

# LHCb AOD Model

- Computing model
- Creation of datasets for physics analysis
- Usage of datasets in physics analysis
- Future directions
- Data Model



# General Considerations

- B physics in LHCb is the analysis of decay channels within families, where each family uses its own dataset
- Large reduction factors between recorded data and analysis datasets
- One cannot talk about data without considering the tools to access them
- Except online raw data all data are in ROOT



# Logical Dataflow Model (1)

$2 \times 10^{10}$

RAW Data (MC)

RAW Data

35 kB

Reconstruction

$2 \times 10^{10}$

r(educed)DST

20 kB

# Events

Preselection  
(Stripping)

Size/Event

$\sim 20 \times 10^7$

RAW + DST

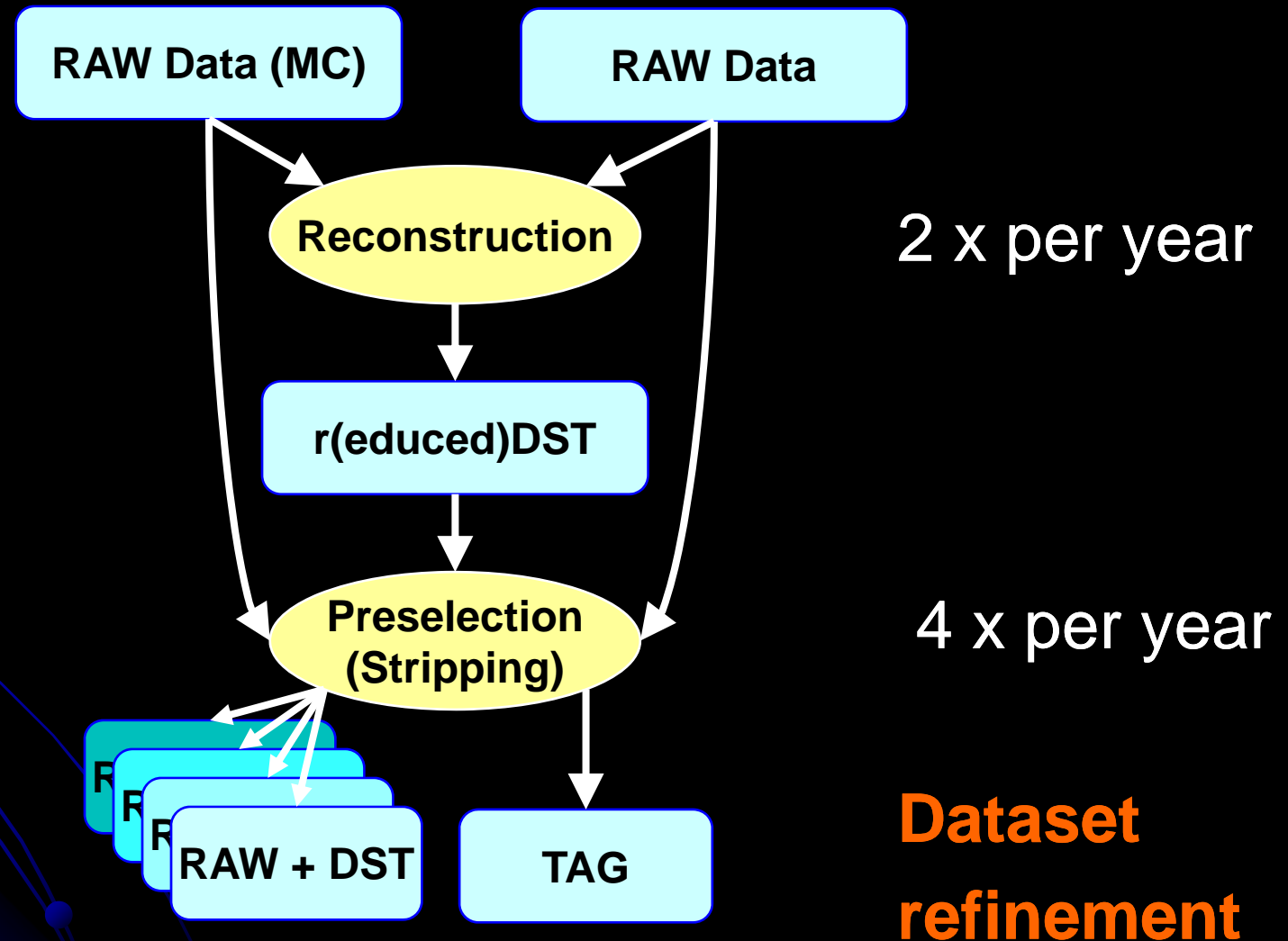
TAG

125 / 1 kB

MC:400 kB

Analysis Input

# Logical Dataflow Model (2)



# Logical Dataflow Model (3)

700 TB

RAW Data (MC)

RAW Data

Tier0 (full)  
+Tier1s

Reconstruction

500 TB

r(educed)DST

Tier1

**Dataset size**

Preselection  
(Stripping)

**Location**

~20 x 1.2 TB

RAW + DST

TAG

Fully  
replicated  
on all Tier1

Other events only used for  
calibration and detector studies

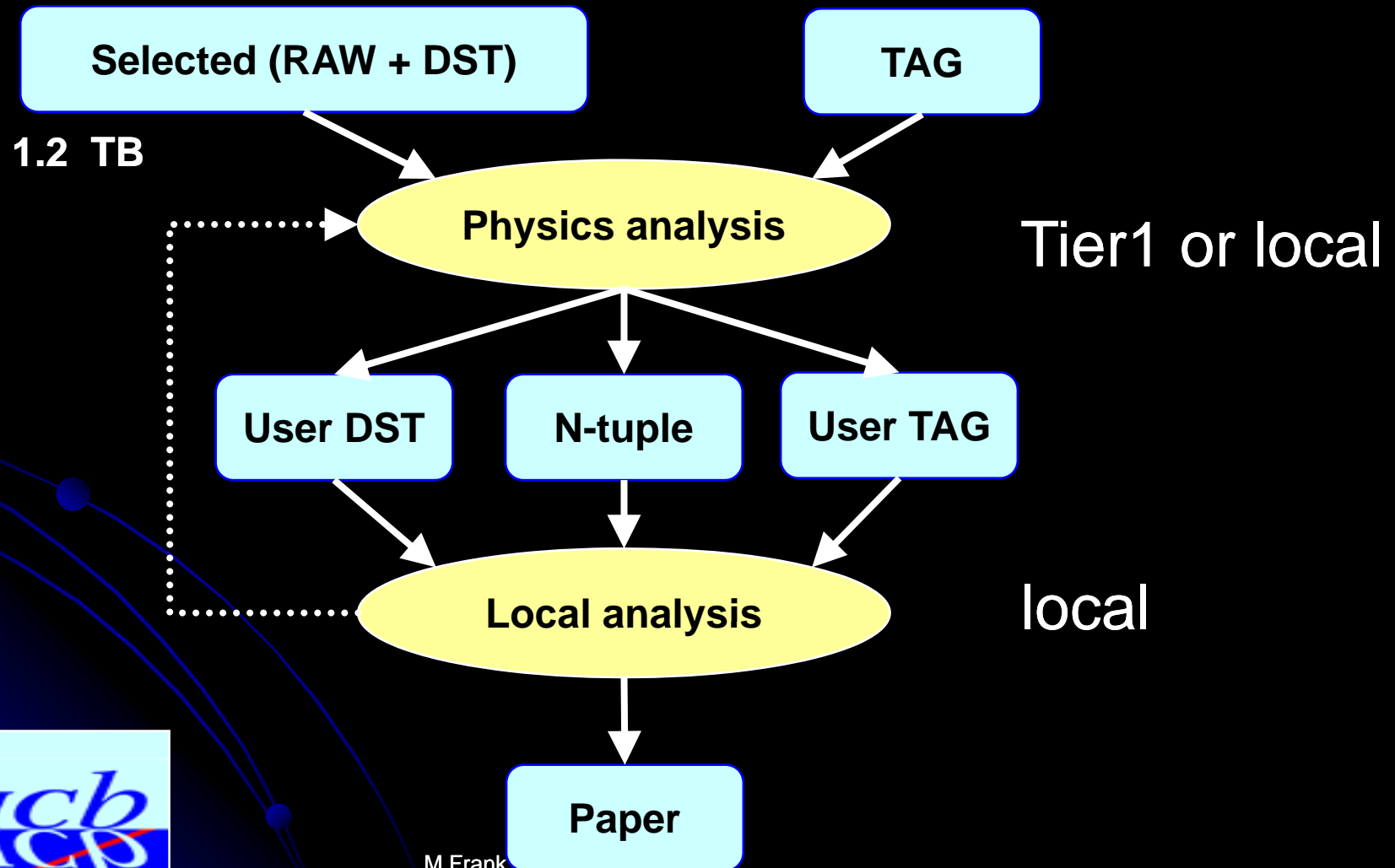
500 GB external HDD: 160 EUR  
This fits on the desktop

