with a half-moon; the second step processes the files for dark sky only. Corresponding production made of two transformations created by the *Production System Interface* (right).

The user can also specify how the transformations are connected, for example if one is the parent of another. The interface will ensure that the child transformation input query matches

the output query of the parent. Furthermore, it is possible to add extra metadata specifications if we want the child transformation to only use a subset of the files produced by the parent (Figure 6).

This interface has been developed in the CTA-DIRAC software extension but we plan to port it to the core DIRAC software stack so that other DIRAC users communities could benefit from it. Since the interface relies on metadata for the workflow description, each community would define its own metadata to describe its specific workflows.

### 4.2 CWL support

The Common Workflow Language (CWL) is a standard for describing workflows. In CWL, a workflow is a process characterized by multiple subprocess steps, where step outputs are connected to the inputs of downstream steps to form a directed acyclic graph (DAG), and independent steps may run concurrently [3]. DAG workflows are among the most used workflows in the HEP community, and DIRAC workflow functionalities have been developed to handle them. CTAO uses DAG workflows as well and the users of the data processing pipeline software describe some of their workflows in CWL to test and run them on their own machine.

A CWL workflow is composed of several steps which can be of two kinds: Javascript expressions or command lines. To run it locally, one can use a runner such as *cwltool*[9]. We have introduced CWL support in CTA-DIRAC to share a standard with users who use CWL to describe and run their workflows locally. We introduced this support at the level of transformations.

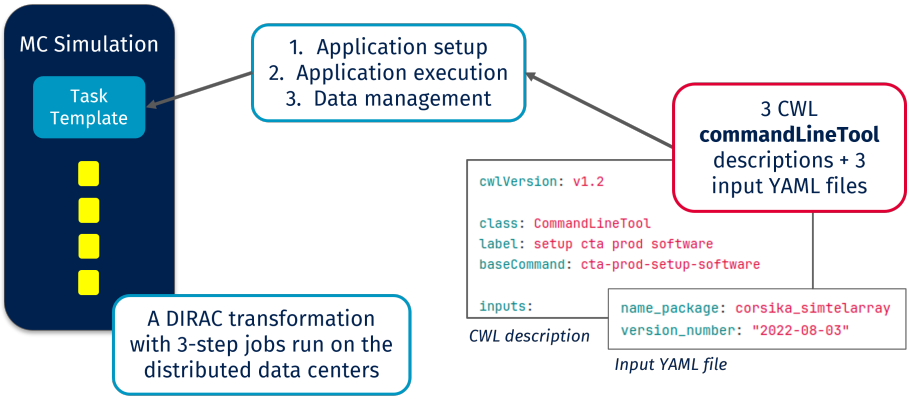


**Figure 7.** Example of the CWL support in the CTAO workflow management system for a Monte Carlo simulation. Each transformation job is composed of 3 steps described in the task template. Each one of these steps is the execution of a command line. This command line can be described in a CWL CommandLineTool. An example of a truncated CWL description and a YAML input file is given for the first command line of the job. CTA-DIRAC translates the CWL descriptions and adds instructions to create a DIRAC transformation.

Each transformation is made of identical jobs differing by a parameter (see section 2.2). For MC simulation transformations for example, the jobs are all statistically independent and differ by one parameter which is the run number used to differentiate between the files produced. Each MC simulation job is composed of the same 3 steps described in the task

[3] https://www.commonwl.org/v1.2/Workflow.html

template (see Figure 7): the application setup, the application execution and the data management of the output files. Each step corresponds here to the execution of a command line. We introduced in CTA-DIRAC the ability to create each step of the job from a CWL description in a *CommandLineTool* [4]. The CWL file describing the CommandLineTool and the YAML file specifying the inputs of the command are translated by CTA-DIRAC to run the command lines inside a transformation job.

For now, we only support CWL command lines, while in the future we also plan to support workflows described in CWL, so that the user could specify in CWL how the command lines are linked. This CWL workflow description would be translated with CTA-DIRAC and the command lines of the transformation jobs would be executed with the correct dependencies.

We also plan to investigate the possibility of using CWL at the level of productions and not only at the level of transformations.

## 5 Conclusion

To conclude, we have been successfully operating a DIRAC instance for CTAO simulations using the DIRAC *Transformation* and *Production Systems* for the workflow management. In CTA-DIRAC, we have also developed a user interface to the DIRAC *Production System* and added support for workflows described in CWL. CTAO will use CTA-DIRAC for the workload and workflow management of simulations and data processing of observation data. The next step will consist of generalizing these developments to include them in the core DIRAC software stack and make them available for all user communities.

## 6 Acknowledgements

This work was conducted in the context of the CTA Consortium. We gratefully acknowledge financial support from the agencies and organizations listed here: https://www.cta-observatory.org/consortium_acknowledgments.

O.G. acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 460248186 (PUNCH4NFDI).

## References

[1] R. Zanin *et al.*, CTA Observatory, CTA Consortium and LST, PoS **ICRC2021** 005 (2022)

[2] A. Tsaregorodtsev and the Dirac Project, J. Phys. Conf. Ser. **513** 032096 (2014)

[3] F. Stagni, A. Tsaregorodtsev, A. Sailer and C. Haen, EPJ Web Conf. **245** 03035 (2020)

[4] L. Arrabito, J. Bregeon, P. Maeght, M. Sanguillon (CTA Consortium) and A. Tsaregorodtsev (DIRAC Consortium), EPJ Web Conf. **251**, 02029 (2021)

[5] M. R. Crusoe, S. Abeln, A. Iosup, P. Amstutz, J. Chilton, N. Tijanić, H. Ménager, S. Soiland-Reyes, B. Gavrilović, C. Goble, and The CWL Community, Commun. ACM **65**, 54-63 (2022)

[6] F. Stagni, A. Tsaregorodtsev, L. Arrabito, A. Sailer, T. Hara, X. Zhang, J. Phys. Conf. Ser. **898**, 092020 (2017)

[7] L. Arrabito *et al.*, CTA Consortium and DIRAC Consortium, EPJ Web Conf. **214**, 03052 (2019)

[8] K. Kosack and M. Peresano on behalf of the CTA Consortium, PoS **ICRC2019**, 717 (2020)

[9] The Software Project, https://github.com/common-workflow-language/cwltool

[4] https://www.commonwl.org/v1.2/CommandLineTool.html