

ALEPH DATA IN EDM4HEP

Reasons, status, perspectives

Gerardo Ganis, Marcello Maggi, Juraj Smiesko, Jacopo Fanini

Motivations

- **EDM4hep test**
 - The new Event Data Model is used for the first time with real data
- **Training on real data**
 - New generations of physicists will have the opportunity to train by analyzing real e^+e^- events
- **New analysis and optimization of algorithms**
 - Innovative analysis techniques (e. g. machine learning algorithms) can be used and tested on LEP data
- **Data preservation**
 - The possibility and ability of extracting new science from LEP data is conserved
- **Validation of simulations tools**

See also:

M. Maggi, [ALEPH data in Key4HEP](#)

G. Ganis, [Opportunities offered by LEP data@EDM4hep](#)



ALEPH Data Reminder

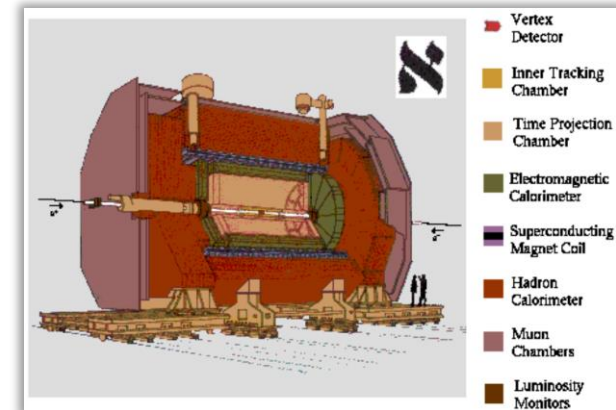
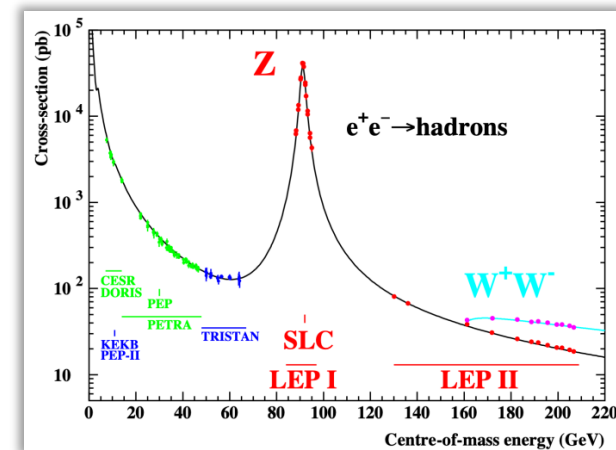
LEP and ALEPH

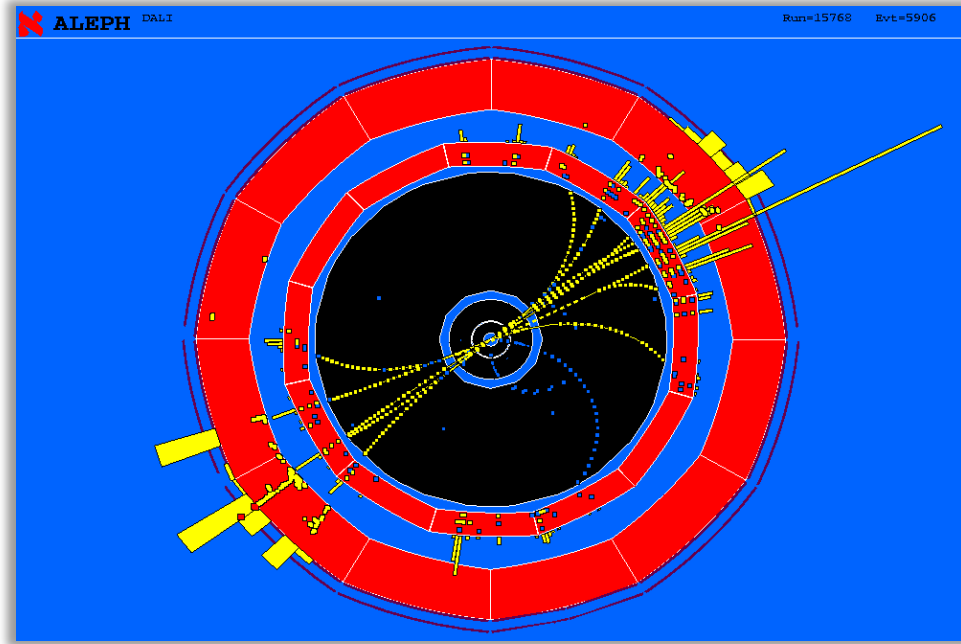
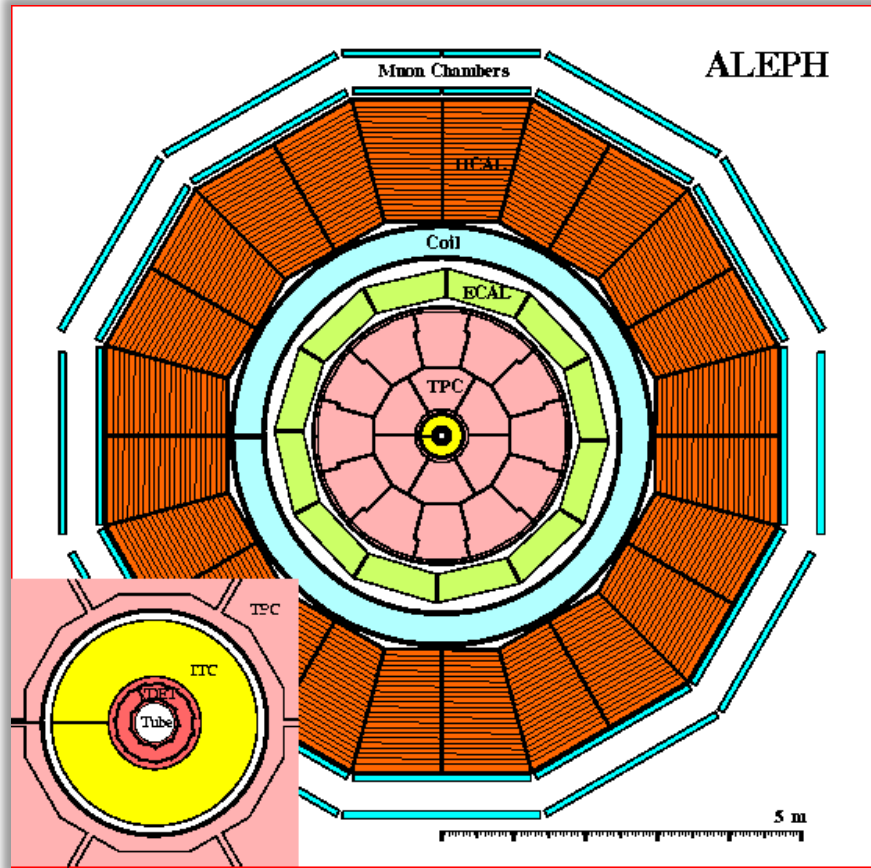
• LEP

- e^+e^- collider
- 1989-1995: Z production @ ~91 GeV
- 1996-2000: W-pair production @ ~160-209 GeV
- 4 experiments: DELPHI, L3, OPAL and...

• ALEPH

- Typical “general purpose” experiment: vertex detector, tracking, solenoid magnet, calorimetry, muon system
- Luminosity
 - LEP1: 200 pb^{-1}
 - LEP2: 688 pb^{-1}
- Statistics
 - $4 \times 10^6 e^+e^- \rightarrow q\bar{q}$
 - $8 \times 10^3 e^+e^- \rightarrow W^+W^-$