# Summary In Short...

- Physicists never see full sample of  $2 \times 10^{10}$  events
- Data are preselected in a collaboration effort (refinement 4 x per year)
- Only analysis datasets defined by the physics groups (+physics coordinator) are used by individual physicists (or ~99 %)
- Analysis datasets contain “everything” and are available “everywhere” ...
  - typically can be replicated to the desktop



# Physicist Analysis Cycle



# Physics Analysis

- All analysis is based on datasets of a size of roughly 1 TB
- All analysis is based on Gaudi
  - Data access from transient event store
  - Various flavours
    - DaVinci: traditional C++ or Loki style
    - GaudiPython: with or without Loki style (Bender)
  - All flavors can be executed interactively
  - ...or using Ganga
    - Job splitting and output merging capabilities
    - GaudiPython eliminates the platform dependency
    - Joint LHCb / ATLAS development





# Loki & Bender

- Based on ideas from
  - `KAL` by H.Albrecht => dense physics code
  - `Pattern` and `GCombiner` by T.Glebe (Hera-B)
  - Developed by Vanya Belyaev
- A world of functors, adapters and metafunctions hiding all technicalities
- Transparent physical content of the code
- Needs a C++ compiler, but looks less like it...
- Bender is the python export of Loki using Reflex



# Loki Example

## Selection criteria

**K:** IP to all PVs  $\chi^2 > 4$

**$\phi$ :**  $|M(K^+K^-) - M(\phi)| < 12 \text{ MeV}/c^2$  ;  
 $\Gamma = 4.5 \text{ MeV}/c^2$  ,  $\sigma = 1.1 \text{ MeV}/c^2$

