

Running productions with DIRAC

L. Arrabito¹, F. Stagni²

¹*LUPM CNRS/IN2P3, France*

²*CERN*

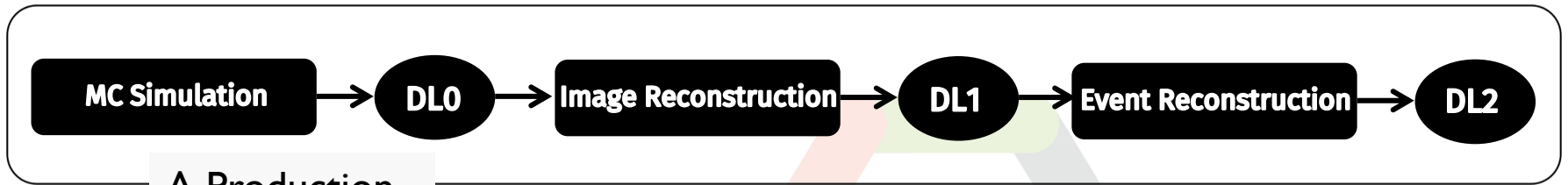


DIRAC & Rucio User Workshop

16th – 20th October 2023, KEK

- ▶ The DIRAC communities that need to manage massive productions with complex workflows use the Transformation System (see [doc here](#)) combined with a higher level system built on top of the it, called Production System
- ▶ Historically each of these communities has developed its own Production System
- ▶ Since v7r0 DIRAC provides its own Production System which is used by CTAO for now (see [doc here](#))
- ▶ We present here how CTAO and LHCb run their productions with DIRAC and LHCbDIRAC Production Systems

Simple workflow example



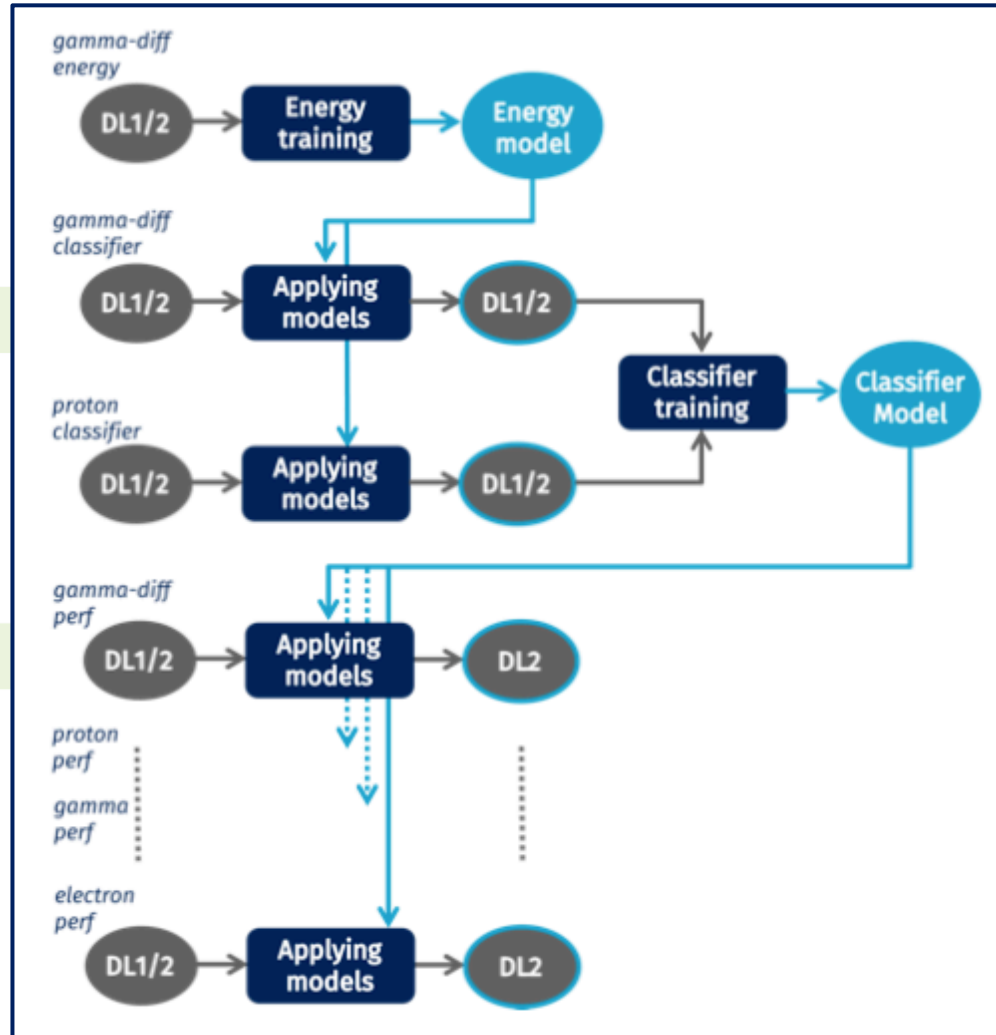
The Production System is a high-level system built on top of the Transformation System.