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

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`

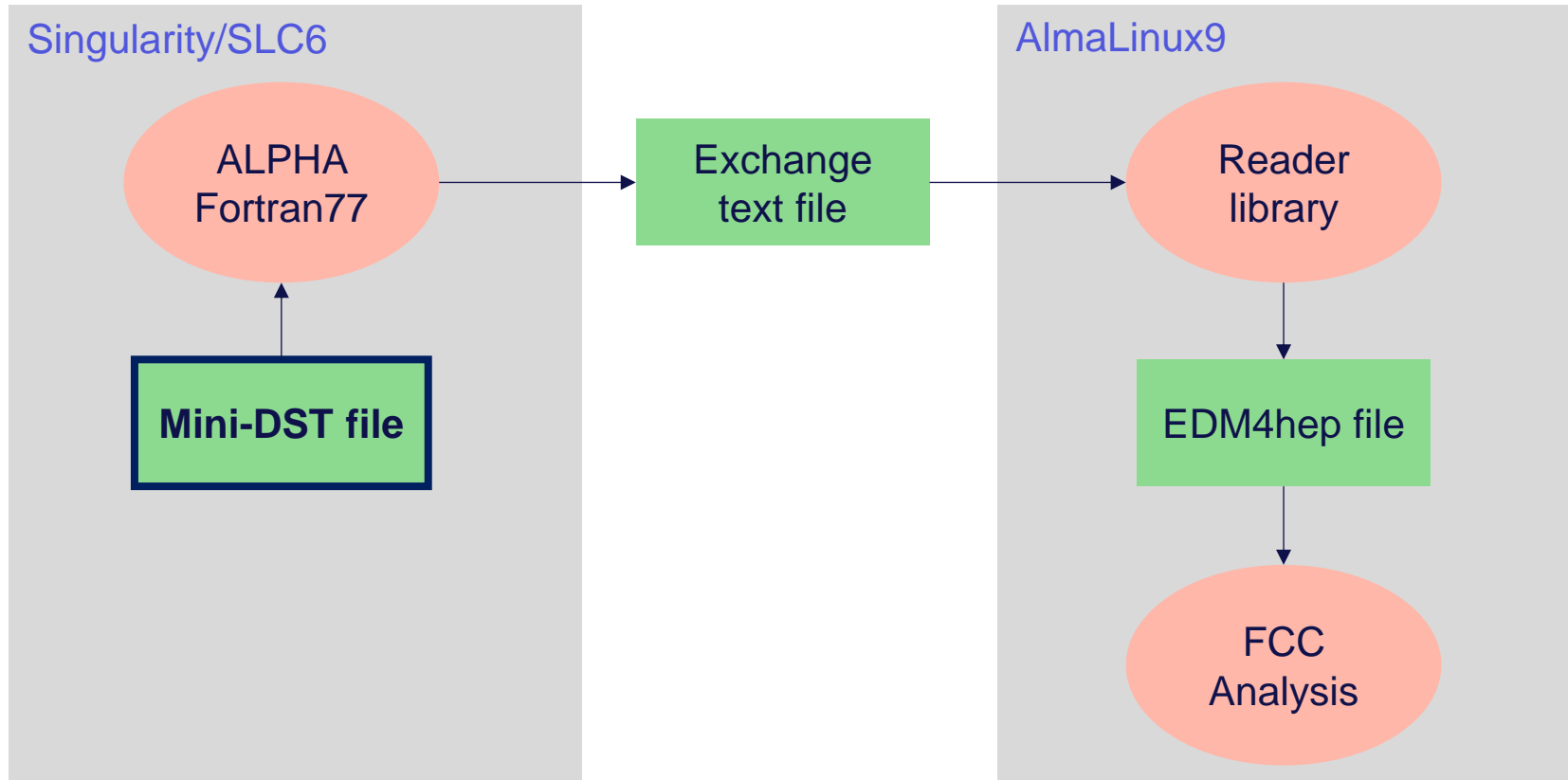
- These environments can still be recreated on today's lxplus (**AlmaLinux9**) via **Apptainer/Singularity + CernVM**

```
[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

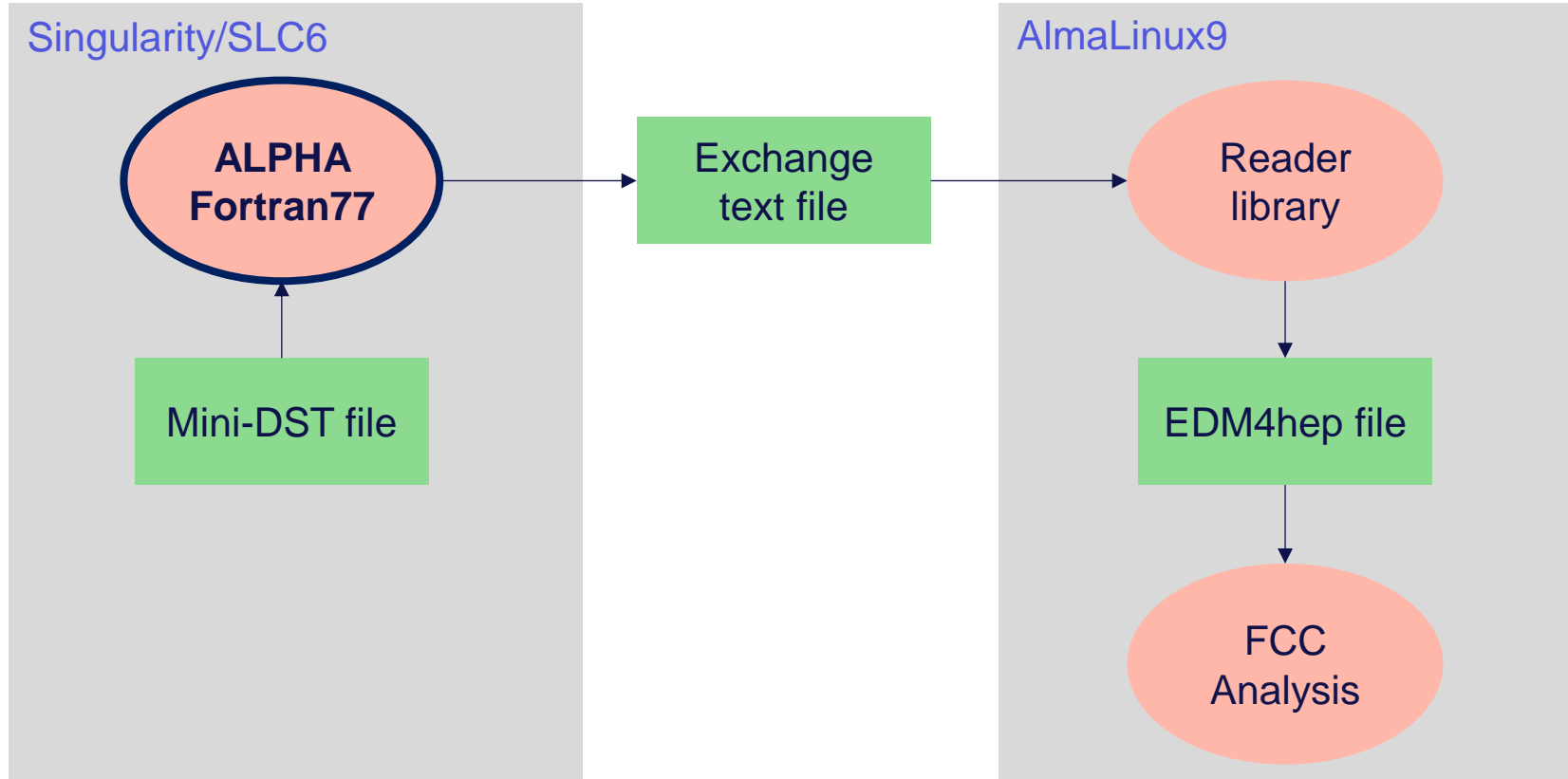


Focus on Mini-DST files

- **Reduced format**
 - Created from DST for space saving
 - One run record per run and at least one event record per run
- **Event records contain most of the data used in analysis**
 - Tracks, vertices, calorimetric objects, energy flow, jets, γ , e , μ identification, HV detector status, trigger
- **Both Mini-DST and DST available on EOS at /eos/experiment/aleph**
 - The access to DST might be a future step



Workflow



ALPHA

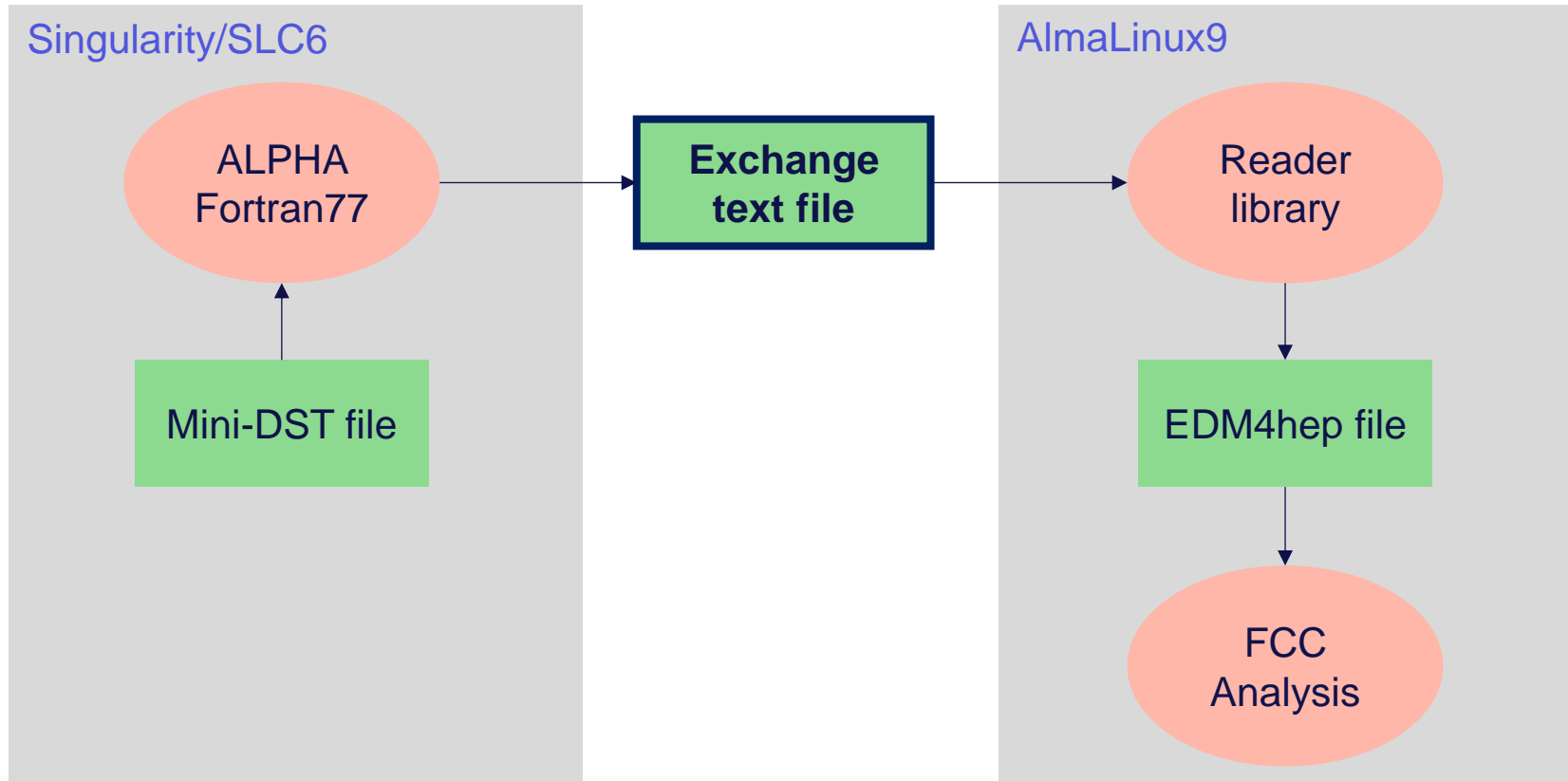
- **It was the ALePh Physics Analysis tool**
 - Used to loop over the ALEPH Mini-DST banks and print them out directly to the exchange file
- **Some technicalities in the process**
 - Two types of banks: linear and tabular
 - In Mini-DST data are stored scaled and integerised, then ALPHA use them to 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