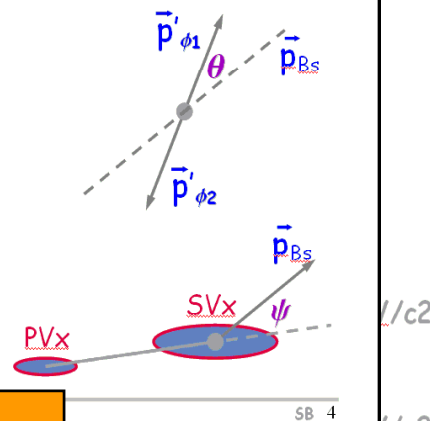
$p(\phi) > 12 \text{ GeV}/c$

**$B_s$ :**  $|M(\phi\phi) - M(B_s)| < 24 \text{ MeV}/c^2$  ;  
 $\sigma = 12.1 \text{ MeV}/c^2$

proper time  $0.2 < \tau/\tau(B_s) < 4.$

$|lv01| < 0.75$  ,  $lv01 = \cos \theta$

direction angle  $|\psi| < 10 \text{ mrad}$



1 PPT slide of cut description

$\chi^2(Vx) < 100$

1 page A4 of c++ code

S.Barsuk [2003]

```

LOKI ALGORITHM( Bs2PhiPhi )
{
  const StatusCode ok ( StatusCode::SUCCESS ) ;
  const StatusCode error ( StatusCode::FAILURE ) ;

  satFilterPassed ( false ) ;

  // get all charged kaons
  Range kplus = select( "K+", -321 == ID ) ;
  Range kminus = select( "K-", 321 == ID ) ;

  for( Loop phi = loop( "K+ K-", 333 ) ; phi ; ++phi )
  {
    if( phi->mass(1,2) > ( 1020 + 50 ) * MeV ) { continue ; }
    phi->save( "phi" ) ;
  }
  Range phis = selected( "phi" ) ;

  Tuple tuple = utuple ( "Bs" ) ;
  for( Loop Bs = loop( "phi phi", 531 ) ; Bs ; ++Bs )
  {
    // take phi,phi pairs with 5 GeV < M < 6 GeV
    if( Bs->mass(1,2) < 5 * GeV ) { continue ; }
    if( Bs->mass(1,2) > 6 * GeV ) { continue ; }

    // take Bs with chi2Vx ( Bs ) < 100
    if( VCHI2( Bs ) > 100 ) { continue ; }

    double mphi1 = M( child( Bs , 1 ) ) ;
    double mphi2 = M( child( Bs , 2 ) ) ;

    Record rec( tuple, "mass,mphi1,mphi2,p,pt,lv01,bschi2",
              M ( Bs ) , mphi1 , mphi2 ,
              P ( Bs ) , PT ( Bs ) , LV01 ( Bs ) , VCHI2( Bs ) ) ;

    Bs->save( "Bs" ) ;
  }

  // get saved Bs
  Range Bs = selected( "Bs" ) ;

  if( !Bs.empty() ) { satFilterPassed ( true ) ; }

  return StatusCode::SUCCESS ;
}
    
```

Ch-light meeting , CERN 27.05.2003

SB 3



# Local Analysis

- Inputs are “user defined” data
  - Tags, N-Tuples, USER-DST
- Reference to event data (RAW+DST) present
- NTuples and Tags are functionally the same
  - ROOT trees in split mode
  - Tag items may be objects  
Splitting depends on ROOT (pointers,...)
- USER DST may be:
  - DST with less objects
  - DST with user defined objects



# Interactive Tools for Local Analysis

- User select their favorite analysis tool
  - PAW, ROOT, PyROOT => without framework
  - GaudiPython => with framework
    - Integration of event display (Panoramix)
    - Full ROOT functionality: e.g. TCanvas
    - HippoDraw
    - Excel



# “User Defined” Data: Future Directions

- Experience showed that complete freedom of user defined data is counterproductive
  - formats do not match
  - No common tools, scripts etc.
- Attempts to define micro DST / Tags for the usage within analysis groups
  - microDST Initiative @ University of Cambridge [U.Kerzel et al.]



# “User Defined” Data: Future Directions

- Envisaged size: very few kB / event
- Data content are high level analysis objects
  - Primary vertices
  - B candidates, their decay products and associated tracks and vertices:  $O(1)$  candidates per event
- Full collaboration with framework
- Common data/tag format
- For the usage within one group ONLY
  - Not every group considers the same candidate as “good”
  - Frequent (re)creation.  
Creation can be done by a single physicist