It automatically instantiates the different transformations that compose a Production.



Two transformations are connected if the output data of T1 intersects the input data of T2. The workflows are **data-driven**.

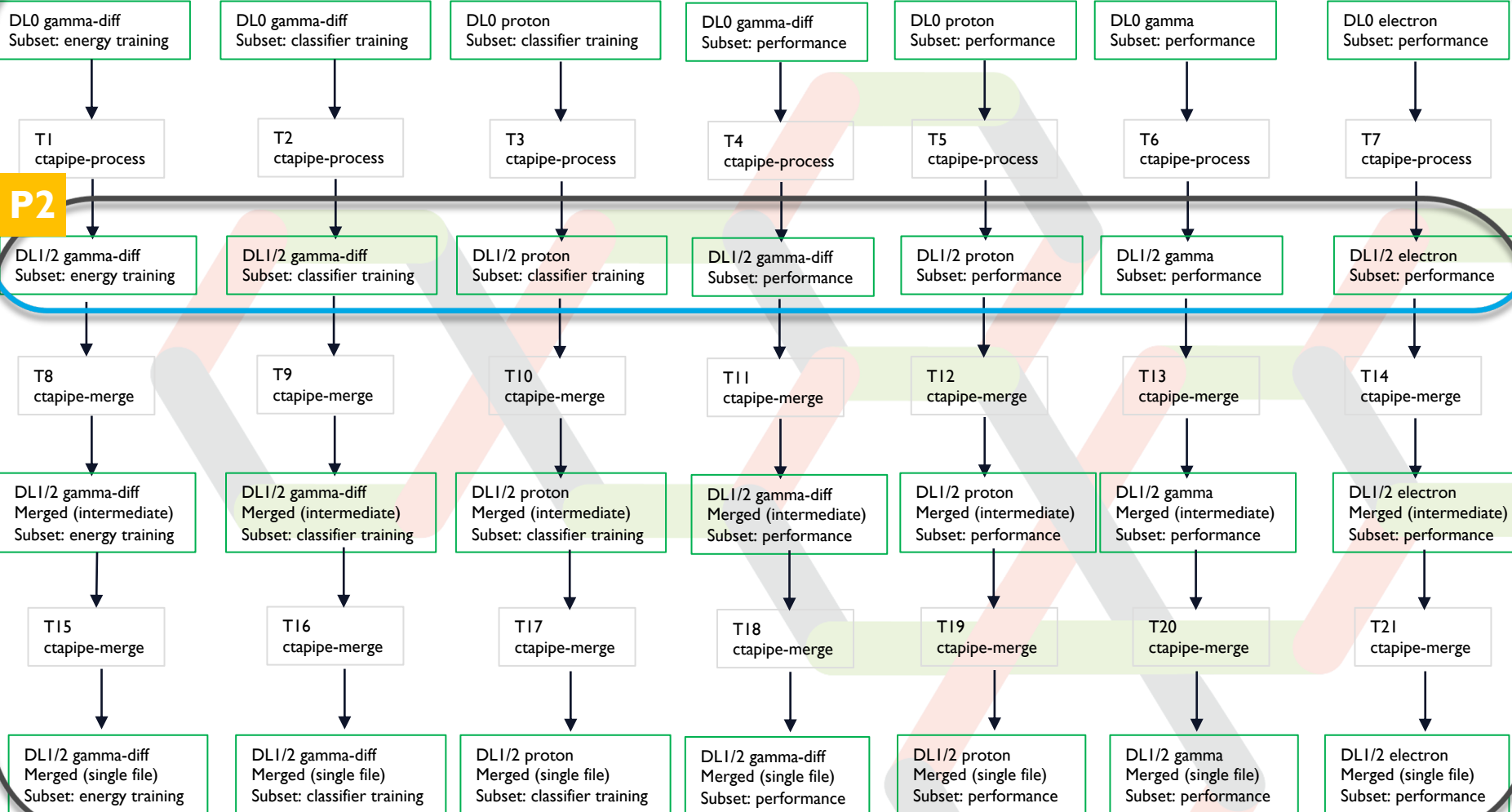
CTAO processing workflow example



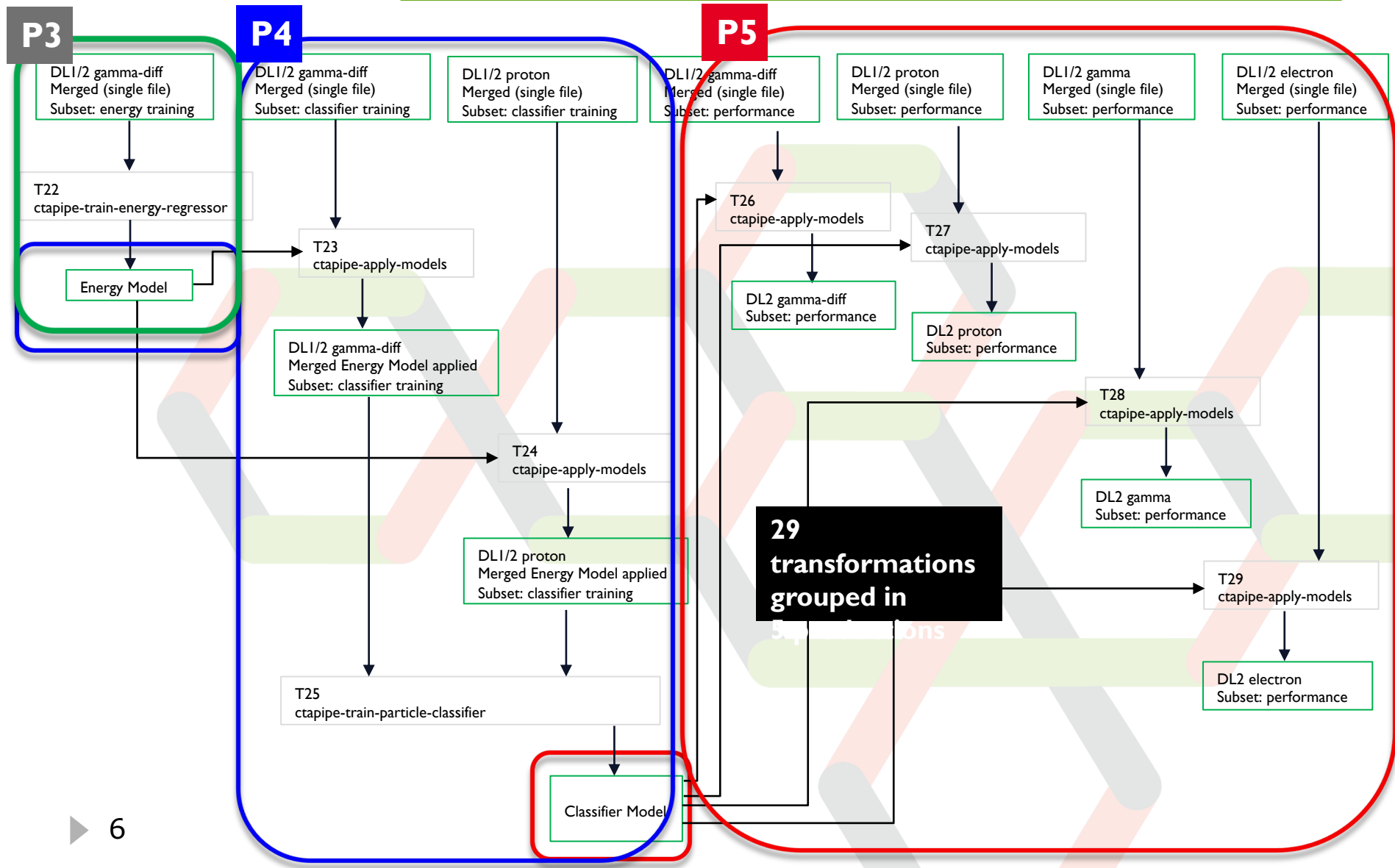
CTAO processing workflow example (detailed view 1/2)

PI

P2



CTAO processing workflow example (detailed view 2/2)



- ▶ The Production System user interface comprises a Python API a limited CLI
 - > Not very practical to configure and submit complex workflows
- ▶ For our convenience in CTADIRAC we have developed a YAML-based user interface (A. Faure)
- ▶ Currently it's specific to CTAO but it can be generalized and port it to vanilla DIRAC

