ALEPH DATA IN EDM4HEP

Status of the project and future outlook

Gerardo Ganis, Marcello Maggi, Juraj Smiesko, Jacopo Fanini

Introduction

- This work follows the arguments presented at the previous DPHEP workshop
  - G. Ganis, Opportunities offered by LEP data@EDM4hep

- It is argued that migration to a ***modern*** **format**
  - Enhances data preservation perspectives
  - In this moment would be of great value for the EW/Higgs factory communities
    - In particular, FCC-ee, for which a feasibility study was in the meantime asked for by the CERN council

- Demonstrator with **ALEPH data**
  - Code and data available on CernVM-FS and EOS

- Modern format: **EDM4hep,** common event data model
  - Part of the Key4hep initiative, addressing future project software needs

Motivations recall

- **Data preservation:** to conserve the possibility and capacity of extracting new science from the data
- **EDM4hep test:** to use the new Event Data Model for the first time with real, non simulated data
- **Training on real data:** to give physicists the opportunity to train by analyzing real data, with a view to FCC-ee
- **New analysis and optimization of algorithms:** to apply and test new analysis techniques (e.g. machine learning algorithms) on LEP data
- **Validation of simulations tools**

See also:
M. Maggi, ALEPH data in Key4HEP, FCC Physics Workshop
J. Fanini, ALEPH data in key4hep, FCC Software Meeting

Data preservation aspects

- EDM4hep, part of Key4hep framework, might become a general and standard data format for data structures and file format of future experiments
  - Longer perspective than an experiment-related format

- A migration of data to a standardized format allows to satisfy FAIR principles
  - **Findable**: new files will be on EOS (as ALEPH data are now)
  - **Accessible**: detach from out-of-dated operative systems
  - **Interoperable**: single standardized framework
  - **Reusable**: no need of experiment-specific expertise

- **Conversion goal**: at least DP level 3 equivalent
  - Energy flow
  - Vertexes w/ covariances
  - Tracks w/ covariances
  - Calo Objects
  - …

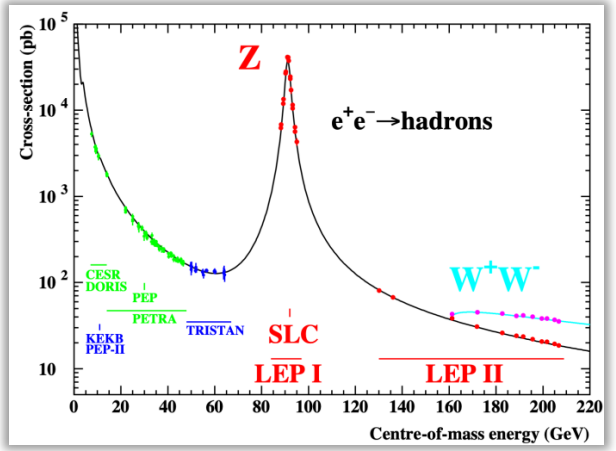
“Perform complete analyses when the existing detector reconstruction and simulated data sets are adequate for the pursued goal”

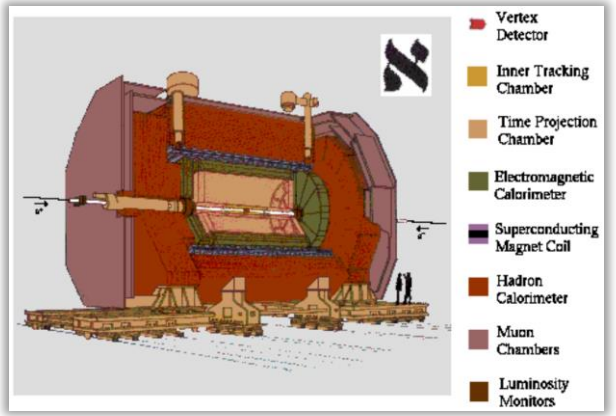
ALEPH Data Reminder

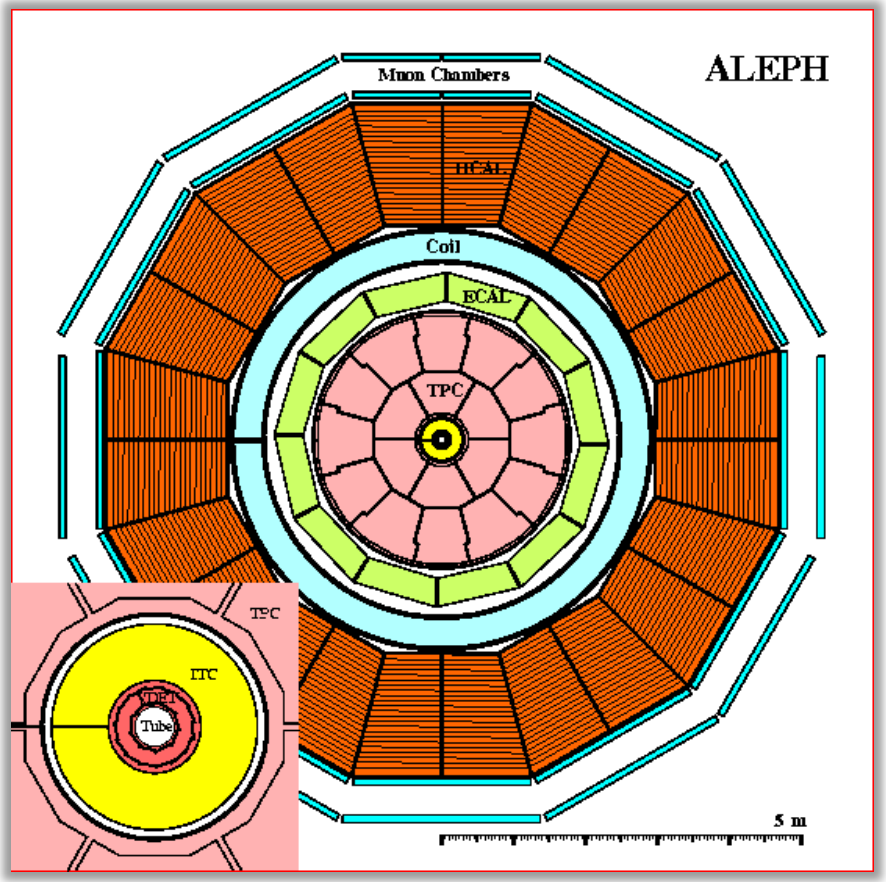
LEP and ALEPH

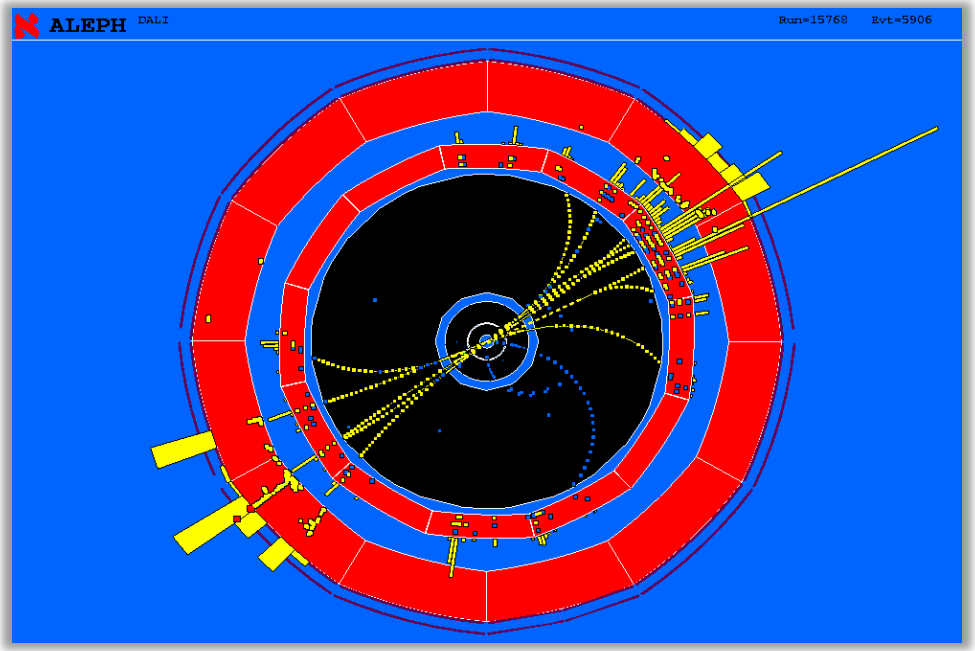
- **LEP**
  - e^+e^- collider
  - Phase 1 (‘89-’95): Z production @ ~91 GeV
  - Phase 2 (‘96-’00): W-pair production @ ~160-209 GeV
  - 4 experiments: DELPHI, L3, OPAL and…