```

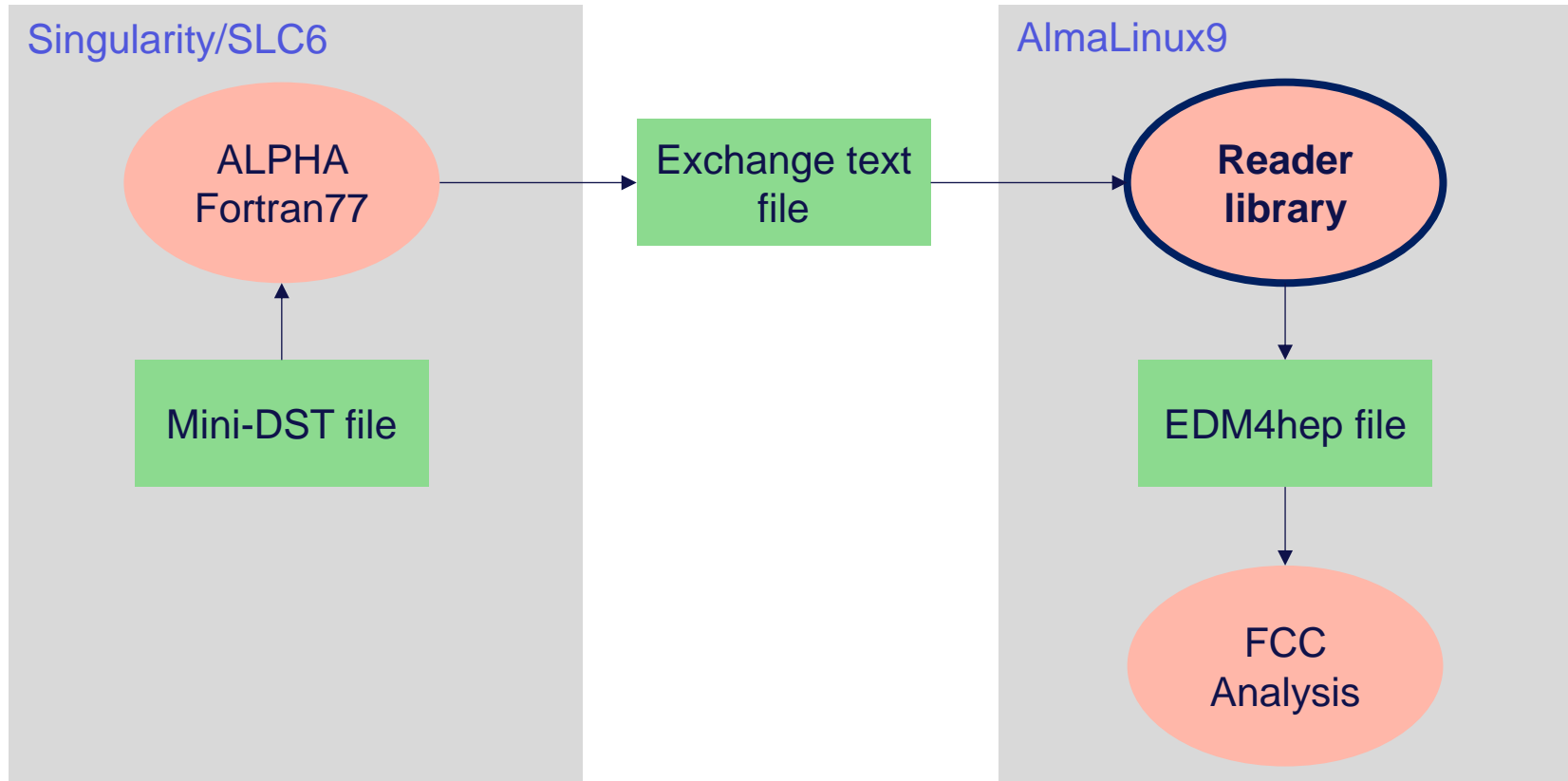
Workflow



Text exchange format

- **Data stored in POT/Julia banks are printed in the text exchange format as integers values**
 - The conversion from integer to float is performed by libraries in the reader program
- **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



Text reader and EDM4hep writer

- **Set of C++ functions and programs to obtain an EDM4hep file**
 - Read the intermediate exchange text file through a dedicated library
 - Convert the integer values in floats (IEEE-754) or strings
 - Fill some EDM4hep 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;
}
```


Even with little data, things can be tricky...