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
    dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
    parentID: 2
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

CLI User Interface

```
cta-prod-submit <prodName> <workflow.yml>
```



Production System

Transformation System

WMS

Computing resources

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
      dataset: Prod5b-LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
    parentID: 2
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

A production is described by several steps, i.e. transformations

Step 1

Step 2

Step 3

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
  parentID:
  dataset: Prod5b_LaPalma_AdvancedBaseline_N5b1x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
  parentID: 1
  dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
  parentID: 2
  dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
    dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
    parentID: 2
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query
- Job configuration

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
  parentID:
  dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
  parentID: 1
  dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
  parentID: 2
  dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query
- Job configuration

An input data query can be specified :

- By a dataset

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
    dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
- ID: 3
  input_meta_query:
    parentID: 2
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 2
    output_extension: alpha_train_en_merged.DL2.h5
    options: --no-dl1-images --no-true-images
    catalogs: DIRACFileCatalog

Common:
  MCCampaign: Prod5bTest
  configuration_id: 15
  base_path: /vo.cta.in2p3.fr/tests/prodsys/MC
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query
- Job configuration

An input data query can be specified :

- By a dataset
- By a parent step (i.e. a query on the metadata of the outputs of the parent)

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
      dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query
- Job configuration

An input data query can be specified :

- By a dataset
- By a parent step (i.e. a query on the metadata of the outputs of the parent)
- By a set of meta-data key-values

Another production example

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
      MCCampaign: PRODS
      array_layout: AlphaNectarcam
      site: Paranal
      particle: gamma-diffuse
      split: train_en
      thetaP: 20.0
      phiP: 0.0
      analysis_prog: ctape-merge
      analysis_prog_version: v0.19.3
      data_level: 2
      outputType: Data
      configuration_id: 8
      merged: 2
      moon: dark
  job_config:
    type: CtapipeTrainEnergy
    version: v0.19.3
    options: -c v3/train_energy_regressor.yml
```

Production description in YAML

```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
      dataset: Prod5b_LaPalma_AdvancedBaseline_NS81x_electron_North_20deg_R1
  job_config:
    type: CtapipeProcessing
    version: v0.19.2
    array_layout: Alpha
    group_size: 2
    output_extension: DL2.h5
    data_level: 2
    options: --config v3/dl0_to_dl2.yml --config v3/prod5b/subarray_north_alpha.yml
- ID: 2
  input_meta_query:
    parentID: 1
    dataset:
  job_config:
    type: Merging
    version: v0.19.2
    group_size: 5
    output_extension: merged.DL2.h5
```

A production is described by several steps, i.e. transformations

Each step is described by :

- Input data query
- Job configuration

An input data query can be specified :

- By a dataset
- By a parent step (i.e. a query on the metadata of the outputs of the parent)
- By a set of meta-data key-values

For each step, the metadata of the outputs are automatically built from input data and job configuration

Another production example

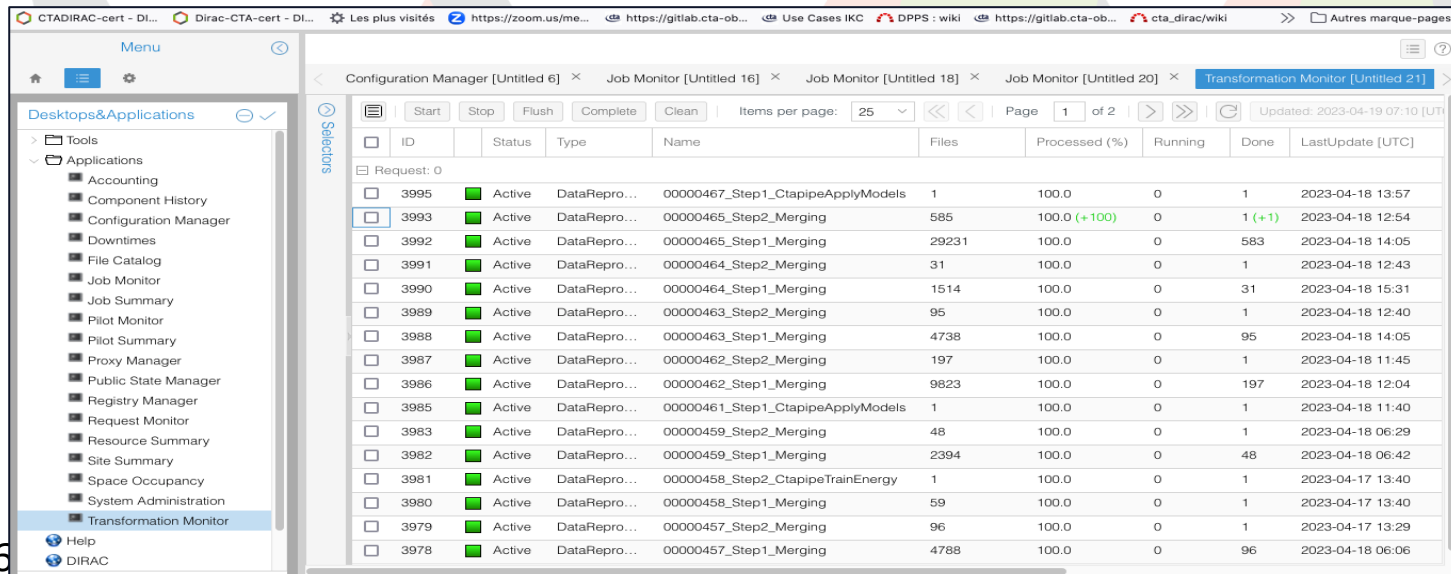
```
ProdSteps:
- ID: 1
  input_meta_query:
    parentID:
      MCCampaign: PRODS
      array_layout: AlphaNectarcam
      site: Paranal
      particle: gamma-diffuse
      split: train_en
      thetaP: 20.0
      phiP: 0.0
      analysis_prog: ctapipe-merge
      analysis_prog_version: v0.19.3
      data_level: 2
      outputType: Data
      configuration_id: 8
      merged: 2
      moon: dark
  job_config:
    type: CtapipeTrainEnergy
    version: v0.19.3
    options: -c v3/train_energy_regressor.yml
```

Production System CLI : start, stop, complete, monitor productions, etc.