- **ALEPH**
  - Typical “onion” experiment: vertex detector, tracking, solenoid magnet, calorimetry, muon system
  - Luminosity
    - LEP1: 200 pb^{-1}
    - LEP2: 688 pb^{-1}
  - Statistics:
    - ~ 4 x 10^6 $e^+e^- \to q\bar{q}$, ~ 8 x 10^3 $e^+e^- \to W^+W^-$






ALEPH
Muon Chambers
HCAL
Coil
ECAL
TPC
TPC
ITC
VDET
Tube
5 m


ALEPH
DALI
Run=15768
Evt=5906

Formats and sizes

- Several formats:
  - **RAW**: direct information from detector, no reconstruction
  - **POT**: reconstructed data
  - **DST**: as POT, but without noise and background
  - **Mini-DST**: high level analysis results, scaled, integerized and compressed

- Sizes (LEP1 data sample):
  - RAW: 2063 GB
  - POT: 975 GB
  - DST: 154 GB
  - Mini-DST: 38 GB

Focus on Mini-DST files

- Reduced format created from DST for space saving
- One run record per run and at least one event record per run
  - Event records: tracks, vertices, calorimetric objects, energy flow and jets, γ, e, μ identification, HV detector status, trigger
- Both Mini-DST and DST available on EOS at /eos/experiment/aleph
- Direct access to DST files possibly as a next step

Computing environment

- Last binary build: **Linux SLC4**
- Last functional environment: **Linux SLC6** (bit to bit validation, no recompilation needed)
  - GCC 3.4, G77 3.4
  - CERN Library 2005

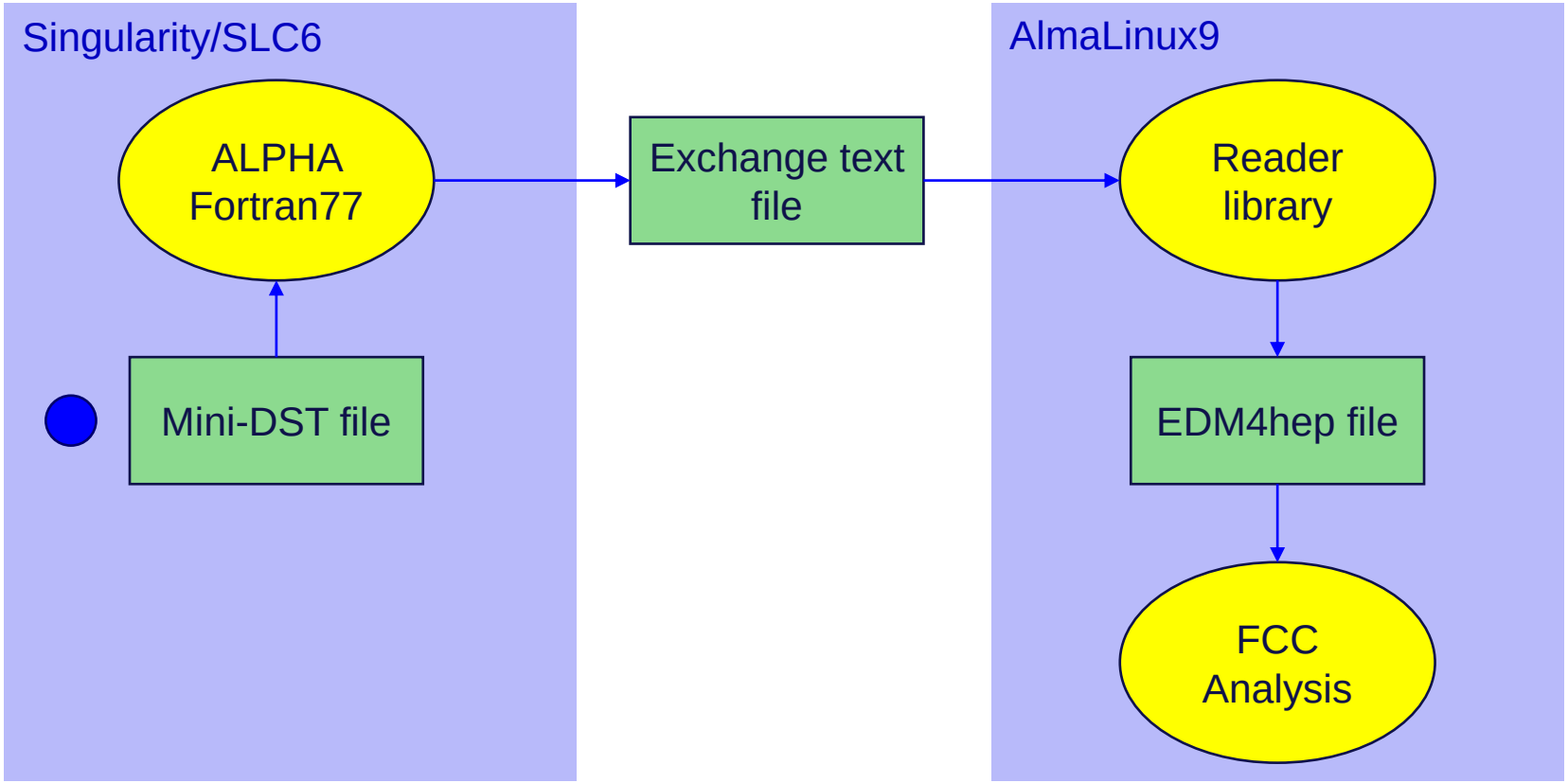
  Available at /cvmfs/aleph.cern.ch

- These environments can still be recreated on **today's lxplus** (**AlmaLinux9**) via **Apptainer/Singularity** + **CernVM**

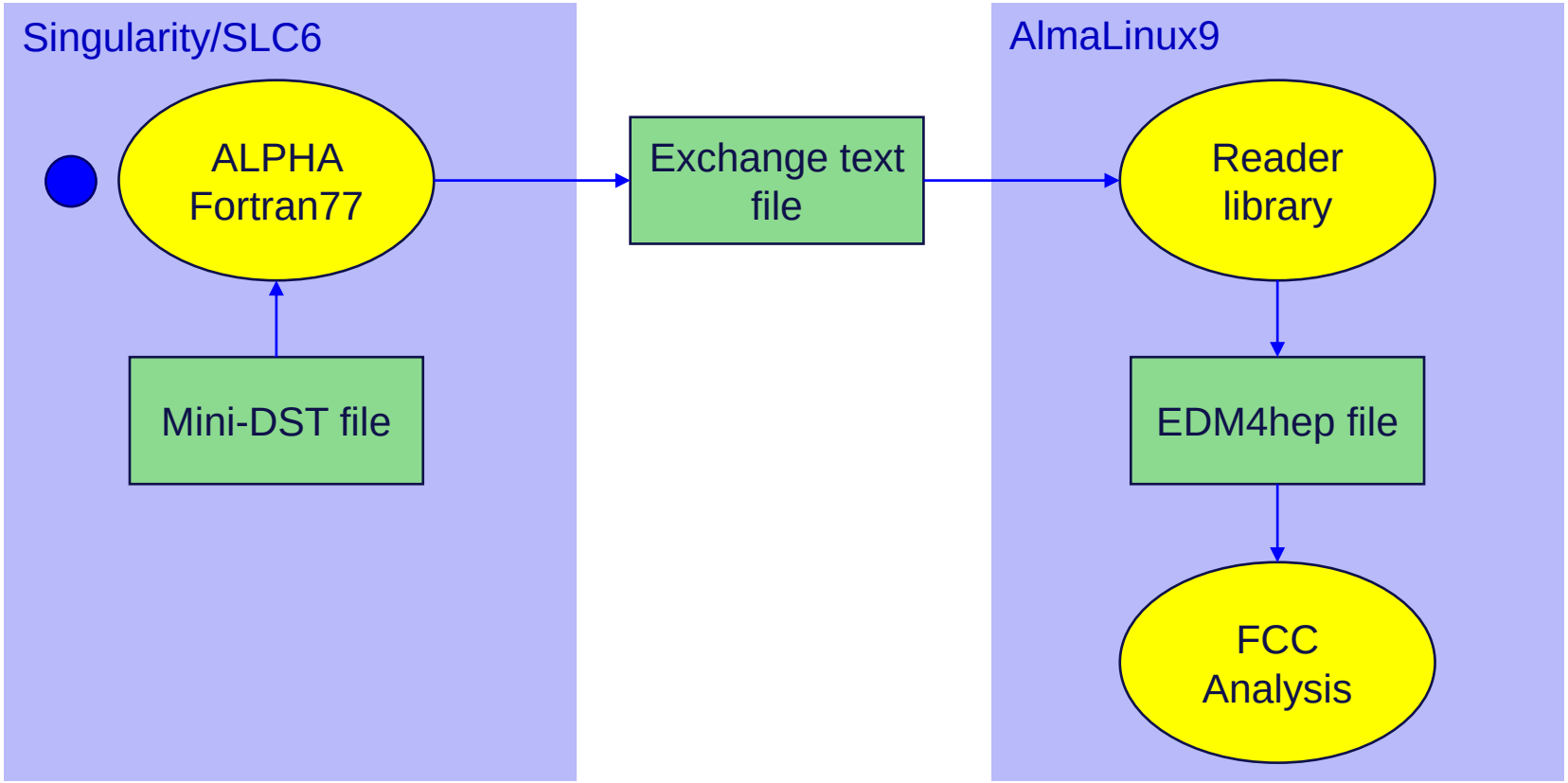
```
[jfanini@lxplus973 ~]$
[jfanini@lxplus973 ~]$ aleph-slc6

 Welcome to ALEPH @ SLC6

 HOME     =  /afs/cern.ch/user/j/jfanini/public/aleph
 WORKDIR  =  /afs/cern.ch/user/j/jfanini
 ALEPHGIT =  /afs/cern.ch/user/j/jfanini/public/aleph/GIT
Singularity SLC6:/afs/cern.ch/user/j/jfanini>
```