```

Subschema: EflowJULPOTBanks
+-----+
| EFOL | Energy FIOw elements
+-----+
.....
1  | Number of words/element (=11)
2  | 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 o
- float       goodnessOfPID // overall
- edm4hep::CovMatrix4f covMatrix // covaria

```

```

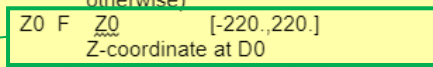
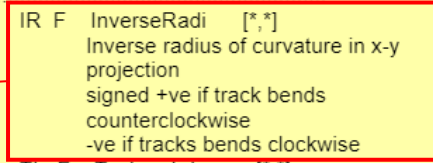
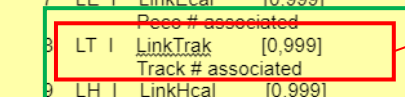
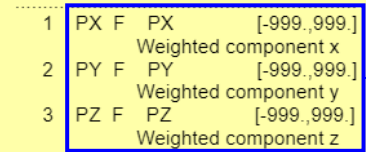
# Parametrized description of a particle track
edm4hep::TrackState:
Members:
- int32_t location // for use with At{0th
- float D0 // transverse impact parameter
- float phi // azimuthal angle
- float omega [1/mm] // is the signed cur
- float Z0 // longitudinal impact paramet
- float tanLambda // lambda is the dip an
- float time [ns] // time of the track at
- edm4hep::Vector3f referencePoint [mm] //
- edm4hep::CovMatrix6f covMatrix // covar

```

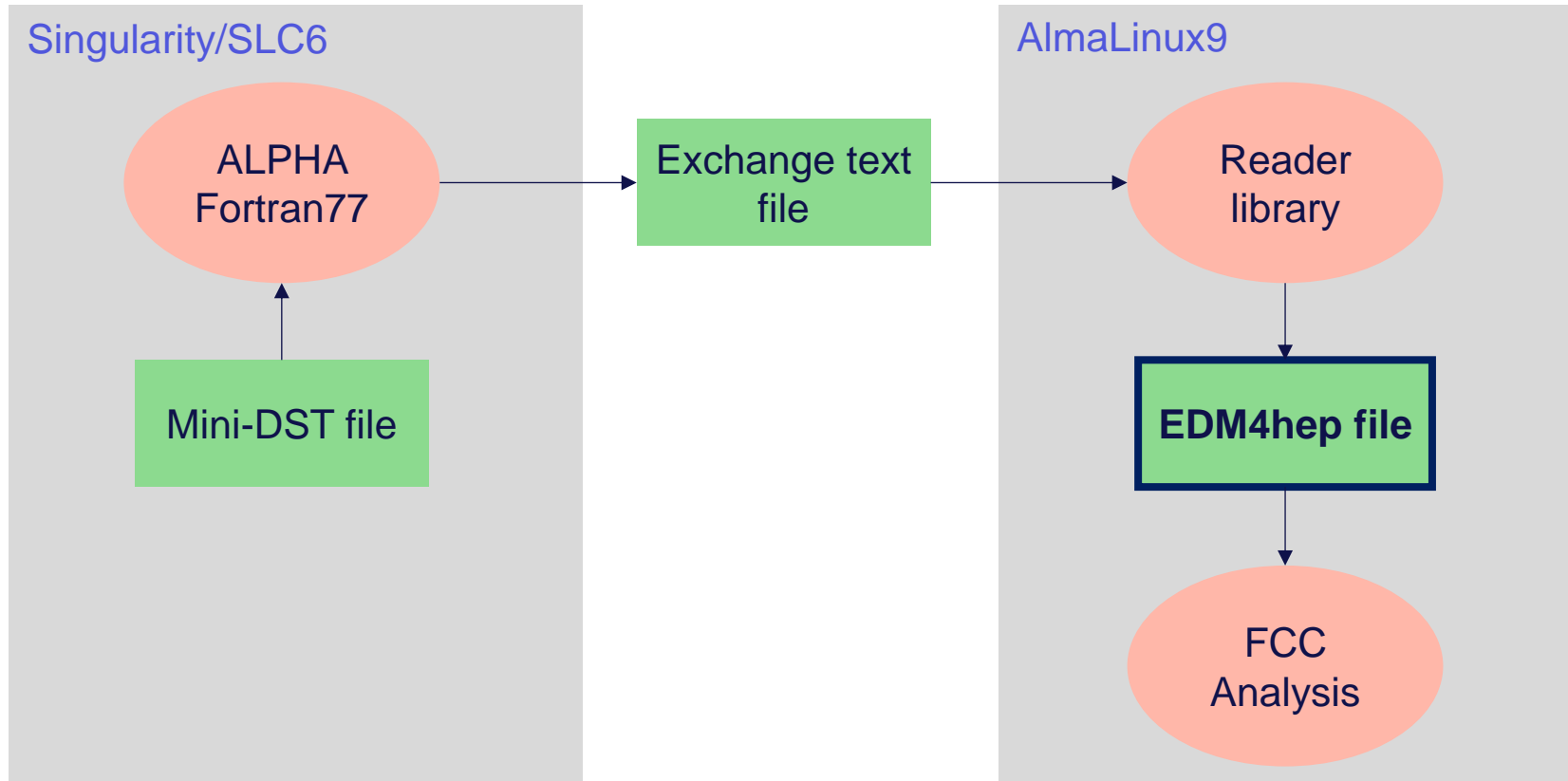
```