```
$ dirac-prod-get-trans 649
```

	<i>TransformationName</i>	<i>Status</i>	<i>F_Proc.</i>	<i>F_Proc.(%)</i>	<i>TransformationID</i>
1	00000649_Step10_Merging	Active	38	100.0	4468
2	00000649_Step11_Merging	Active	9977	100.0	4469
3	00000649_Step12_Merging	Active	139	100.0	4470
4	00000649_Step13_Merging	Active	5	100.0	4471

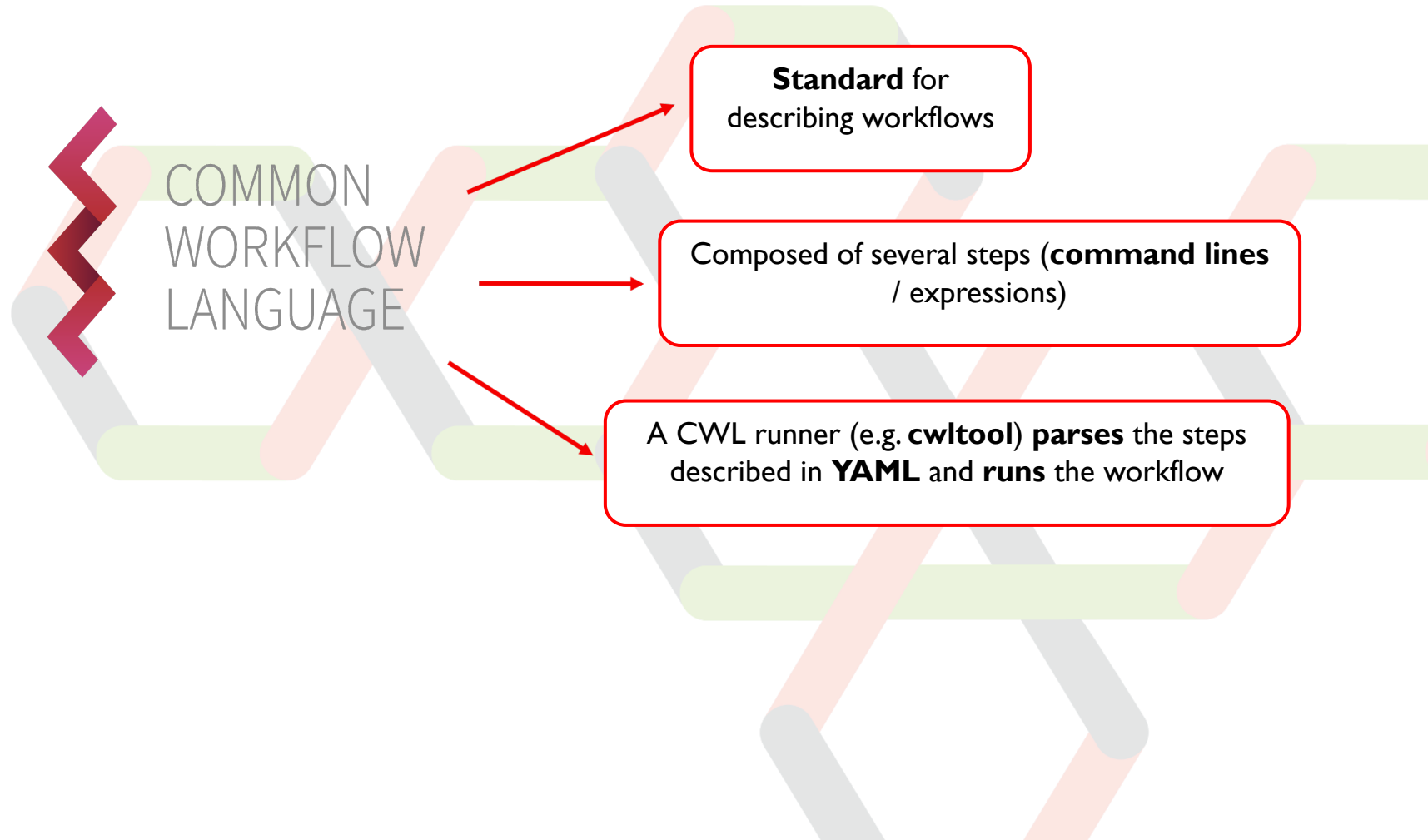
WebApp Transformation Monitor



The screenshot shows the 'Transformation Monitor' web application. The interface includes a navigation menu on the left with categories like 'Tools' and 'Applications'. The main content area displays a table of jobs with columns for ID, Status, Type, Name, Files, Processed (%), Running, Done, and LastUpdate [UTC]. The table shows several active jobs, with the second row (ID 3993) highlighted. The status of all jobs is 'Active' and 'Processed (%)' is '100.0'.

ID	Status	Type	Name	Files	Processed (%)	Running	Done	LastUpdate [UTC]
3995	Active	DataRepro...	00000467_Step1_CtapeApplyModels	1	100.0	0	1	2023-04-18 13:57
3993	Active	DataRepro...	00000465_Step2_Merging	585	100.0 (+100)	0	1 (+1)	2023-04-18 12:54
3992	Active	DataRepro...	00000465_Step1_Merging	29231	100.0	0	583	2023-04-18 14:05
3991	Active	DataRepro...	00000464_Step2_Merging	31	100.0	0	1	2023-04-18 12:43
3990	Active	DataRepro...	00000464_Step1_Merging	1514	100.0	0	31	2023-04-18 15:31
3989	Active	DataRepro...	00000463_Step2_Merging	95	100.0	0	1	2023-04-18 12:40
3988	Active	DataRepro...	00000463_Step1_Merging	4738	100.0	0	95	2023-04-18 14:05
3987	Active	DataRepro...	00000462_Step2_Merging	197	100.0	0	1	2023-04-18 11:45
