

# A New Physics Analysis Framework

Thomas Kuhr  
University of Hamburg  
Thomas.Kuhr@desy.de



ROOT Workshop June 14<sup>th</sup> 2001

# Overview

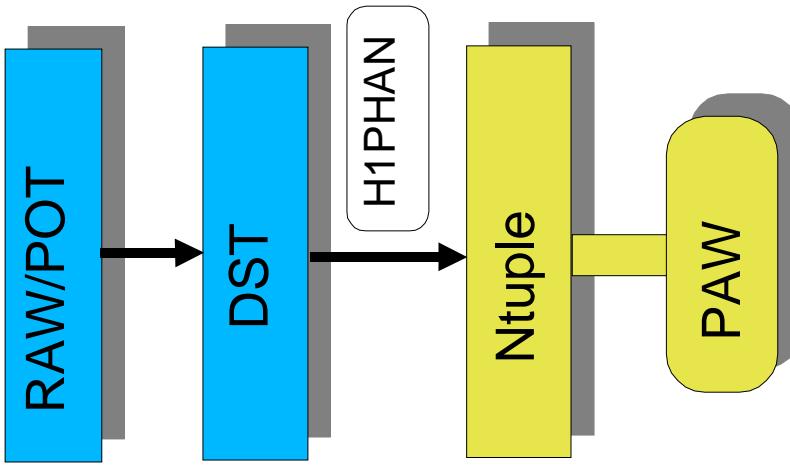
- **Data Storage Model**
  - Three layer structure, conversion from old format, persistent relations
- **Data Access / Analysis Framework**
  - H1Tree class, Run catalog, branch pointers
- **Event Display**

# Old Framework

- Data stored in **BOS** format
- BOS banks described by **DDL**
- Data processed by **H1PHAN** package
- Output: ntuple in **HBOOK** file analysed with **PAW**

→**Stable, well established but diverged and inhomogeneous environment based on old technology**

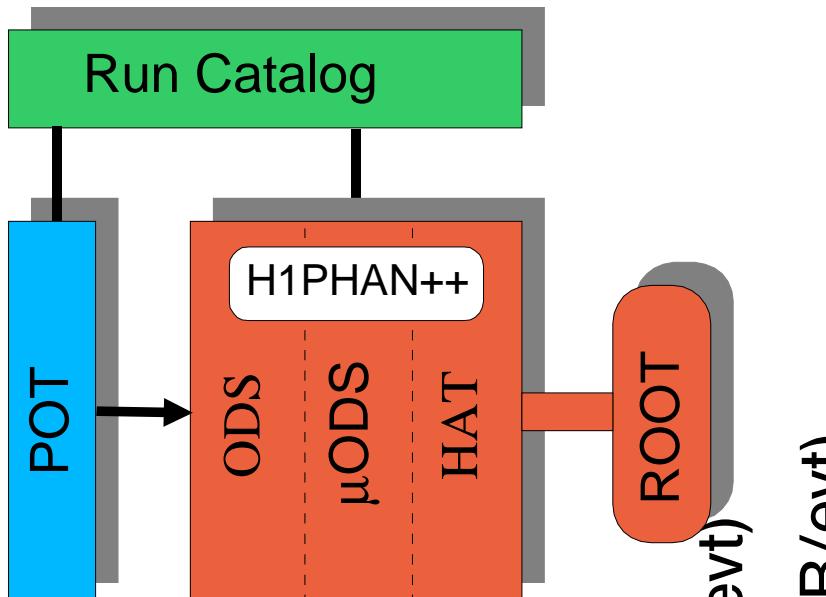
→**No backward connection ntuple to data**



# New Storage Model

\* Based on ROOT

- \* Three hierarchical layers (separate files with TTrees):
  - ◆ Reconstruction: ODS (Object Data Storage, 13 kB/evt)



- ◆ Particle: μODS (micro Object Data Storage, 1 kB/evt)
  - ◆ Event: HAT (H1 Analysis Tag, 0.4 kB/evt)
- \* Additional layer: user tree