Subschema: JULPOTFitTrack
+-----+
| FRFT | Global Geometrical track FIT
+-----+ NR=0.(JUL)
.....
1  | Number of words/track (=30)
2  | 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
   | momentum around the origin, -ve
   | otherwise)
5  Z0 F  Z0         [-220.,220.]
   | Z-coordinate at D0

```

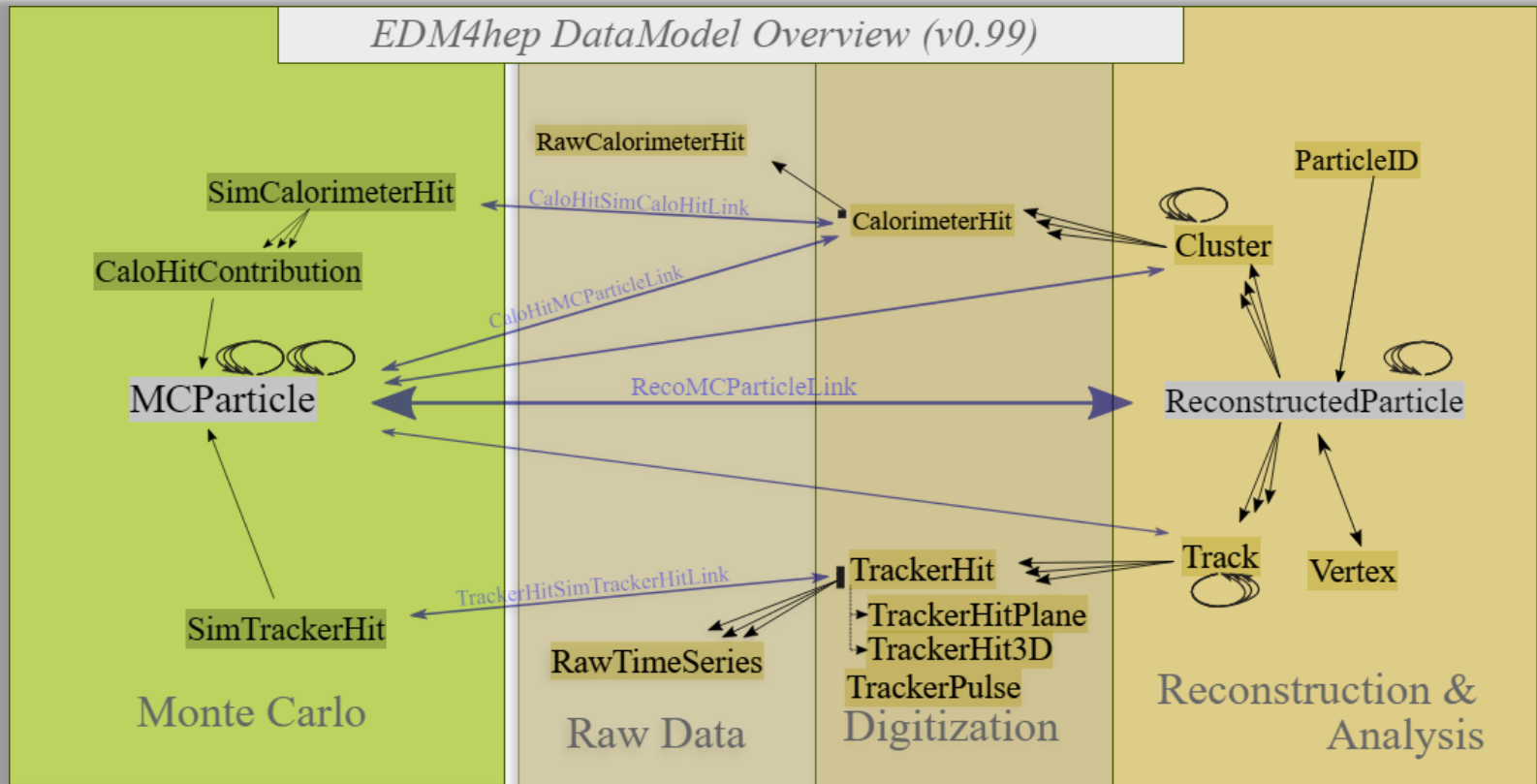


Workflow

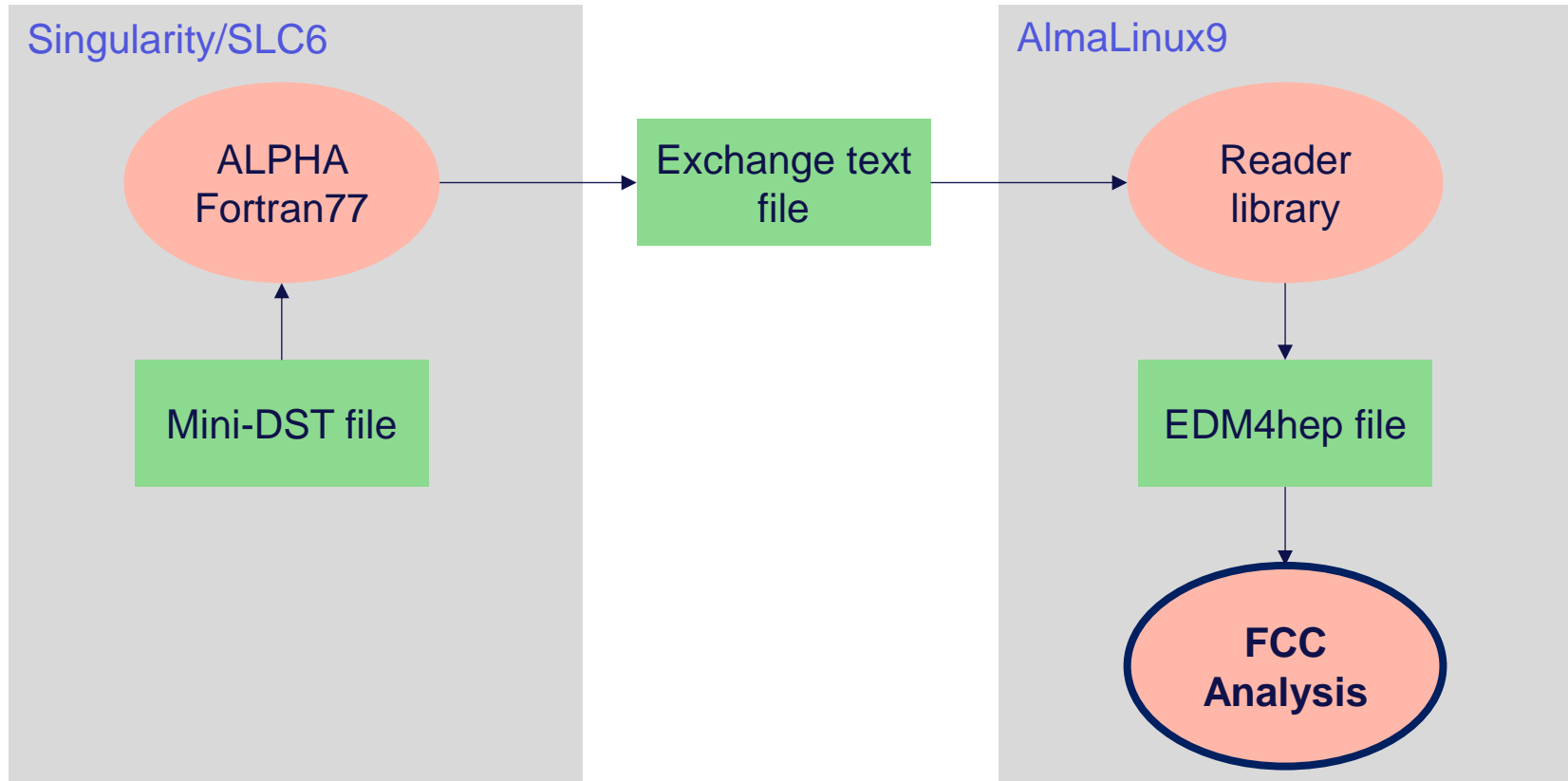


EDM4hep structures

- **It is part of the Key4hep framework**
 - Might become a general and standard data format for data structures and file format of future experiments
 - More details in Juan's and Andre's talks
- **One-to-one correspondence between a particle from particle flow algorithm and an EDM4hep ReconstructedParticle**
 - Conversion goal: energy flow, vertexes w/ covariances, tracks w/ covariances, calorimetric objects, ...
- **It is possible to store data that do not have a correspondent place in the EDM using user-defined collections and extensions**
 - This makes the EDM rigorous but allows flexibility when needed



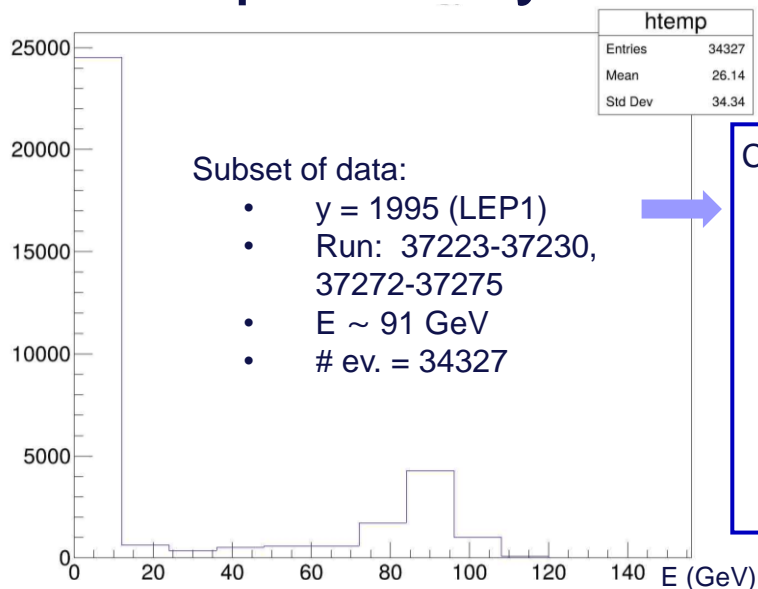
Workflow



FCCAnalysis

- **Common framework for FCC-related analyses**
 - Taking EDM4hep ROOT files as input 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**
 - More details in [Juraj's talk](#)

Simple analysis: CLASS 16 E_{TOT}

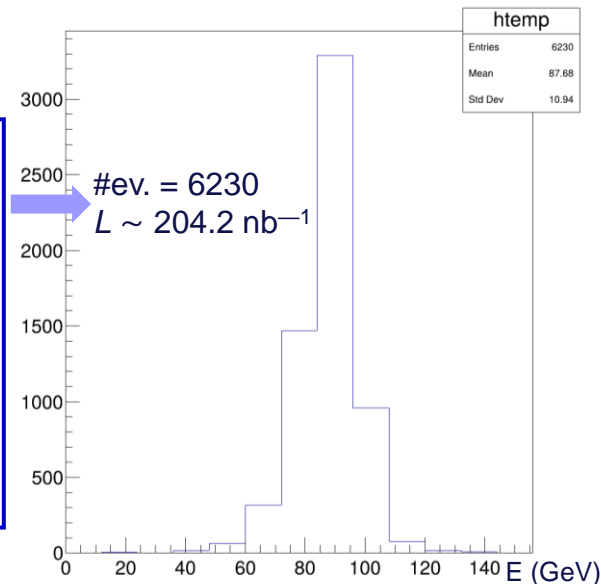


Subset of data:

- $y = 1995$ (LEP1)
- Run: 37223-37230, 37272-37275
- $E \sim 91$ GeV
- # ev. = 34327

Cuts: (CLASS 16, "hadronic events")

- At least 5 TPC tracks with:
 - $|D0| < 2$ cm
 - $|Z0| < 10$ cm
 - TPC coordinates ≥ 4
 - $|\cos(\Theta)| < 0.95$
- Total energy of TPC tracks satisfying the above conditions $> 0.1 E_{CM}$



#ev. = 6230