3986	Active	DataRepro...	00000462_Step1_Merging	9823	100.0	0	197	2023-04-18 12:04
3985	Active	DataRepro...	00000461_Step1_CtapeApplyModels	1	100.0	0	1	2023-04-18 11:40
3983	Active	DataRepro...	00000459_Step2_Merging	48	100.0	0	1	2023-04-18 06:29
3982	Active	DataRepro...	00000459_Step1_Merging	2394	100.0	0	48	2023-04-18 06:42
3981	Active	DataRepro...	00000458_Step2_CtapeTrainEnergy	1	100.0	0	1	2023-04-17 13:40
3980	Active	DataRepro...	00000458_Step1_Merging	59	100.0	0	1	2023-04-17 13:40
3979	Active	DataRepro...	00000457_Step2_Merging	96	100.0	0	1	2023-04-17 13:29
3978	Active	DataRepro...	00000457_Step1_Merging	4788	100.0	0	96	2023-04-18 06:06

- ▶ Initial support of transformations described in CWL (see Alice's talk at CHEP 2023)

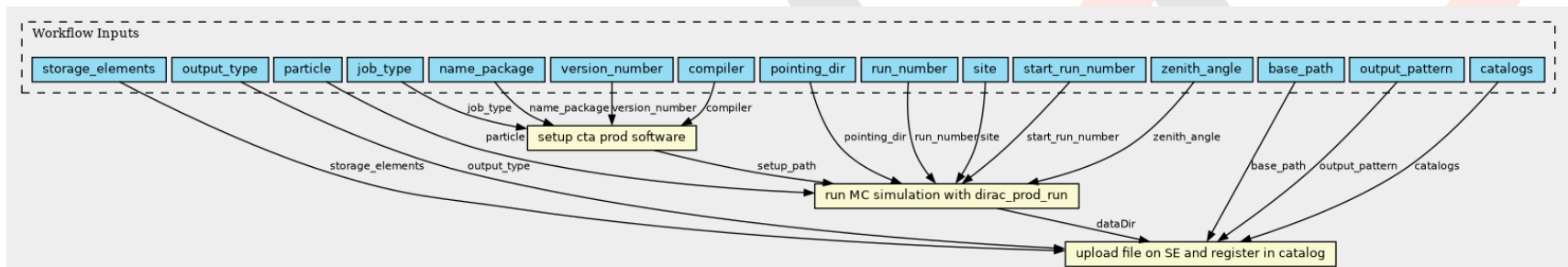


► Motivations

- Use a common workflow language both for local execution (e.g. by software developers for testing purpose) and execution through DIRAC

► Advantages of using CWL

- Standard language
- Syntax validation
- Workflow graph automatically generated



- ▶ In principle CWL could be used to describe workflows at different levels :
 - ▶ Job workflows : the different steps (executables) executed within a job
 - ▶ Production workflows : the different transformations composing a production

In CTADIRAC we have developed a tool :

cta-prod-submit-from-cwl

to parse CWL descriptions at job level and to submit transformations

cta-prod-submit-from-cwl

1. Parses a workflow description in CWL using cwltool functions to extract the command lines and builds a DIRAC job (using DIRAC Job API)
2. From the job description, it builds and submits a transformation

Transformation