→ Common environment for H1 and user data

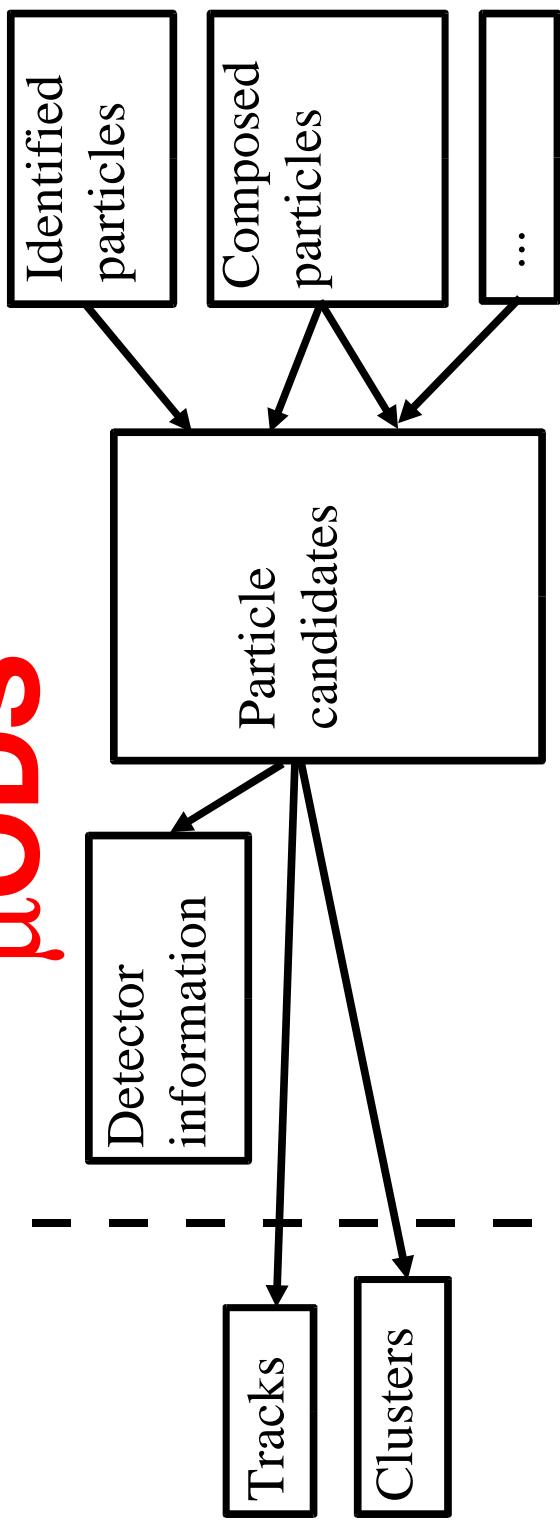
# ODS

- ◆ 1-to-1 correspondence to DST
- ◆ ODS class  $\leftrightarrow$  BOS bank
- ◆ ODS classes automatically generated from DDL
- ◆ New classes for **tracks** and **clusters** with user friendly accessor methods replace some of the ODS banks
- ◆ **Backward conversion** ODS to BOS possible

# HAT and $\mu$ ODS

- $\times$  New layers without correspondence in old scheme
- $\times$  HAT: Event level information, e.g. kinem.
- $\times$  Fast event selection with HAT
- $\times$   $\mu$ ODS: Particle information
- $\times$  Make expert knowledge of physics working groups persistent

# $\mu$ ODS



- x List of particle candidates **without double counting**

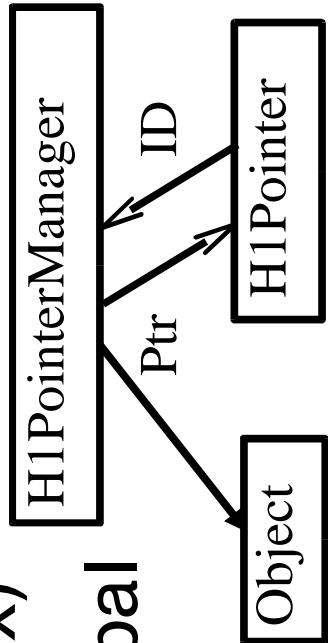
- x **Particle candidate:** kinematics, pointers to track and cluster
- x **Identified/composed particle(s):** pointer(s) to particle candidate(s)

# Migration of Filling Code

- H1 is a running experiment
  - No time to rewrite analysis code from scratch
  - Sustain quality / precision of data
- Presently mostly using Fortran code on old data format to fill μODS and HAT
- Gradually replacing Fortran code by new OO code

# H1Pointer

- Need to store relations between objects (e.g. id. particle → particle cand. → track)
- H1 specific solution provided by **H1Pointer**
- Pointer relation is translated to **unique integer ID** (tree + branch + entry index)
  - Unique IDs maintained by global **H1PointerManager**
  - Persistent arrays of pointers (TObjArray) provided by **H1ArrayPointer** class
    - Data automatically loaded when H1 Pointer dereferenced



# H1Tree and Run Catalog

- Encapsulates access to trees
- File names for run/event numbers obtained from run catalog (MySQL database)
- File access only when necessary
- Event selection on HAT
- Creates/uses event lists
- Writes/reads user tree

# Branch Pointers

- Consistent and transparent access to data in all layers
  - No type cast
  - Pointer operators overloaded
  - Type check
  - Source file automatically generated
- ```
H1EvtKinePtr kine( "Kinematics" );  
Float_t Q2 = kine->GetQ2e();
```