$L \sim 204.2 \text{ nb}^{-1}$

Same approach has been performed with subsets at $E \sim 89$ GeV and $E \sim 93$ GeV. This allow to estimate the hadronic cross section as

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

$$\sigma_{had@89GeV} = (9.8 \pm 0.1) \text{ nb}^*$$

$$\sigma_{had@91GeV} = (30.5 \pm 0.4) \text{ nb}^*$$

$$\sigma_{had@93GeV} = (14.4 \pm 0.1) \text{ nb}^*$$

* 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
- **EDM4hep file contains only a small part of the data dumped in the text file**
 - Further data size evaluation and conversion optimization are foreseen

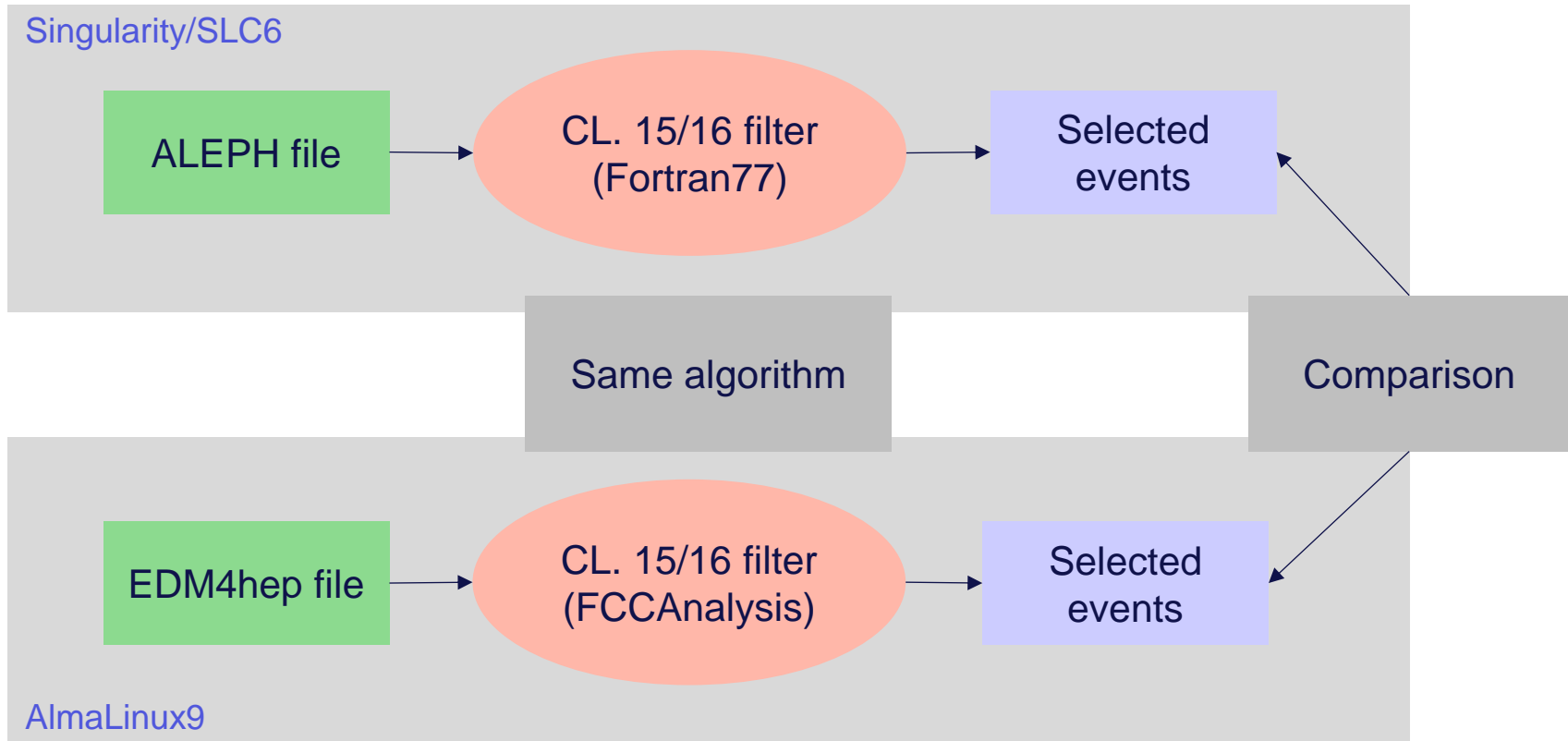


Validation

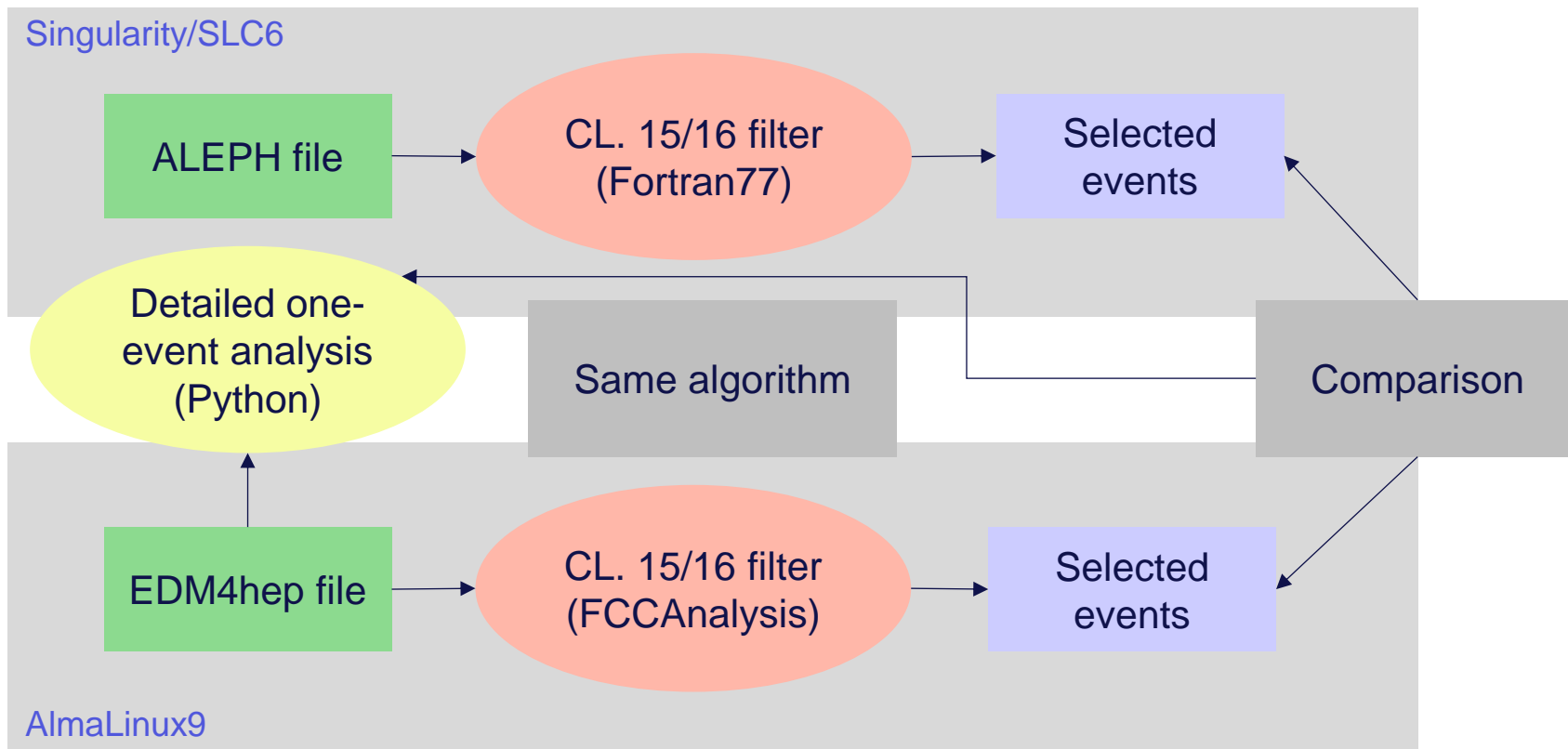
Validation results

- **A detailed comparison of old/new formats for CLASS 15 (dilepton candidates) and CLASS 16 (hadronic events) selections has been done**
 - In ALEPH it is possible to easily select the events belonging to a certain CLASS through a flag, but there is no control on the filter algorithm
 - An ALEPH selection algorithm (Fortran77), following the CLASS 15/16 discriminants, has been re-written
 - The same algorithm was applied on EDM4hep files using FCCAnalysis and a comparison of selected events was performed
 - A detailed study of events selected by only one of the filters was done through a Python script

Validation workflow



Validation workflow



Example: CLASS 16 on 1993/Y15223.10.AL

- **Only 2 events mis-selected (= not selected by FCCAnalysis filter but selected by Fortran filter) out of more than 200k (< 0.001%)**
 - Caveat: the filter algorithm is looping over the tracks, but there is not 1:1 correspondence between tracks and PF objects (= ReconstructedParticles)

