10 Years of Object-Oriented Analysis on H1

Paul Laycock

Department of High Energy Physics, Oliver Lodge Laboratory, University of Liverpool, Liverpool, L69 7ZE, UK

E-mail: laycock@hep.ph.liv.ac.uk

Abstract. Over a decade ago, the H1 Collaboration decided to embrace the object-oriented paradigm and completely redesign its data analysis model and data storage format. The event data model, based on the ROOT framework, consists of three layers - tracks and calorimeter clusters, identified particles and finally event summary data - with a singleton class providing unified access. This original solution was then augmented with a fourth layer containing user-defined objects.

This contribution will summarise the history of the solutions used, from modifications to the original design, to the evolution of the high-level end-user analysis object framework which is used by H1 today. Several important issues are addressed - the portability of expert knowledge to increase the efficiency of data analysis, the flexibility of the framework to incorporate new analyses, the performance and ease of use, and lessons learned for future projects.

1. Introduction

The H1 experiment analysed electron-proton data from the HERA machine at DESY, Hamburg. During the luminosity upgrade at the turn of the millennium, H1 decided to change their analysis framework to an object-oriented paradigm, concentrating on the data storage model [1, 2, 3, 4].

2. The Analysis Models

2.1. The old analysis model

Figure 1 shows the old data analysis model of H1. All of the triggered RAW data from the detector was reconstructed using the H1 reconstruction software package written in Fortran into a BOS [5] bank format. The entire reconstruction output was stored as Physics Output Tape (POT), a summary of which was written to (a more easily accessible) Data Summary Tape (DST). The actual physical storage media of course changed with time, while the names did not.

The DST was processed privately by private ntuple production code to produce the analysis data format. This was usually the HBOOK format, and physics analysis was performed using PAW [6]. The scope for sharing analysis information was fairly limited, as the variable definitions depended on the privately maintained ntuple production code which varied from group of data analysts to another. Meanwhile, the private production of ntuples was usually quite inefficient, resulting in many copies of essentially the same data, wasting resources.

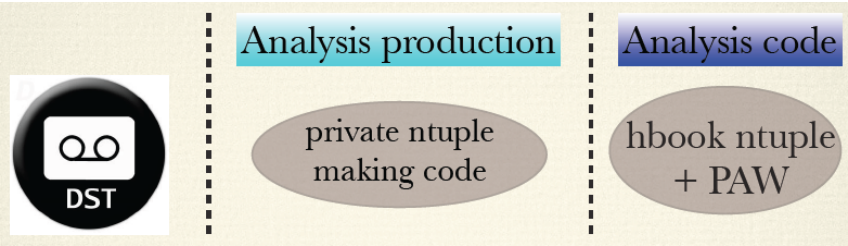

Figure 1. The old data analysis model.

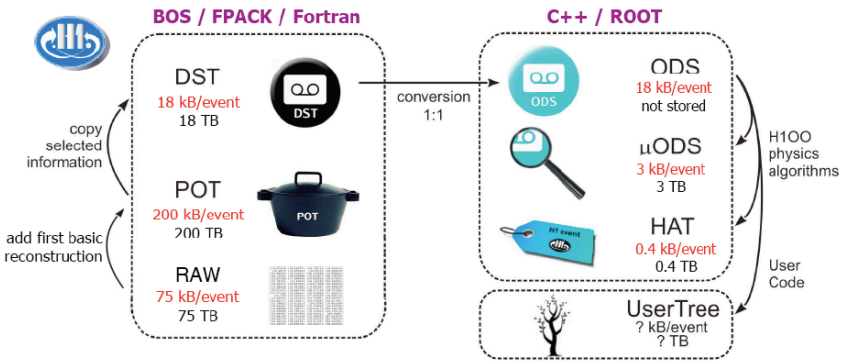

Figure 2. The new data storage model of H1.

2.2. The H1OO data analysis model

Figure 2 shows the object-oriented data analysis model. The aim was to centrally produce the analysis data storage format, replacing the custom ntuples used previously. Private ntuple production was not prohibited and was used by some individuals, but the vast majority of the collaboration enjoyed and still enjoys central production of their analysis data format.

2.3. The H1OO data storage model

The DST remains the source of all derived analysis formats. The object-oriented data format consists of several layers, and all data storage layers are encapsulated in the H1Tree interface. This object provides smart access to the data, such that the user doesn't need to know which layer is accessed.