# Example

```
H1Tree tree(273473, 278085);  
  
tree.SelectHAT("fQ2e>1");  
  
H1PartScatterElectArrayPtr electrons;  
  
while (tree.Next()) {  
  
    Int_t nElectrons = electrons.GetEntries();  
  
    for (Int_t i = 0; i < nElectrons; i++) {  
  
        Float_t Q2 = electrons[i]->GetQ2();  
  
        Float_t E = electrons[i]->GetCluster()->GetE();  
  
    }  
}
```

get ODS, μODS, HAT file from run catalog

create branch pointer to array of scattered electrons

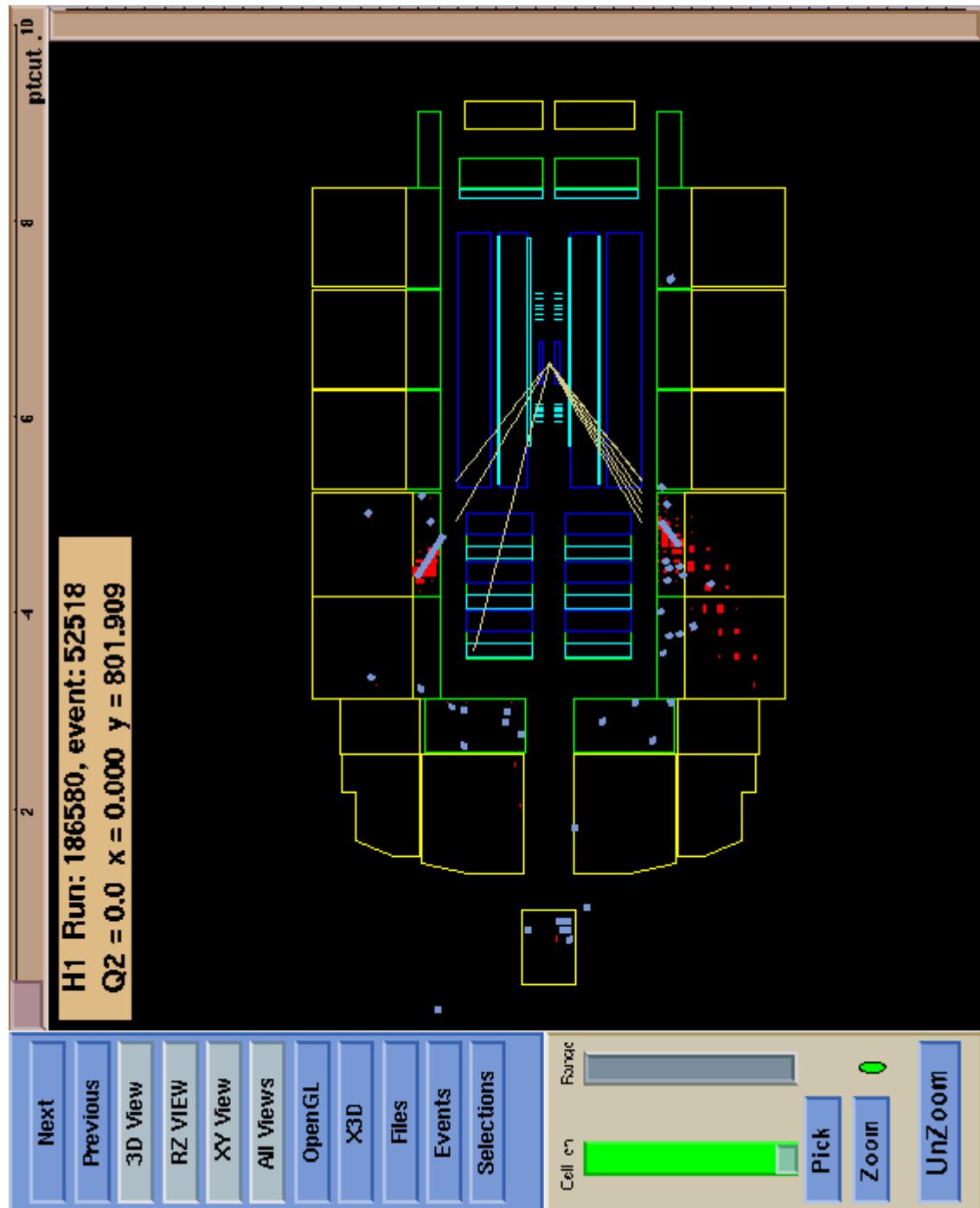
access  $i^{\text{th}}$  electron

access ODS object via  
H1Pointer on μODS

# Event Display

- Integrated in OO-framework
  - e.g. event selection
- Can display data in old and new format
  - Modern GUI
  - Pick and inspect
- Also understands commands of old event display
  - Backward compatible

# An Event



Thomas Kuhr

# Conclusions

- ✓ A new ROOT-based data storage model has been implemented
- ✓ An object-oriented analysis framework has been established
- ✓ ROOT extensions: `H1Tree`, `run catalog`, `H1Pointer`, `branch pointer`, `event display`
- ✓ Migration from Fortran to C++ analysis `code` is in progress
- ✓ First analyses have been started

# Wish list

- Ideas to improve the very useful **html documentation**:
  - Link to header files
  - Show inherited methods
  - Gzip ps files
  - Index page per directory/package
- Execution of history files
- recommendations/reviews of ROOT capable **CASE tools** (IDE, editor, debugger, profiler, UML)