Conversion chain

Workflow


Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

Workflow


Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

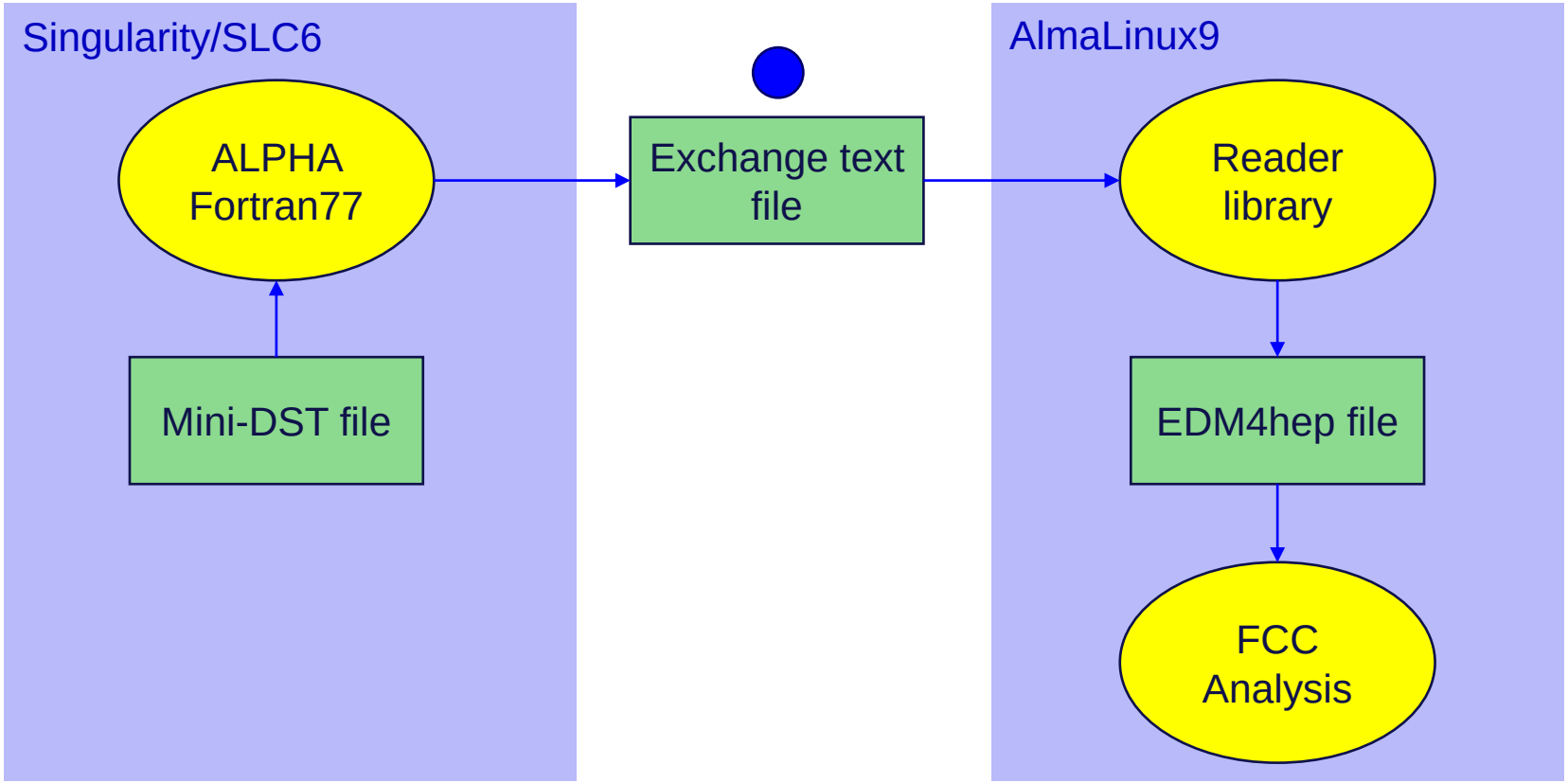
ALPHA

- The **ALeph PHysics Analysis** tool (ALPHA) has been used to loop over the ALEPH Mini-DST banks and print out them directly

- Some issues:
  - Two types of banks: linear and tabular
  - In Mini-DST data are stored scaled and integerised, then ALPHA use them to fill POT/Julia banks, where data are mainly floats (and some strings)
  - Some correction/calibration data are hardcoded in ALPHA algorithms and not included in the data banks

```
      SUBROUTINE DUMPBANKI(NAME, UNIT, IER)
      IMPLICIT    NONE
#include "qdecl.h"
#include "qcde.h"
      CHARACTER*4 NAME
      INTEGER     UNIT, IER
C - Local variables
      INTEGER     I, J
      INTEGER     NLINK, IBK, IBANK, NROWS, NCOLS, NAMIND
C - ALPHA macros
#include "qmacro.h"
      IER = 0
C - Connect to the bank
      IBK= NAMIND(NAME)
      IF( IBK.EQ.0 ) THEN
        WRITE(*,*) 'DUMPBANK: Bank ', NAME, ' cannot be found! '
        IER = 1
        RETURN
      END IF
      IBANK = IW(IBK)
C - Number of rows
      NROWS = LROWS(IBANK)
      NCOLS = LCOLS(IBANK)
      WRITE(UNIT, *) NAME, NROWS, NCOLS
C - Loop
      COUNTER = 3
      DO I=1,NROWS
        DO J=1, NCOLS-1
            WRITE(UNIT,1001) ITABL(IBANK,I,J)
        END DO
           WRITE(UNIT,2001) ITABL(IBANK,I,NCOLS)
      END DO
 1001 FORMAT(I20, $)
 2001 FORMAT(I20)
      RETURN
      END
```