The Object Data Storage (ODS) layer is entirely equivalent to the DST, but now Track and Cluster classes are used for the data representation. Object-oriented bank classes also exist, providing DST, POT and RAW data access if and as necessary. The ODS was generally not used for analysis, as the content of the two derived layers described next was usually found to be sufficient. However, it can be produced and stored if necessary (if the ODS content will be repeatedly analysed), or created on-the-fly from DST (more performant for rare access).

Particle finders, shown in Figure 3, are used to produce the content stored in the next object-oriented layer, the micro-ODS or MODS. The particle-finding algorithms used represent the best knowledge of H1 by definition, and thus all analyses benefit from this best knowledge. Event summary information, e.g. the primary vertex coordinates and the number of each type of particle stored on the MODS, is stored in the third layer, the H1 Analysis Tag or HAT. The HAT also contains experimental conditions such as beam energies and other information critical for analysis calculations. Together, the MODS and HAT layers are the analysis data format,

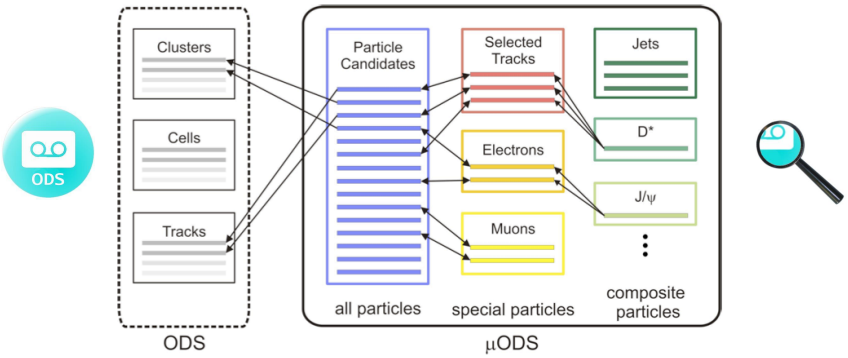

Figure 3. The Particle Finder model.

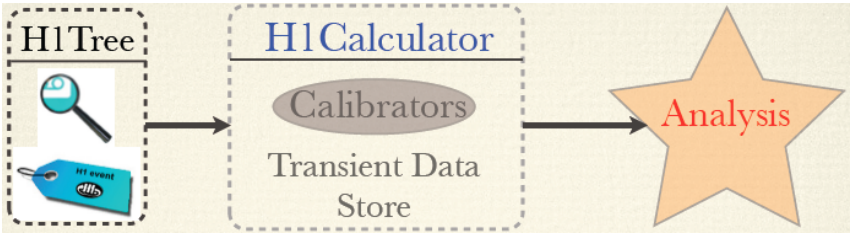

Figure 4. The transient data interface, H1Calculator.

centrally produced, for the vast majority of H1 analyses.

A fourth and final optional layer, the UserTree, allows ultimate flexibility by allowing a user-defined storage layer. If this layer was found to be useful for several groups, it entered the central production framework and was centrally maintained. It's worth noting that there are currently three UserTree packages in the core H1OO framework, suggesting that this flexibility was both necessary and useful.

3. Transient Data

The original H1OO concept concentrated on unifying the data analysis format and optimising resource usage for its production, and this was very successful. However, there were one or two limitations to this model. Chief among these was the lack of treatment of transient data, i.e. quantities calculated or recalculated at run time. This was found to be particularly relevant for the evaluation of systematic uncertainties, where derived quantities need to be recalculated based on a systematically shifted base quantity. It was also relevant when analysts wanted to apply the latest calibrations on-the-fly, rather than wait for a full-scale production of data and Monte Carlo.

3.1. Transient data interface

To address these problems, a transient data interface called the H1Calculator was designed and implemented [7, 8], as shown in Figure 4. This reads the persistent data provided by the H1Tree, and stores transient derived quantities. The latest calibration can then be applied easily, and systematic uncertainty evaluation simply requires recalculating the derived quantities. All data access in user analysis code goes through this interface, guaranteeing a consistent treatment of the data.

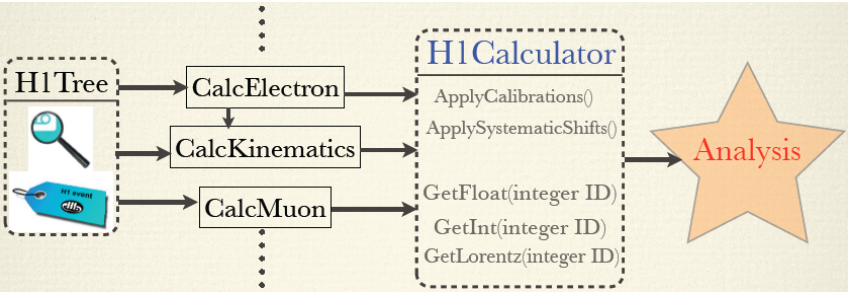

Figure 5. A more detailed view of the transient data interface.