```
trans = Transformation()
trans.setType("MCSimulation")
trans.setBody(job.workflow.toXML())
trans.addTransformation()
```

DIRAC job

```
job.setExecutable("env setup")
job.setExecutable("run application")
job.setExecutable("data management")
```

```
name_package: corsika_simtelarray
version_number: "2022-08-03"
```

Input YAML file

cwlVersion: v1.2 CWL description

```
class: CommandLineTool
Label: setup cta prod software
baseCommand: cta-prod-setup-software
```

```
inputs:
  name_package:
    type: string
    inputBinding:
      prefix: -p
      position: 1
  version_number:
    type: string
    inputBinding:
      prefix: -v
      position: 2
```

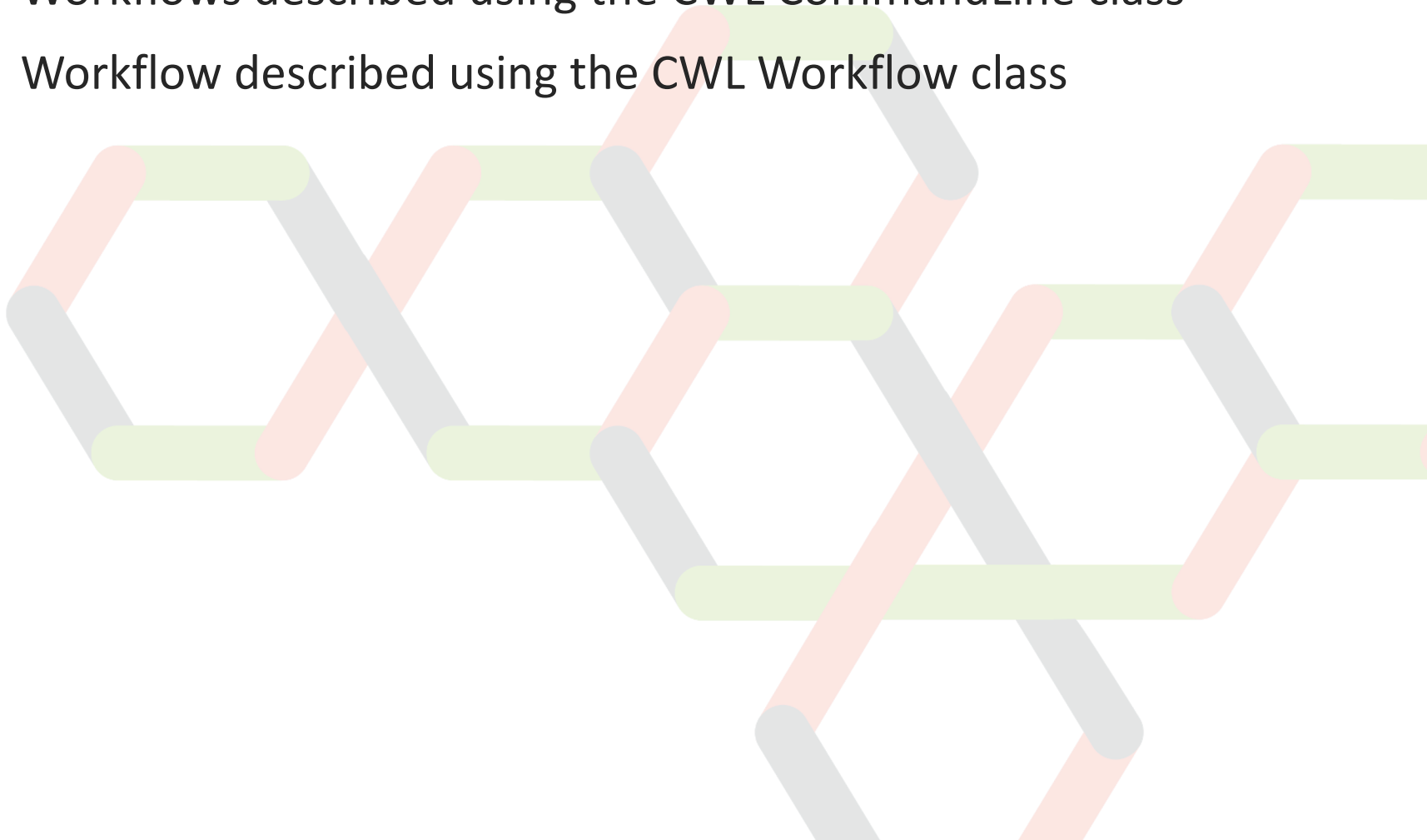
```
outputs:
  dirac:
    type:
      type: array
      items: File
    outputBinding:
      glob: "dirac*"
```

- ▶ We want to generalize our tool to build any kind of transformation
- ▶ We would like to explore the possibility to use CWL to describe DIRAC productions
- ▶ Explore how to take benefit from CWL scatter/gather functionality



Backup

- ▶ ***cta-prod-submit-from-cwl*** supports
 - ▶ Workflows described using the CWL CommandLine class
 - ▶ Workflow described using the CWL Workflow class



cta-prod-submit-from-cwl

- ▶ Uses *cwltool* functions to generate the command lines from CWL description
- ▶ Builds DIRAC Jobs from the generated command lines

```
job.setExecutable( "command-line-1")  
job.setExecutable( "command-line-2")  
....
```

- ▶ Current limitations
 - ▶ It doesn't support Javascript expression