Workflow


Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

Text exchange format

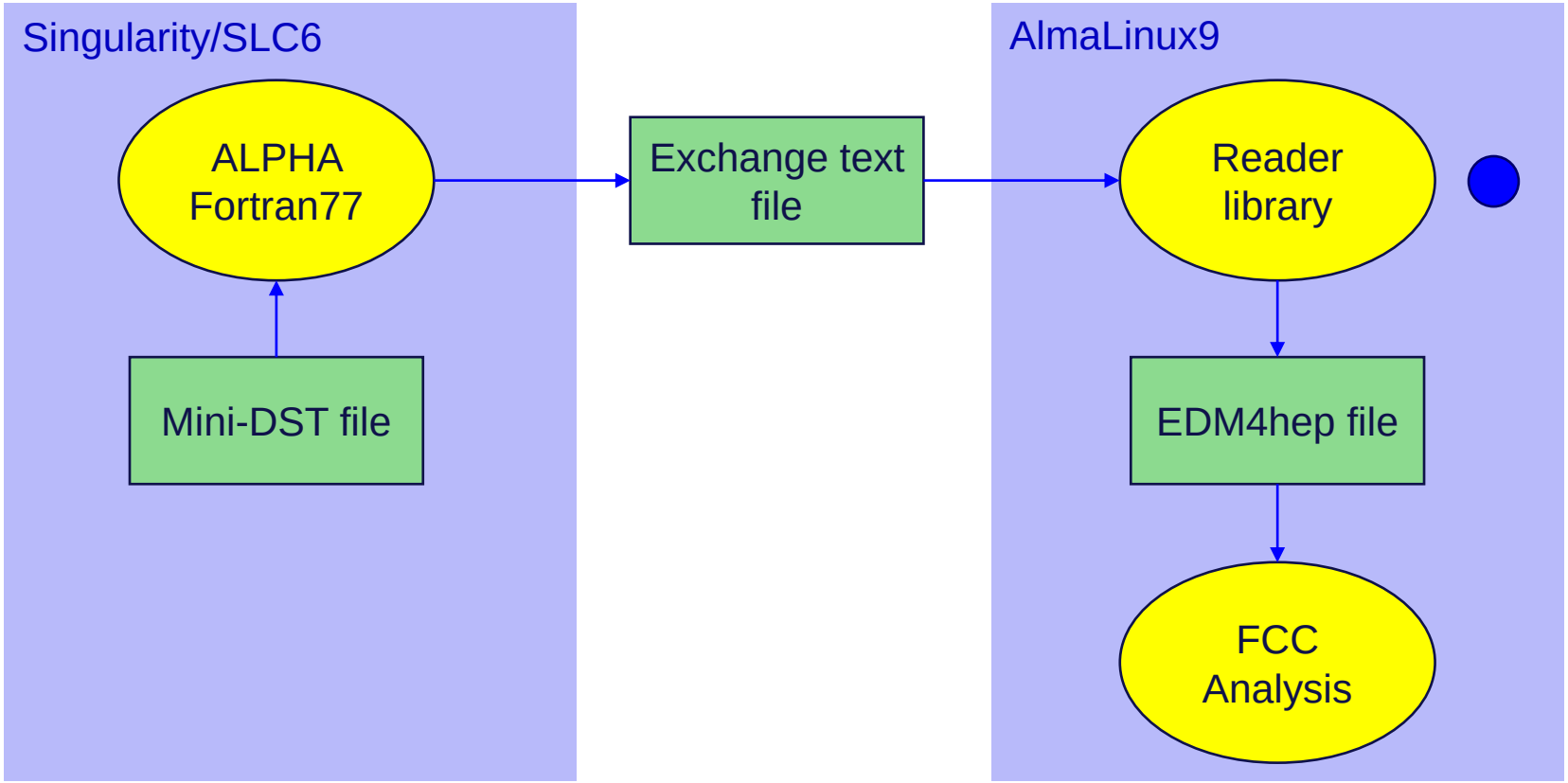
- Data stored in POT/Julia banks are printed in the text exchange format as integers values

- It has been decided to use a simple **text file**:

+
  - Easy to produce, directly by Fortran routines
  - Easy to read, directly by C++ routines
  - Allows to remove a layer of code (with respect, e.g., to JSON)

−
  - Hand-written code for reading the text file and filling EDM4hep data structures

Workflow


Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

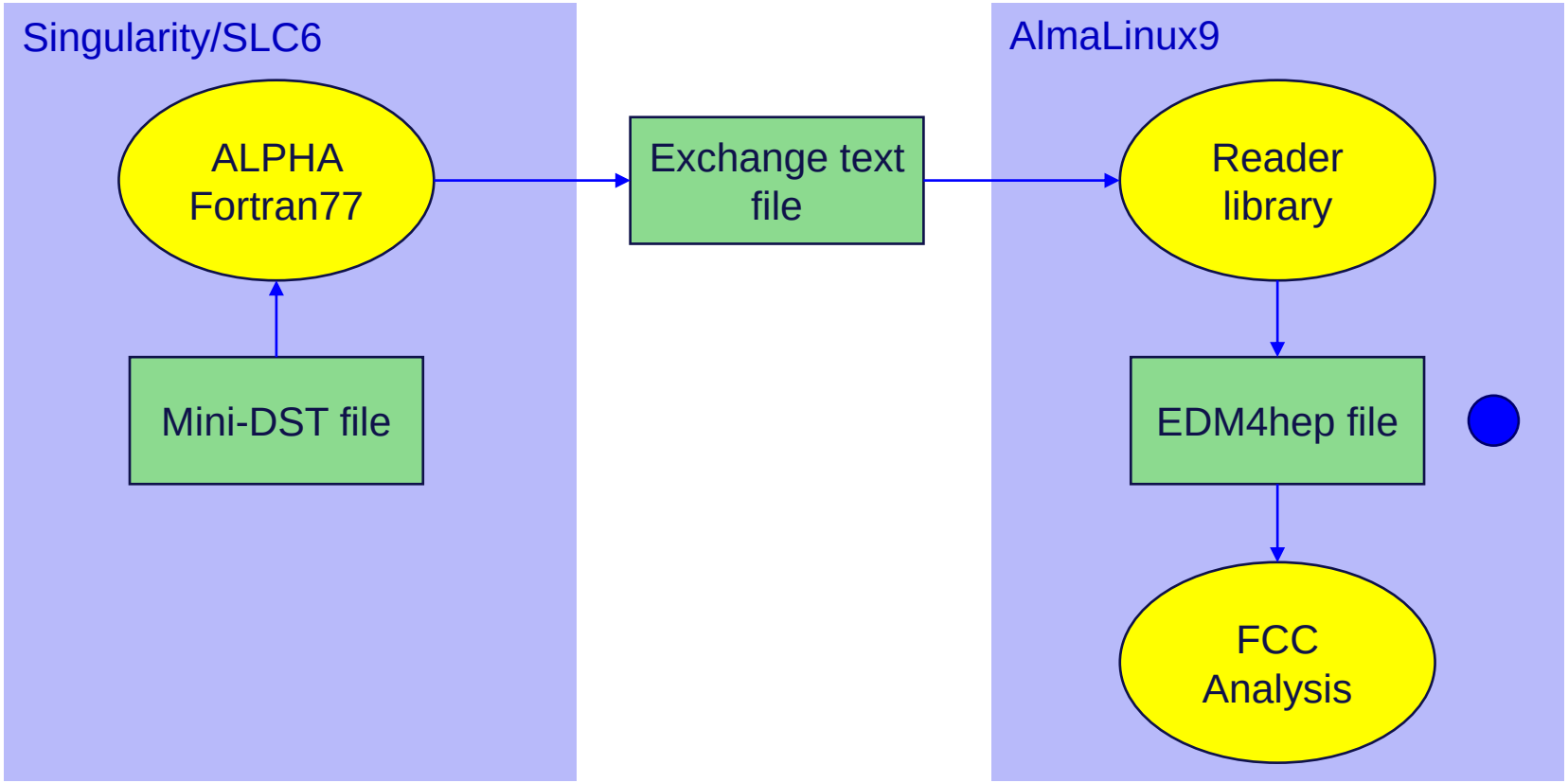
Text reader and EDM4hep writer

- C++ functions and programs for
  - Reading the intermediate exchange text file through a dedicated library
  - Converting the integer values in floats (IEEE-754) or strings
  - Filling some data structures and relations