```

WRITE(*, *) "===== TRACKS ===== "
TCOUNTER = 0
TTOTEN = 0
DO 102 TI = 0, 100
    TENARRAY(TI) = 0.0
102 CONTINUE
DO KNCHT = KFCHT, KLCHT
    TPTOT = QP(KNCHT)
    TPX = QX(KNCHT)
    TPY = QY(KNCHT)
    TPZ = QZ(KNCHT)
    TEN = QE(KNCHT)
    TMASS = QM(KNCHT)
    TDO = ABS(QFRFD0(KNCHT))
    TZ0 = ABS(QFRFZ0(KNCHT))
    TTPC = KFRTNT(KNCHT)
    TCOSTH = ABS(COS(PI/2 - ATAN(QFRFTL(KNCHT))))
    TCOSTHP = ABS(TPZ/TPTOT)
    WRITE(*, 103) TPTOT, TPX, TPY, TPZ, TEN, TMASS, TDO,
    TZ0, TTPC, TCOSTH, TCOSTHP
103 FORMAT(11(F10.4, 2X))

```

```

IF (TDO .GE. 2 .OR. TZ0 .GE. 10 .OR. TTPC .LT. 4 .OR.
. TCOSTHP .GT. 0.95) THEN
    WRITE(*,*) "-->BAD PART.: CUTS NOT PASSED"
ELSE
    TENARRAY(TCOUNTER) = TEN
    TCOUNTER = TCOUNTER + 1
ENDIF
ENDDO

WRITE(*,*) "# PART.: ", TCOUNTER
IF (TCOUNTER < 5) THEN
    WRITE(*,*) "--> BAD EV.: TOO FEW PARTICLES"
ELSE
    DO TI = 0, 100
        TTOTEN = TTOTEN + TENARRAY(TI)
    ENDDO
    IF (TTOTEN < 0.1 * QELEP) THEN
        WRITE(*,*) "--> BAD EV.: ", TTOTEN, " < ", 0.1*QELEP
    ELSE
        WRITE(*,*) "GOOD EV.! ", TTOTEN, " > ", 0.1*QELEP
        WRITE(64, *) KRUN, KEVT, KEVT
    ENDIF
ENDIF
ENDIF

```

Validation with MIT

- A group from MIT re-analyzed a subset of ALEPH data in recent years
 - CLASS 16 hadronic events (≥ 5 charged particles)
 - High level information: reconstructed particles from particle flow, V0s, dE/dX

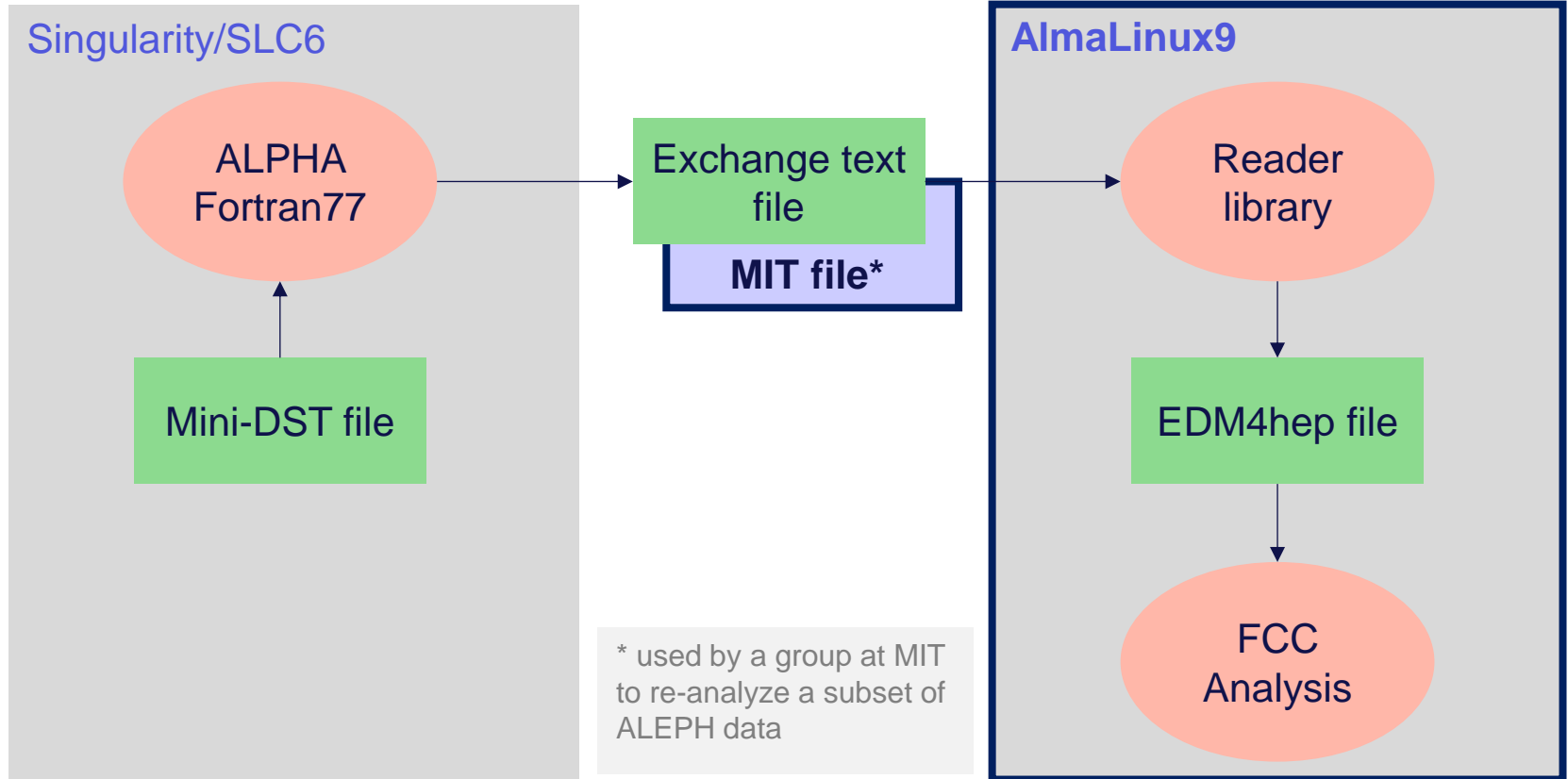
```