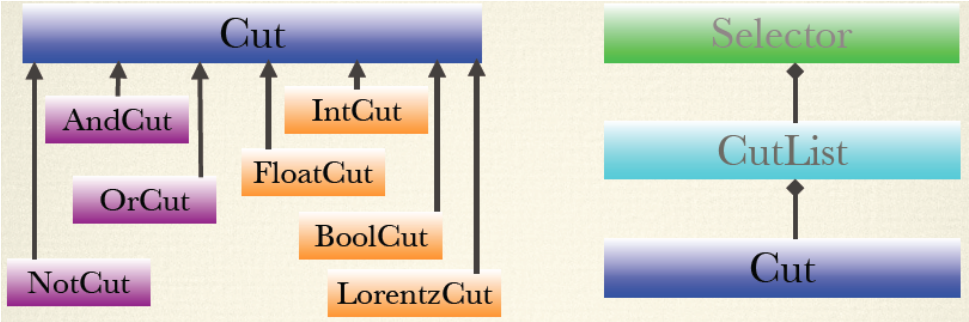

Figure 6. The event selector classes.

In practice, the scope for derived quantities for all H1 analyses is very large, and the H1Calculator is composed of several smaller, themed calculator classes which deal with specific quantities, e.g. one for electron quantities, another for event kinematics. A generic interface to the data provides access to variables by type (integer, TLorentzVector, etc.), which then allows user classes to be decoupled from the details of this structure. The main H1Calculator class itself then provides a simple interface to variables by integer ID (the generic interface) as well as switches to apply calibrations and systematic shifts, as shown in Figure 5.

4. Higher Level Analysis Objects

Following the generic interface to the transient data, more end-user classes could be defined. One particular highlight are the event selector classes [7], composed of lists of cut objects. A cut object returns a boolean answer based either on a variable read from the transient data interface or on a logical combination of other cut objects. This simple but very useful set of classes is shown in Figure 6. They also provide detailed debugging information and cutflow statistics. These classes could be passed from analyst to analyst to apply particular event selections, together with other simple classes responsible for histogram management [7], i.e. classes which book and fill sets of (related) histograms. These simple organisational aides proved to be very useful, not least in the context of data quality and software validation, as well as in physics analysis.

4.1. Analysis objects

Figure 7 shows a flow diagram of a simplified physics analysis. The selector and histogram manager objects can be matched to "Event Selection" and "Fill Histogram", respectively. The decisions made by the selector are passed on to an object or process which "Bins the event",

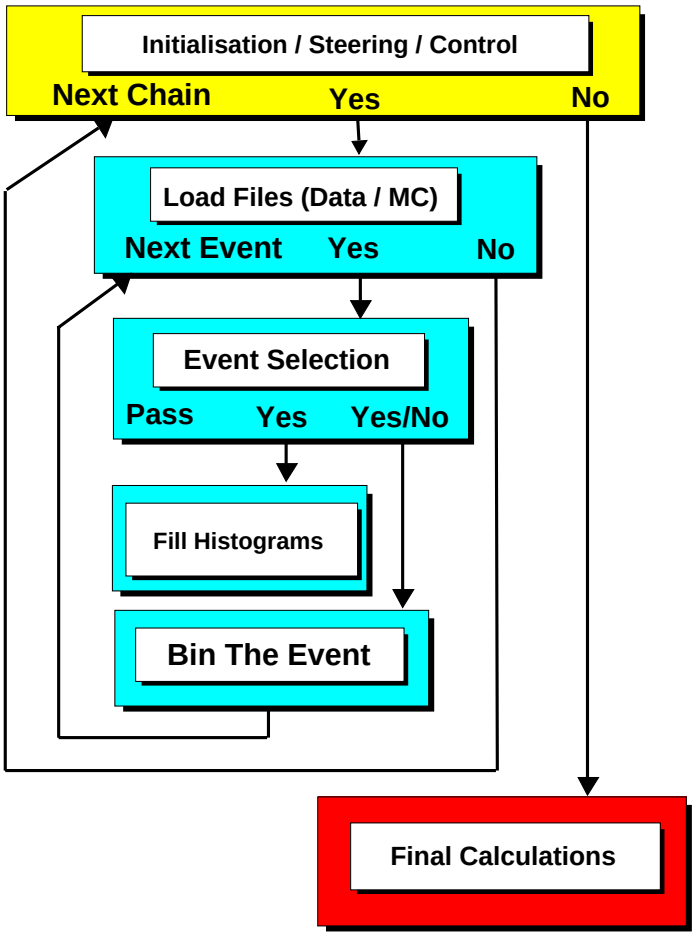

Figure 7. Flow diagram of physics analysis.

determining the reconstructed and truth-level bins in order to correct the data to a final physics measurement (taken care of in "Final Calculations"). The event information itself is made available to these various objects via the H1Calculator.