- Using EDM4hep nightlies, first stable release is expected soon

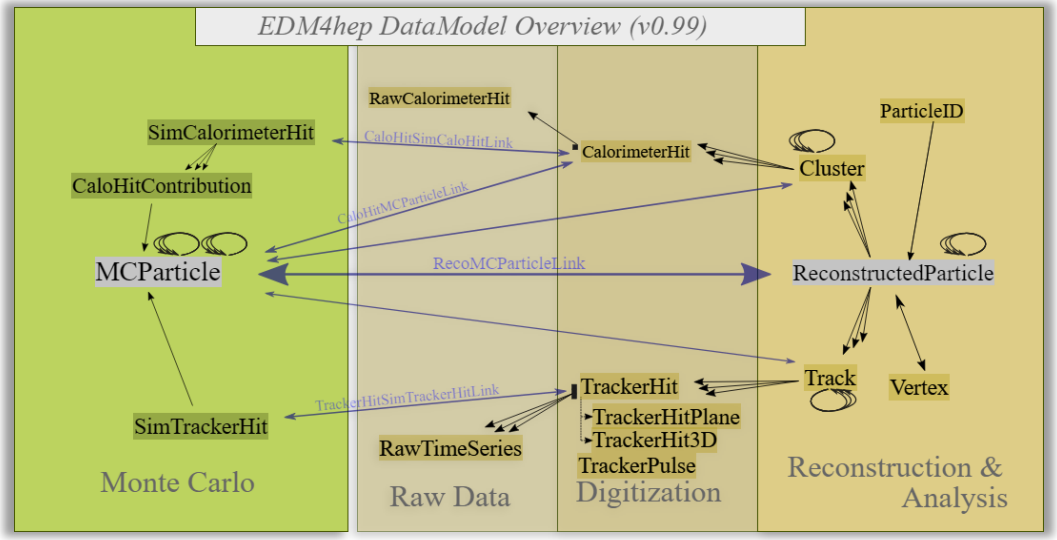
```
// Function to convert an integer to a float with IEE754
float intToFloat(uint32_t intRepresentation) {
    union {
        uint32_t i;
        float f;
    } converter;

    converter.i = intRepresentation;
    return converter.f;
}
```

Workflow


Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

EDM4hep structures

- One-to-one correspondence between a particle from Particle Flow algorithm and an EDM4hep ReconstructedParticle
- Links to Tracks, CaloObjects, Vertexes and V0s
- **Next step:** convert Monte Carlo data, which have also the truth information



- Even with little data, things can be tricky…

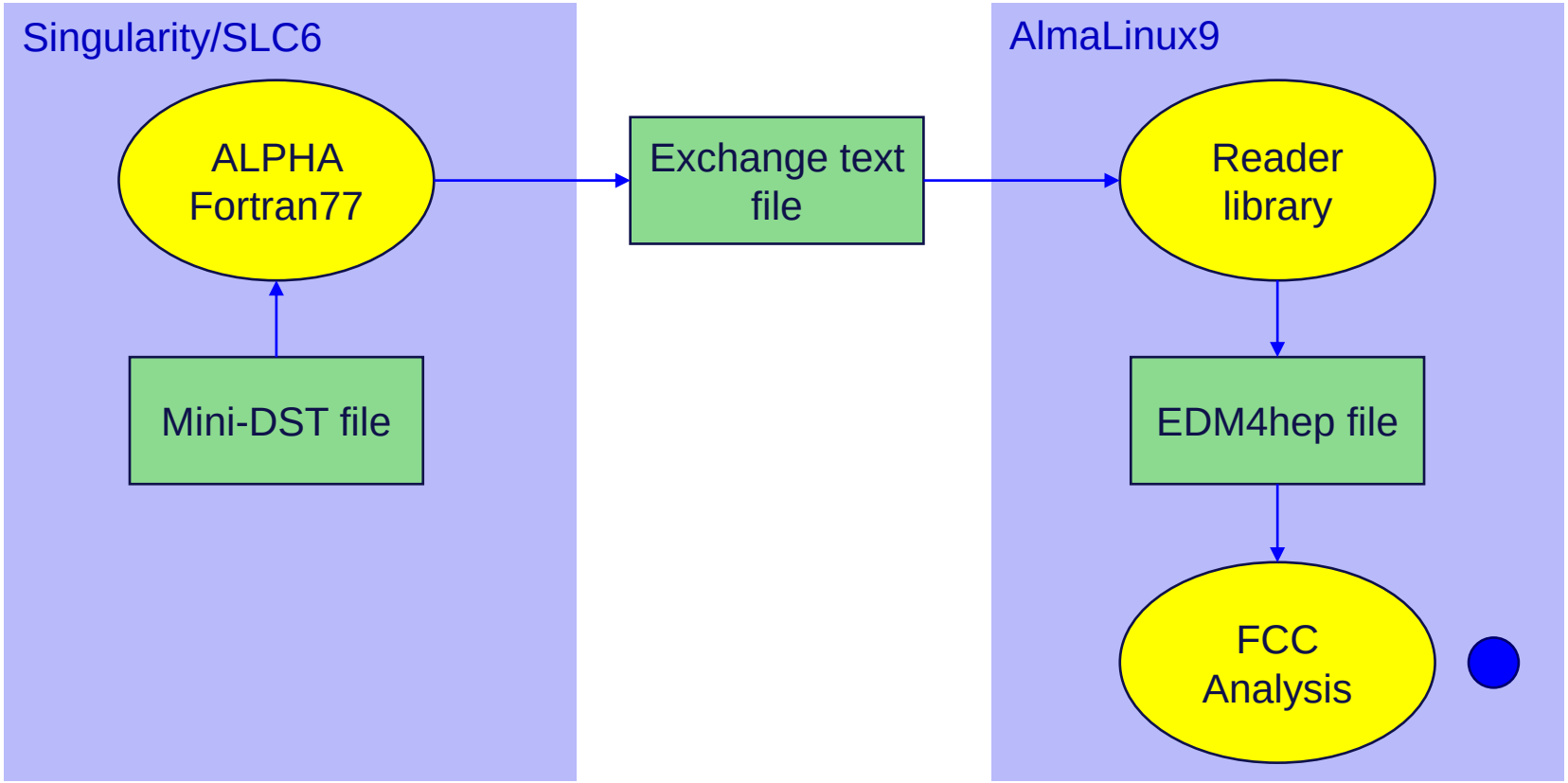
```
                    Subschema: EflowJULPOTBanks
+------+
| EFOL |  Energy FlOw eLements
+------+

  1      I  Number of words/element (=11)
  2      I  Number of elements

  1   PX F   PX            [-999.,999.]
         Weighted component x
  2   PY F   PY            [-999.,999.]
         Weighted component y
  3   PZ F   PZ            [-999.,999.]
         Weighted component z
  4   EW F   EnergyW       [-999.,999.]
         Element weighted by E-Flow coefficients
  5   WE F   WEight        [0.0,99.]
         Weight applied to E-Flow element
  6   TY I   TYpe          [0,10]
         Object type
         0 = Track
         1 = Electron
         2 = Muon
         3 = Track from V0
         4 = Electromagnetic
         5 = Ecal hadron/residu
         6 = Hcal element
         7 = Lcal element
  7   LE I   LinkEcal      [0,999]
         Peco # associated
  8   LT I   LinkTrak      [0,999]
         Track # associated
  9   LH I   LinkHcal      [0,999]
         Phco # associated
  10  LC I   LinkCalo      [0,999]
         Calobject # associated
  11  LJ I   LinkJet       [0,100]
         Jet # associated
```

```
#------- ReconstructedParticle
edm4hep::ReconstructedParticle:
  Description: "Reconstructed Particle"
  Author: "EDM4hep authors"
  Members:
    - int32_t                PDG             // PDG of
    - float                  energy [GeV]    // energy
    - edm4hep::Vector3f      momentum [GeV]  //  parti
    - edm4hep::Vector3f      referencePoint [mm] // re
    - float                  charge          // charge
    - float                  mass  [GeV]     //  mass c
    - float                  goodnessOfPID   // overall
    - edm4hep::CovMatrix4f   covMatrix       // covaria
```

```
   # Parametrized description of a particle tra
  edm4hep::TrackState:
    Members:
      - int32_t location // for use with At{Othe
      - float D0 // transverse impact parameter
      - float phi // azimuthal angle
      - float omega [1/mm] // is the signed curv
      - float Z0 // longitudinal impact paramete
      - float tanLambda // lambda is the dip ang
      - float time [ns] // time of the track at
      - edm4hep::Vector3f referencePoint [mm] //
      - edm4hep::CovMatrix6f covMatrix // covari
```

```
Subschema: JULPOTFitTrack
+------+
| FRFT |  Global Geometrical track FiT
+------+  NR=0.(JUL)

  1      I  Number of words/track (=30)
  2      I  Number of tracks

  1   IR F   InverseRadi     [*,*]
         Inverse radius of curvature in x-y
         projection
         signed +ve if track bends
         counterclockwise
         -ve if tracks bends clockwise
  2   TL F   TanLambda       [*,*]
         tangent of dip angle
  3   P0 F   Phi0            [0.0,6.3]
         Phi at closest point of approach to
         line x=y=0
  4   D0 F   D0              [-180.,180.]
         Closest distance of approach to line
         x=y=0
         in the x-y plane (impact parameter)
         (signed +ve if particle has a positive
         angular
         momemtum around the origin, -ve
         otherwise)
  5   Z0 F   Z0              [-220.,220.]
         Z-coordinate at D0
```