 - ▶ All inputs needed for the execution must be present in the command line, *e.g.* we don't support *initialWorkDirRequirement*

- ▶ A CWL workflow is made of different steps
 - ▶ The execution order depends on their I/O relations
 - ▶ If 2 steps are independent *cw/tool* executes them in parallel

cta-prod-submit-from-cwl

- ▶ Determines the execution order using the same logic as *cw/tool* and builds DIRAC jobs but with all steps in sequence

```
Step1
  Inputs : in1, in2
  Outputs : out1
Step2
  Inputs : out1
  Outputs : out2
```



```
job.setExecutable("command-line-1")
job.setExecutable("command-line-2")
....
```

- ▶ **Current limitations**

- ▶ The tool is not generalized yet to build any kind of transformation starting from CWL description
 - ▶ The Body of the transformation is built from CWL description, while the other attributes are not, i.e. Type, Group Size, InputDataQuery, etc.

- ▶ **CWL functionalities not supported by our tool**

- ▶ I/O of array type
- ▶ Sub workflows
- ▶ Conditional workflows
- ▶ Scattering workflows

- ▶ We could imagine a CWL description where
 - ▶ A CWL Workflow corresponds to a DIRAC production
 - ▶ CWL sub-workflows correspond to DIRAC transformations
 - ▶ In order to link the different transformations the Production System uses their Input and Output Queries
 - ▶ We could determine which sub-workflows are linked together based on their I/O
 - ▶ For each sub-workflow we build a transformation where the OutputQuery is built from the job parameters and the InputQuery is built from the OutputQuery
- > However this looks quite specific to each users community