Two other organisational structures can also be seen, namely at the level of looping over one set of data or Monte Carlo, where we used the term "chain" from PAW, and finally the analysis level itself.

Correspondingly, Analysis and AnalysisChain objects also proved to be useful organisational classes [7]. An Analysis object is composed of several AnalysisChains, which each contain one or more Histogram Managers. The Event Selector is an object defined at the Analysis level. Data access goes through the H1Calculator. This simple model is shown in Figure 8 and allows a common framework for nearly all stages of a physics analysis. This in turn allowed better collaboration between physics groups, and the easy exchange of high-level analysis code.

5. Conclusions

During the luminosity upgrade at the turn of the millennium, H1 decided to change their analysis framework to an object-oriented paradigm, concentrating on the data storage model. This move was very successful in unifying the data storage model and analysis formats of H1. The flexibility to have a user defined data storage layer proved crucial in several analyses.

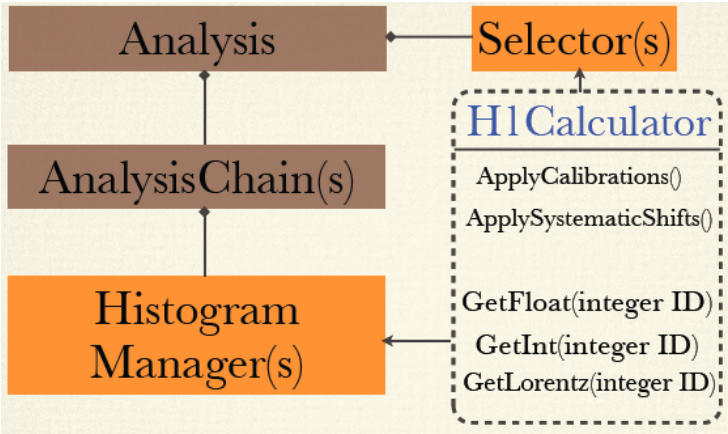

Figure 8. Analysis object model.

The development of a transient data interface improved physics analysis in many cases, especially those where access to the latest calibrations was critical and/or complicated systematic effects had to be evaluated. It paved the way for further developments which allowed for a more efficient exchange of higher-level physics analysis tools, up to and including entire analyses.

Acknowledgments

The material presented here is the culmination of many years of work by members of the H1 Collaboration. I'd like to thank all of those who worked on the H1OO project and in particular my friend and collaborator Dave South.

References

[1] Berthon U, Benisch T, Gerhards R, Grab C, Hadig T and Van Mechelen P, New data analysis environment in H1, *International Conference on Computing in High-Energy Physics and Nuclear Physics (CHEP 2000), Padova, Italy, 7-11 Feb 2000*

[2] Benisch T, Berthon U, Gerhards R, Grab C and Hadig T, New Data Storage Model For H1, *International Conference on Computing in High-Energy Physics and Nuclear Physics (CHEP 2000), Padova, Italy, 7-11 Feb 2000*

[3] Peez M [H1 Collaboration], The New object oriented analysis framework for H1, eConfC **0303241** (2003) THLT007 [physics/0306124].

[4] Katzy J, H1OO - an analysis framework for H1, *Computing in High-Energy Physics (CHEP '04), Interlaken, Switzerland, 27 Sep - 1 Oct 2004*

[5] Blobel V, BOS and related packages, *Erice 1990, Proceedings, Data structures for particle physics experiments, 1-6*

[6] Brun R, Couet O, Vandoni C E and Zanarini P, Paw: A General Purpose Portable Software Tool For Data Analysis And Presentation, *Comput. Phys. Commun.* **57** (1989) 432.

[7] Laycock P, A Measurement of the Diffractive Reduced Cross-Section $\sigma_r^{D(3)}$ at High Q^2 with the H1 Detector at HERA, PhD thesis (2003)

[8] South D, Events with Isolated Leptons and Missing Transverse Momentum in e+p Collisions at HERA, PhD thesis (2003)