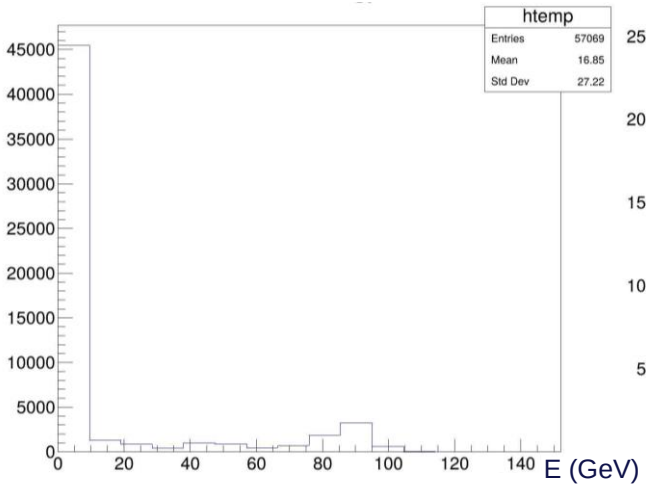
Workflow

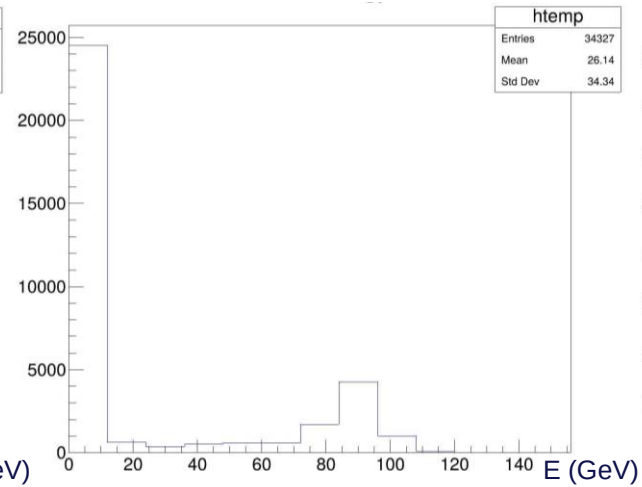

Singularity/SLC6
ALPHA
Fortran77
Mini-DST file
Exchange text
file
AlmaLinux9
Reader
library
EDM4hep file
FCC
Analysis

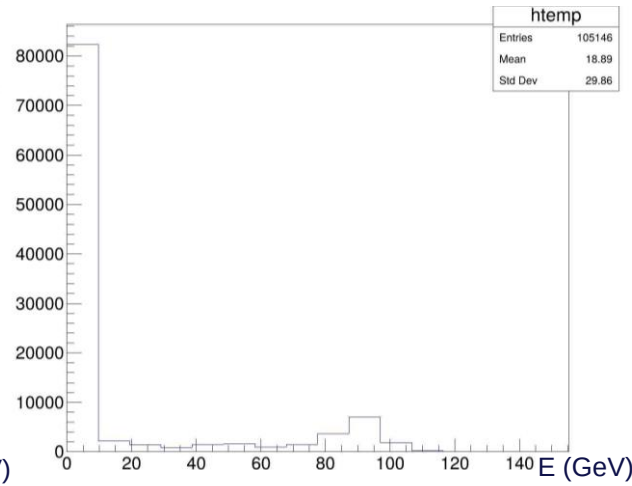
FCCAnalysis

- **Common framework** for FCC related analyses, taking EDM4hep input ROOT files and producing histograms
- Based on **ROOT RDataFrame** for the construction of the computational graph
- Actions are lazy evaluated
- Some analysis routines are pre-defined, users can define their own directly as JIT compiled C++ functions/functors
- It is still under development to meet the needs of the FCC community

Looking at some data: E_{TOT}







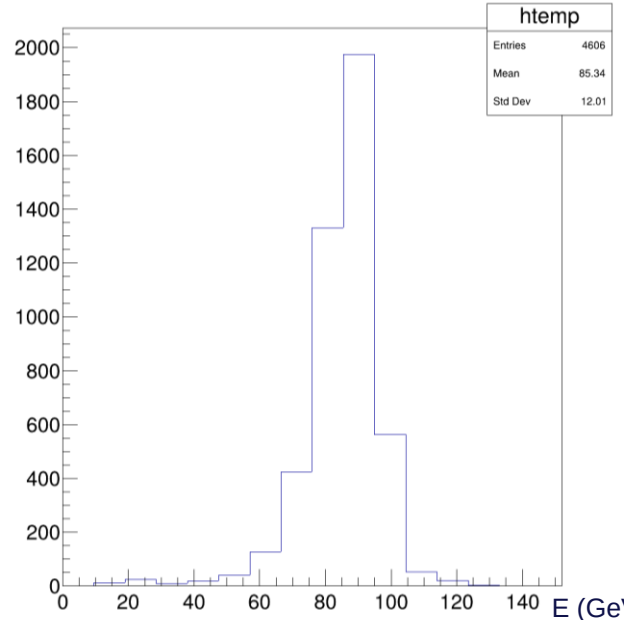
- Subset of data:
  - y = 1995 (LEP1)
  - Run: 37202-37211
  - E ~ 89 GeV
  - # ev. = 57069

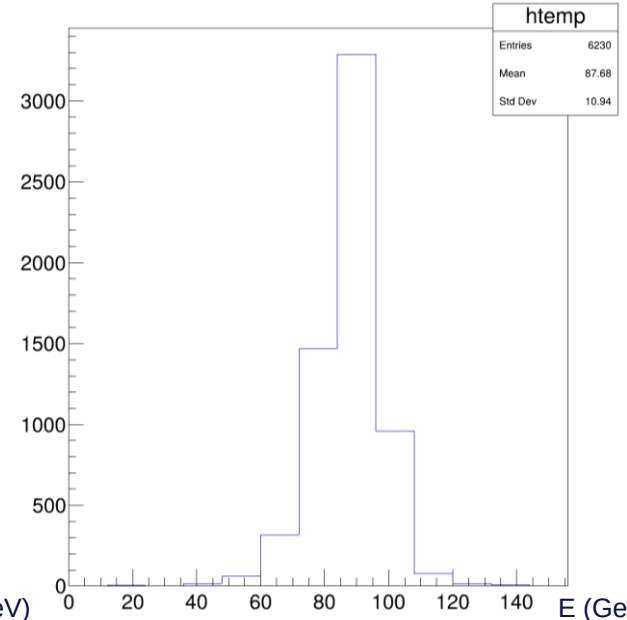
- Subset of data:
  - y = 1995 (LEP1)
  - Run: 37223-37230, 37272-37275
  - E ~ 91 GeV
  - # ev. = 34327

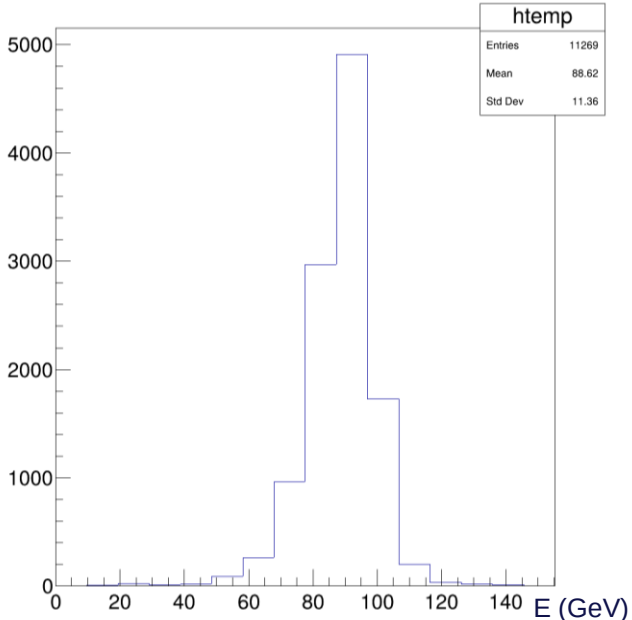
- Subset of data:
  - y = 1995 (LEP1)
  - Run: 37196-37199, 37217, 37218, 37245-37218
  - E ~ 93 GeV
  - # ev. = 105146