ALEPH_DATA RUN = 35482 EVENT      15 ECM = 91.650 GEV
Primary vertex info Flag = 4 vx = -0.9882 vx = 0.8308 ax = 0.0019 ey = 0.0000
px= -0.375 py= -0.845 pz= 0.035 m= 0.140 charge= 1.0 pwrflag= 0 lock= 1 dB= -0.725 zD= 1.155 ntpc= 16 nttc= 0 nvdet= 1 track= 1 de/dx code=0 (e-) -6.56 (p1-) 0.45 (K-) -11.91 (p) -27.42
px= -0.264 py= -0.026 pz= 0.018 m= 0.140 charge= -1.0 pwrflag= 0 lock= 1 dB= -0.047 zD= 1.373 ntpc= 11 nttc= 2 nvdet= 2 track= 2 de/dx code=0 (e-) -2.65 (p1-) 0.56 (K-) -10.64 (p) -24.37
px= 6.591 py= 1.108 pz= 0.591 m= 0.140 charge= 1.0 pwrflag= 0 lock= 1 dB= -0.009 zD= 1.338 ntpc= 17 nttc= 2 nvdet= 2 track= 3 de/dx code=0 (e-) -3.14 (p1-) -0.35 (K-) 2.04 (p) 3.41
px= 30.342 py= 4.278 pz= 1.145 m= 0.140 charge= -1.0 pwrflag= 0 lock= 1 dB= -0.006 zD= 1.537 ntpc= 15 nttc= 0 nvdet= 2 track= 4 de/dx code=0 (e-) -2.00 (p1-) -0.71 (K-) 0.68 (p) 1.75
px= -7.908 py= -1.061 pz= -0.332 m= 0.140 charge= -1.0 pwrflag= 0 lock= 1 dB= 0.009 zD= 1.331 ntpc= 21 nttc= 0 nvdet= 2 track= 5 de/dx code=0 (e-) -3.45 (p1-) -0.35 (K-) 2.26 (p) 3.85
px= -2.927 py= -0.017 pz= -0.687 m= 0.140 charge= -1.0 pwrflag= 0 lock= 1 dB= 0.004 zD= 1.343 ntpc= 18 nttc= 3 nvdet= 2 track= 6 de/dx code=0 (e-) -2.84 (p1-) 0.40 (K-) 2.42 (p) 2.89
px= -1.499 py= -0.338 pz= 0.108 m= 0.140 charge= 1.0 pwrflag= 0 lock= 1 dB= 0.424 zD= 0.932 ntpc= 20 nttc= 4 nvdet= 0 track= 7 de/dx code=0 (e-) -5.40 (p1-) -0.50 (K-) 0.95 (p) -0.99
px= 1.498 py= 0.681 pz= 0.439 m= 0.140 charge= 1.0 pwrflag= 0 lock= 1 dB= -0.011 zD= 1.323 ntpc= 17 nttc= 2 nvdet= 2 track= 8 de/dx code=0 (e-) -5.48 (p1-) 0.08 (K-) 1.94 (p) 0.89
px= -3.652 py= -0.185 pz= -0.575 m= 0.140 charge= 1.0 pwrflag= 0 lock= 1 dB= -0.162 zD= 0.576 ntpc= 11 nttc= 2 nvdet= 0 track= 9 de/dx code=0 (e-) -6.14 (p1-) -3.43 (K-) -1.60 (p) -1.80
px= -0.960 py= 0.849 pz= 0.198 m= 0.000 charge= -1.0 pwrflag= 0 lock= 1 dB= 0.008 zD= 1.325 ntpc= 14 nttc= 0 nvdet= 1 track= 11 de/dx code=0 (e-) -4.47 (p1-) -0.34 (K-) -0.45 (p) -4.65
px= 0.418 py= 0.139 pz= 0.306 m= 0.140 charge= -1.0 pwrflag= 0 lock= 1 dB= -0.193 zD= 1.345 ntpc= 15 nttc= 2 nvdet= 2 track= 13 de/dx code=0 (e-) -6.66 (p1-) 1.31 (K-) -6.34 (p) -19.28
px= 1.857 py= 0.245 pz= 0.030 m= 0.000 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= 0.822 py= 0.140 pz= 0.069 m= 0.000 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= 1.333 py= 0.117 pz= 0.260 m= 0.000 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= 0.959 py= 0.203 pz= 0.198 m= 0.000 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= 1.350 py= 0.585 pz= -0.109 m= 0.000 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= -2.373 py= -0.260 pz= 0.081 m= 0.022 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= -3.243 py= -0.473 pz= 0.049 m= 0.001 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= -2.128 py= 0.011 pz= -0.584 m= 0.021 charge= 0.0 pwrflag= 4 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
px= -9.851 py= -1.656 pz= -0.410 m= 1.269 charge= 0.0 pwrflag= 5 lock= 1 dB= -1.000 zD= -1.000 ntpc= 0 nttc= 0 nvdet= 0 track= 0 de/dx code=1 (e-) -1.00 (p1-) -1.00 (K-) -1.00 (p) -1.00
vx= -7.49 vy= -1.23 vz= 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= 0.93 vz= 1.34 ch12 = 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 ch12 = 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 ch12 = 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 ch12 = 0.000 type=0 Ntrk= 2
Track= 7 px= -1.502 py= -0.327 pz= 0.111
Track= 6 px= -2.927 py= -0.026 pz= -0.687
vx= -0.09 vy= 0.04 vz= 1.32 ch12 = 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

```

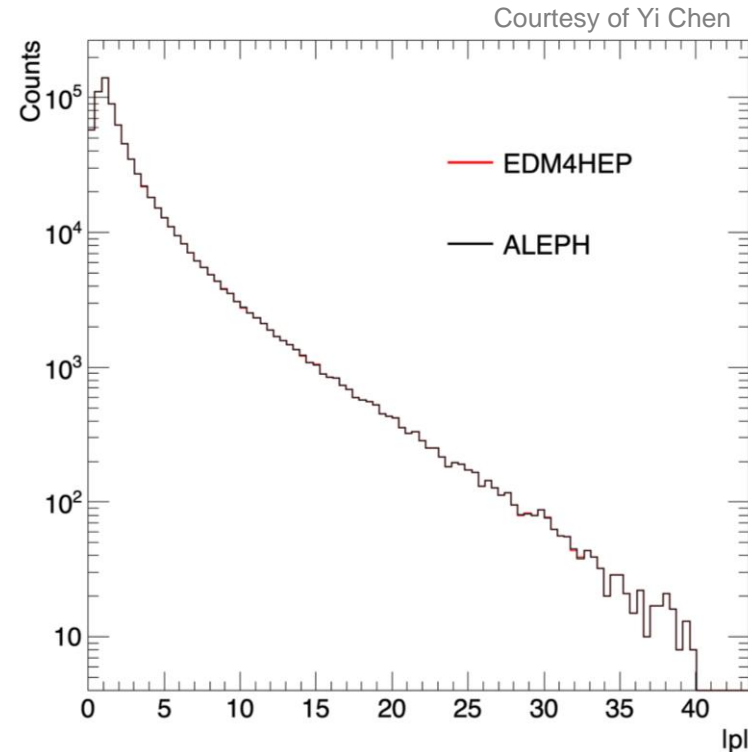
See also:
 Yi Chen et al., [First measurement of anti-kT jet spectra and jet substructure using the archived ALEPH e+e- data at 91.2 GeV, 41st ICHEP](#)
 Yi Chen, [Revisiting the ALEPH Archived e+e- Data, CERN EP Seminar](#)

Workflow with MIT



Strategy and preliminary results

- **A tutorial to extract a flat ROOT TTree from the EDM4hep file using EDM4hep Utilities has been provided**
 - In this way EDM4hep files are easily analyzable via ROOT
 - Users can define their functions to store in the TTree the relevant quantities for their analysis
 - Same approach can be applied to analyze EDM4hep files using Python
- **Positive preliminary feedbacks**



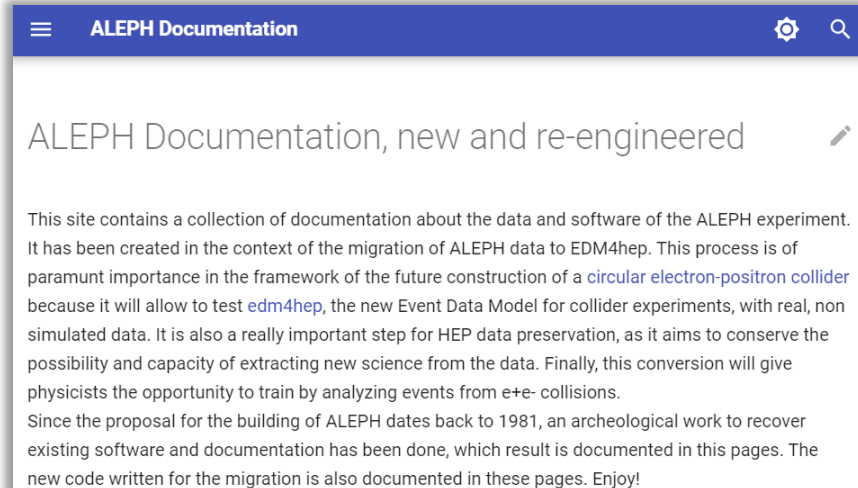


Summary & Outlook

Summarizing: achievements

- **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**
- **Began validation of the workflow**
 - Translated CLASS 15 and CLASS 16 selection algorithms in FCCAnalysis/Python
 - Provided MIT colleagues with tools to validate the workflow
- **Collected archeological information and tutorials on a website (beta)**

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



ALEPH Documentation

ALEPH Documentation, new and re-engineered

This site contains a collection of documentation about the data and software of the ALEPH experiment. It has been created in the context of the migration of ALEPH data to EDM4hep. This process is of paramount importance in the framework of the future construction of a [circular electron-positron collider](#) because it will allow to test [edm4hep](#), the new Event Data Model for collider experiments, with real, non simulated data. It is also a really important step for HEP data preservation, as it aims to conserve the possibility and capacity of extracting new science from the data. Finally, this conversion will give physicists the opportunity to train by analyzing events from e+e- collisions. Since the proposal for the building of ALEPH dates back to 1981, an archeological work to recover existing software and documentation has been done, which result is documented in this pages. The new code written for the migration is also documented in these pages. Enjoy!

Work ahead

- **Extend to Monte Carlo data, including truth information**
 - The MC ALEPH code is available: for new MC sets, a converter to use new generator files (e. g. HepMC3) is being developed to generate new simulations
- **Convert all 1994 data and related MC for evaluation of selected users**
 - Some groups have already manifested interest the evaluation
 - Loop with users feedbacks to correct bugs and improve the process (e. g. understanding which variables they ask for analysis)
- **Convert all data files and make them available**
 - Including existing MC files
- **Migrate database with meta-data (fill, run, luminosity, det. status, ...) to a modern backend and interface, including MC**
 - Including new location of files on EOS
- **Go on with validation**
 - Translating old selection algorithms and applying them to converted data to study the differences



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