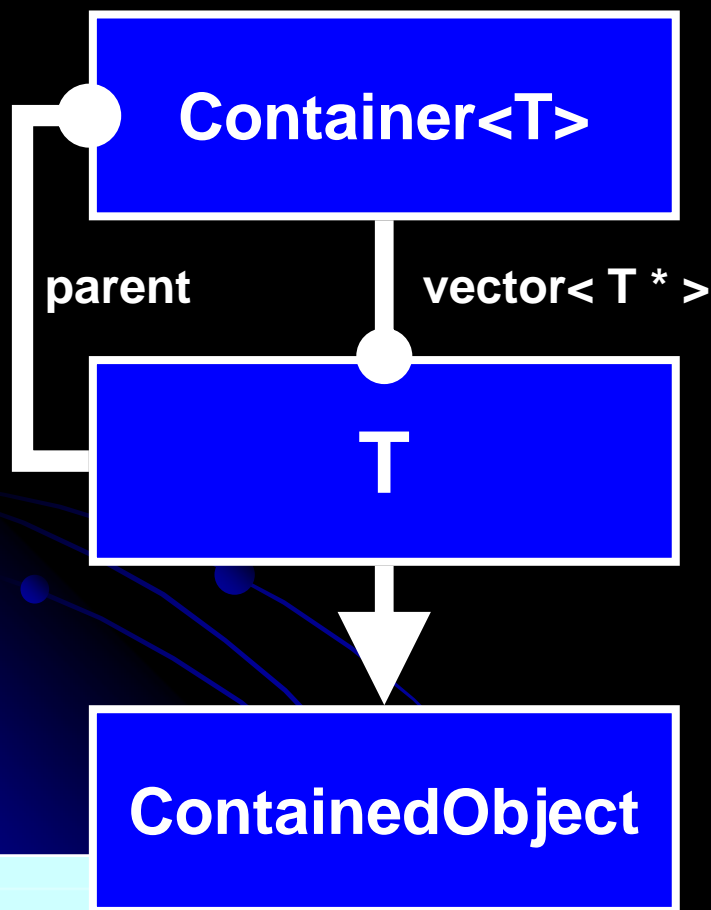


# “User Defined” Data: Future Directions

- Currently 2 directions
  - “microDST”           => POOL/ROOT data file
  - Event tag++           => N-tuple enhanced by objects
  - Splitting will depend on the object model
- Final data content not yet clear
  - Prototype suggests data reduction of  
MC FullDst : Tag++ or microDST is 200:1



# Data Model



- Sufficiently simple
- > 99 % of all data
  - FullDST
  - probably also microDST
- Homomorph Container<T>
  - Despite all “experts”
- Top Level TTree branches
- We are interested in “forced splitting” of top level branches
  - HowTo is known, but only using ROOT internal knowledge





# ROOT Streaming

- We do not always save all data
  - References: expansion of 2 integers
- Interest for:
  - Pre-write object processing
  - Post-read object processing
- Currently implemented using
  - Custom Streamer +  
TClass::ReadBuffer / TClass::WriteBuffer



# Comparison

	Unit	ALICE		ATLAS	CMS	LHCb	
Hoffmann 2k		p-p	Pb-Pb				
RAW Event	MB	1	25	1	1	0.125	
REC/ESD Event	MB	0.1	2.5	0.5	0.5	0.1	
AOD Event	kB	10	250	10	10	20	
TAG Event	kB	1	10	0.1	1	1	
<b>LHCC 2005</b>						<b>Comp.TDR</b>	
RAW Event	MB	1	12.5	1.6	1.5	0.025	
REC/ESD Event	MB	0.2	2.5	0.5	0.25	0.125	
AOD Event	kB	50	250	100	50	2..5 (?)	microDST
TAG Event	kB	10	10	1	10	1	TAG
						2..5 (?)	TAG++



# Conclusions

- LHCb AODs are not main analysis input
  - Combination of FullDST and Tag like quantities
  - Analysis groups access directly stripped FullDST
    - ...but from reduced preselected streams
    - ...data are present everywhere
    - ...may produce summary data (AOD)
  - “AOD”s are expected to be very small
    - ...and created by individuals / analysis groups
    - ...common format
    - ...but the meaning of the content may depend on the analysis group or analysis dependent