Simple cuts

- Cuts: (CLAS 16, “**hadronic events**”)
  - At least 5 TPC tracks satisfying:
    - |D0| < 2 cm
    - |Z0| < 10 cm
    - TPC coordinates ≥ 4
    - |cos(Θ)| < 0.95
  - Total energy of TPC tracks satisfying the above conditions > 0.1 E_{CM}







Quick exercise: $\sigma_{hadr.}$ estimation

- The cross section of the reaction

$$e^+e^- \rightarrow hadr.$$

can be as a first approximation evaluated as

$$\sigma = \frac{\#\, ev.}{\mathcal{L}}$$

- E ~ 89 GeV
  - #ev. = 4606
  - $\mathcal{L}$ ~ 471.6 nb^{-1}

$\boldsymbol{\sigma = (9.8 \pm 0.1)\ \mathbf{nb}^*}$

- E ~ 91 GeV
  - #ev. = 6230
  - $\mathcal{L}$ ~ 204.2 nb^{-1}

$\boldsymbol{\sigma = (30.5 \pm 0.4)\ \mathbf{nb}^*}$

- E ~ 93 GeV
  - # ev. = 11269
  - $\mathcal{L}$ ~ 781.8 nb^{-1}

$\boldsymbol{\sigma = (14.4 \pm 0.1)\ \mathbf{nb}^*}$

* the error is the statistical error

Data sizes

- Taking as example the file /eos/experiment/aleph/LEP1/DATA/MINI/1995/Y15223.44.AL
  - Original file: 314 MB
  - Text file: 1.6 GB
  - Zipped text file: 272 MB
  - EDM4hep ROOT file: 155 MB

- Caveat: EDM4hep file contains only a small part of the data dumped in the text file
  - Further data size evaluation needed

Validation

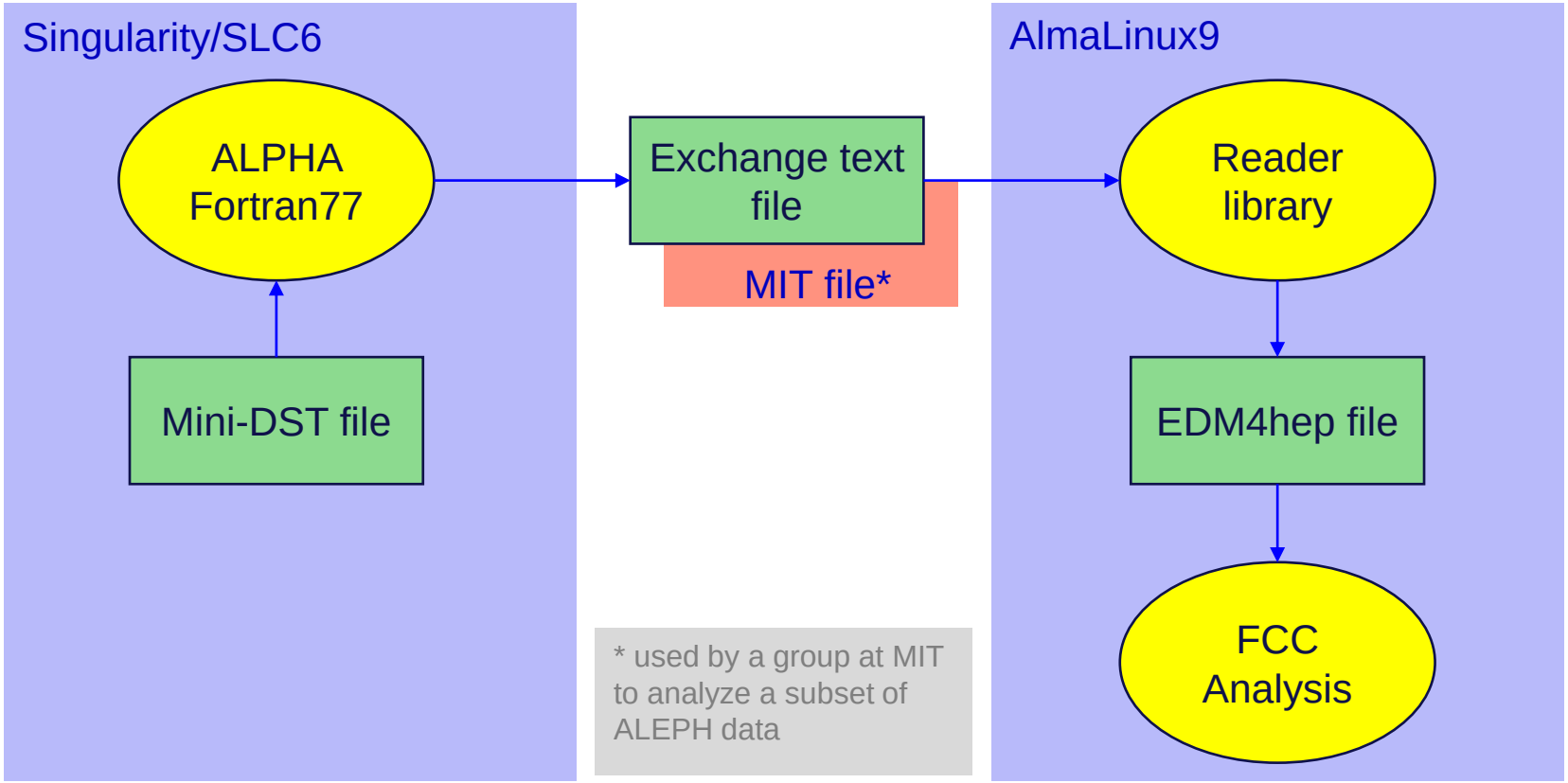
Validation plans

- Target
  - Detailed comparison of old/new formats for standard selections (CLAS 16, …)
  - Perform the same ALEPH analysis via FCCAnalysis (or ROOT) and compare the results with the same analysis performed in the past using Fortran-only routines

- In the meantime
  - A group from MIT analyzed a subset of ALEPH data, thanks to M. Maggi
  - Ongoing collaboration for a validation using their results

See also:
- Yi Chen et al., First measurement of anti-kT jet spectra and jet substructure using the archived ALEPH e+e- data at 91.2 GeV, 41st ICHEP
- Yi Chen, Revisiting the ALEPH Archived e^+e^- Data, CERN EP Seminar

Workflow - MIT



The “MIT format”

```
ALEPH_DATA RUN = 35482 EVENT    15 ECM =   91.650 GEV
 Primary vertex info flag =  4 vx = -0.0802 vy =  0.0308 ex =  0.0019 ey =  0.0000
px=   -0.375 py=   -0.045 pz=    0.035 m=    0.140 charge=  1.0 pwflag=  0 lock= 1 d0=   -0.725 z0=    1.155 ntpc= 16 nitc=  0 nvdet= 1 track=   1 de/dx code=0 (e-)  -6.56 (pi-)   0.45 (K-) -11.91 (p) -27.42
px=   -0.264 py=   -0.026 pz=    0.018 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=   -0.047 z0=    1.373 ntpc= 11 nitc=  2 nvdet= 2 track=   2 de/dx code=0 (e-)  -2.65 (pi-)   0.56 (K-) -10.64 (p) -24.37
px=    6.591 py=    1.108 pz=    0.591 m=    0.140 charge=  1.0 pwflag=  0 lock= 1 d0=   -0.009 z0=    1.338 ntpc= 17 nitc=  2 nvdet= 2 track=   3 de/dx code=0 (e-)  -3.14 (pi-)  -0.35 (K-)   2.04 (p)   3.41
px=   30.342 py=    4.278 pz=    1.145 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=   -0.006 z0=    1.337 ntpc= 15 nitc=  0 nvdet= 2 track=   4 de/dx code=0 (e-)  -2.00 (pi-)  -0.71 (K-)   0.68 (p)   1.75
px=   -7.908 py=   -1.061 pz=   -0.332 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=    0.009 z0=    1.331 ntpc= 21 nitc=  0 nvdet= 2 track=   5 de/dx code=0 (e-)  -3.45 (pi-)  -0.35 (K-)   2.26 (p)   3.85
px=   -2.927 py=   -0.017 pz=   -0.687 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=    0.004 z0=    1.343 ntpc= 18 nitc=  3 nvdet= 2 track=   6 de/dx code=0 (e-)  -2.84 (pi-)   0.40 (K-)   2.42 (p)   2.89
px=   -1.499 py=   -0.338 pz=    0.108 m=    0.140 charge=  1.0 pwflag=  0 lock= 1 d0=    0.424 z0=    0.932 ntpc= 20 nitc=  4 nvdet= 0 track=   7 de/dx code=0 (e-)  -5.40 (pi-)  -0.50 (K-)   0.95 (p)  -0.99
px=    1.498 py=    0.681 pz=    0.439 m=    0.140 charge=  1.0 pwflag=  0 lock= 1 d0=   -0.011 z0=    1.323 ntpc= 17 nitc=  2 nvdet= 2 track=   8 de/dx code=0 (e-)  -5.48 (pi-)   0.08 (K-)   1.94 (p)   0.09
px=   -3.652 py=   -0.185 pz=   -0.575 m=    0.140 charge=  1.0 pwflag=  0 lock= 1 d0=   -0.162 z0=    0.576 ntpc= 11 nitc=  2 nvdet= 0 track=   9 de/dx code=0 (e-)  -6.14 (pi-)  -3.43 (K-)  -1.60 (p)  -1.00
px=   -0.960 py=    0.049 pz=   -0.215 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=    0.008 z0=    1.325 ntpc= 14 nitc=  0 nvdet= 1 track=  11 de/dx code=0 (e-)  -4.47 (pi-)  -0.34 (K-)  -0.45 (p)  -4.05
px=    0.418 py=    0.139 pz=    0.306 m=    0.140 charge= -1.0 pwflag=  0 lock= 1 d0=   -0.193 z0=    1.345 ntpc= 15 nitc=  2 nvdet= 2 track=  13 de/dx code=0 (e-)  -6.66 (pi-)   1.31 (K-)  -6.34 (p) -19.28
px=    1.857 py=    0.245 pz=    0.030 m=    0.000 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=    0.822 py=    0.140 pz=    0.069 m=    0.000 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=    1.333 py=    0.117 pz=    0.260 m=    0.000 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=    0.959 py=    0.203 pz=    0.198 m=    0.000 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=    1.350 py=    0.585 pz=   -0.109 m=    0.000 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=   -2.373 py=   -0.260 pz=    0.081 m=    0.022 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=   -3.243 py=   -0.473 pz=    0.049 m=    0.001 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=   -2.128 py=    0.011 pz=   -0.584 m=    0.021 charge=  0.0 pwflag=  4 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
px=   -9.851 py=   -1.656 pz=   -0.410 m=    1.269 charge=  0.0 pwflag=  5 lock= 1 d0=   -1.000 z0=   -1.000 ntpc=  0 nitc=  0 nvdet= 0 track=   0 de/dx code=1 (e-)  -1.00 (pi-)  -1.00 (K-)  -1.00 (p)  -1.00
vx=    -7.49 vy=    -1.23 vz=     1.85 chi2 =    0.000 type=0 Ntrk=  2
 Track=     1 px=  -0.377 py=  -0.011 pz=   0.037
 Track=     2 px=  -0.259 py=  -0.059 pz=   0.013
vx=    -0.11 vy=     0.03 vz=     1.34 chi2 =    0.000 type=0 Ntrk=  2
 Track=     3 px=   6.585 py=   1.108 pz=   0.590
 Track=     4 px=  30.165 py=   4.248 pz=   1.137
vx=    -6.15 vy=    -0.76 vz=     1.79 chi2 =    0.000 type=0 Ntrk=  2
 Track=     7 px=  -1.505 py=  -0.311 pz=   0.108
 Track=     2 px=  -0.260 py=  -0.054 pz=   0.018
vx=    -5.00 vy=    -0.63 vz=     1.12 chi2 =    0.000 type=0 Ntrk=  2
 Track=     7 px=  -1.505 py=  -0.314 pz=   0.113
 Track=     5 px=  -7.907 py=  -1.084 pz=  -0.332
vx=    -1.95 vy=     0.02 vz=     0.90 chi2 =    0.000 type=0 Ntrk=  2
 Track=     7 px=  -1.502 py=  -0.327 pz=   0.114
 Track=     6 px=  -2.927 py=  -0.026 pz=  -0.687
vx=    -0.09 vy=     0.04 vz=     1.32 chi2 =    0.000 type=0 Ntrk=  2
 Track=     8 px=   1.498 py=   0.681 pz=   0.438
 Track=    13 px=   0.416 py=   0.145 pz=   0.307
primary vertex compatibility track   1 chi= -999.00 track   2 chi= -999.00
primary vertex compatibility track   3 chi= -999.00 track   4 chi= -999.00
primary vertex compatibility track   7 chi= -999.00 track   2 chi= -999.00
primary vertex compatibility track   7 chi= -999.00 track   5 chi= -999.00
primary vertex compatibility track   7 chi= -999.00 track   6 chi= -999.00
primary vertex compatibility track   8 chi= -999.00 track  13 chi= -999.00
END_EVENT
```

- «Hadronic» events (≥ 5 charged particles)
- High level informations: reconstructed particles from particle flow, V0s, dE/dX

Chain validation w/ MIT

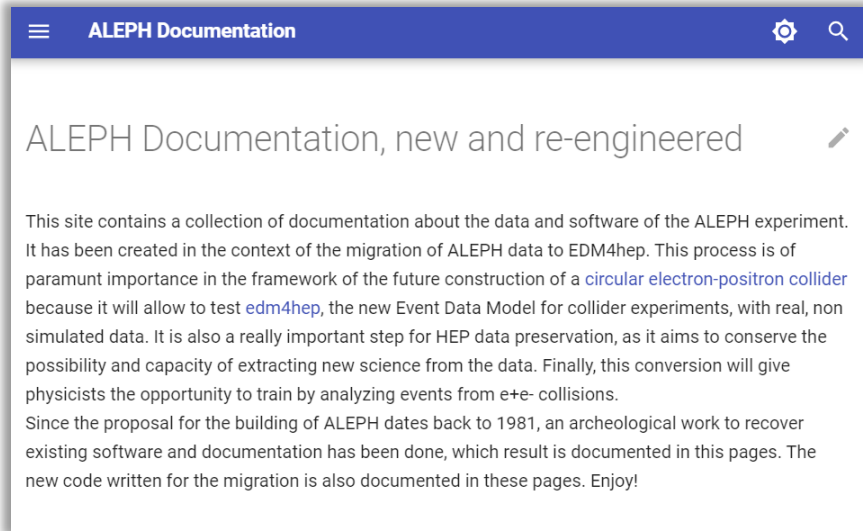
- Since ROOT is the most widely diffused analysis framework in HEP, it is convenient to make the EDM4hep files easily analyzable with it
- A tutorial with an example to extract a flat ROOT TTree from the EDM4hep file using EDM4hep Utilities has been produced
  - For the use of MIT colleagues
- Users can define their functions to store in the TTree the relevant quantities for validation/analysis
- Waiting for their feedback

Summary & Outlook

Summarizing: what has been done

- Found a way to access directly low level data (ALPHA)
- Defined a general exchange format for data extraction (text file)
- Filled EDM4hep structures and relations
- Provided MIT colleagues with tools to do initial validation of the conversion chain
- Collect archeological information and tutorials on a website (beta version)

Set up a new chain of programs to convert the original ALEPH files (Mini-DST) to EDM4hep files, which can be analyzed with FCCAnalyses



Work ahead

- Prepare for processing **Monte Carlo data**, including truth information
- Understand better the conventions to be used in the **EDM4hep production**
- Migrate **database with meta-data** (fill, run, luminosity, detector status, …) to a modern backend and interface
  - Including new location of files on EOS
- Better quantify the average **size** of the intermediate text file and the EDM4hep output file
- **Validate the workflow** by repeating a full analysis previously done by ALEPH collaboration (and processing MIT feedback)

Thank you for your attention

In depth documentation

- Data Preservation in HEP: paper by DPHEP collaboration on data preservation reasons and strategies
- ALEPH GitLab: source code and some general information about the experiment
- ALEPH website: old public webpage of the ALEPH collaboration
- ALPHA User's Guide: description of ALPHA analysis routines
- EDM4hep GitHub: source code of the general Event Data Model