User Defined Data in the New Analysis Model of the BaBar Experiment

Guglielmo De Nardo and Luca Lista, *representing the BaBar collaboration*

***Abstract*-- The BaBar experiment has recently revised its Analysis Model. One of the key ingredient of BaBar new Analysis Model is the support of the capability to add to the Event Store user defined data, which can be the output of complex computations performed at an advanced stage of a physics analysis, and are associated to analysis objects. In order to provide flexibility and extensibility with respect to object types, template generic programming has been adopted. In this way the model is non-intrusive with respect to reconstruction and analysis objects it manages, not requiring changes in their interfaces and implementations. Technological details are hidden as much as possible to the user, providing a simple interface. In this paper we present some of the limitations of the old model and how they are addressed by the new Analysis Model.**

I. Introduction

BABAR is a general purpose solenoidal detector operating at the PEP-II asymmetric B-Factory. The experiment has taken data since 1999 at and slightly below the Y(4S) resonance and has accumulated 140 fb^{-1} of integrated luminosity. The detector and the accelerator are described in detail elsewhere [1]-[2].

In April 2002 BaBar computing was reviewed. Among the various recommendations, the report of the committee suggested to reconsider the Analysis Model. In response to these recommendations, a new computing model has been proposed to the collaboration in December 2002.

This paper briefly describes the Analysis Model that has been very successfully used so far, the limitations that have been recognized, and the new model that has been adopted.

II. The Previous Analysis Model

The software is designed to abstract the persistency layer in order to decouple as much as possible the transient objects from the persistency implementation. This choice permits to avoid dependencies in transient code from the particular persistency implementation.

The primary persistency technology is based on Objectivity [3], a commercial product that had been recognized in a first stage as a reliable Object Oriented database management system, having all desirable capabilities needed even by the future experiments in the LHC era. This choice has been now reconsidered and the new general consensus among the high energy physics community is to use ROOT-IO based persistency, instead.

The data model consists of several layers of complementary event data formats, containing information about the reconstructed event at different levels of detail. The hierarchical organization and the contents of each data component have been determined on the basis of different use cases from detector studies to physics analysis. A detailed discussion of these topics is outside the scope of this paper and can be found in [4]-[5].

Almost every physics analysis is based on the Tag and Aod data formats only, the highest level ones. The former being used for fast filtering of the interesting events, the latter containing a summary of measurements performed by the detector subsystems for each of the stable particle candidate that has been reconstructed. The type of information stored varies from simple quantities like, for example, the number of layers traversed in central drift chamber, to more sophisticated derived quantities like the measured nuclear interaction length in the muon system or the shape parameters of the calorimeter clusters. On the basis of these analysis level objects the physics code of the OO BaBar software performs the particle identification of the stable particles and the reconstruction of intermediate, unstable (in the BaBar physics software jargon "composite") particles like neutral pions, K_S, charmonium resonances, charmed mesons, and so on, from a variety of exclusive decay modes.

The reconstruction of the composite particles is performed by means of a modular and configurable set of tools developed for this particular purpose. The usage of a common set of algorithms and the definition of a standard *modus operandi* ensure the robustness of the physics output. On the other side, the task of building the composite candidates from their hypothetical decay products is intrinsically very expensive in terms of CPU usage, because of the high number of combinations involved, especially in cases of high multiplicity decay modes or when a complex genealogy is reproduced.

Manuscript received October 29, 2003.

G. De Nardo was with the Istituto Nazionale di Fisica Nucleare and Universita' di Napoli, Napoli, Italy. He is now with the Stanford Linear Accelerator Center, Menlo Park CA 94025 (telephone +1 650-926-2495 email: denardo@slac.stanford.edu)

L.Lista is with the Istituto Nazionale di Fisca Nucleare, Napoli, Italy (telephone +39 081-67-6908 e-mail: lista@na.infn.it)

Presented at IEEE 2003 Nuclear Science Symposium and Medical Imaging Conference including Sessions on Nuclear Power Systems, Portland, OR 19-24 Oct 2003

Work supported in part by the Department of Energy contract DE-AC02-76SF00515

SLAC, Stanford University, Stanford, CA 94309

Therefore, the necessity of the execution of this CPU intensive code should be kept to minimum.

In general, after the construction of the composite particle candidates, selection criteria, aiming at high selection efficiency, are further applied in order to reduce background events and select only potentially interesting events.

In most of the cases, physics analyses does not run on every event of the entire data set but take advantage of an intermediate step of data reduction. This process is called event skimming in BaBar. In practice, all the physics events are processed centrally and events belonging to categories, determined on the basis of certain properties of the event, are copied to different database files, so that analysis working groups may run their application on reduced data sets (up to few percent of the original data sets). The persistency technologies supported are both Objectivity and a custom ROOT-IO based BaBar implementation [6].

Then the last step inevitably consists in the production of large PAW ntuples containing all the information which is considered to be useful for further analysis. Usually, analysis working groups take care of the ntuple production for their users, and store them in dedicated disk space. The data stored in the ntuples very often comprise the high level analysis dependent quantities and the composite particle candidate information calculated by physics analysis modules, as well as all the basic information already present in the original Aod component of the Event Store, *de facto* doubling the physics analysis storage needs.

Always, the rest of physics analysis is then performed directly on the ntuples, or a ROOT translation of them.

The ntuples contains all the output of the physics code like specific analysis dependent quantities, composite candidates with all the kinematical and vertex information added. Indeed, at the certain point of the analysis chain, the physics analyst is happy with the composite candidates he has and it is not likely he may want to introduce new candidates that have not been considered so far. Much of the effort is focused on the refinement of the existing candidates to enhance the signal events with respect to the background.

The most natural way to proceed would be to persist the composite candidates and then continue the analysis in the OO BaBar framework from there, but the event store lacks the functionality to persist the composite particle candidates, and the physics analysts' solution is to use PAW ntuples as an additional implementation of persistency.

The model has been so far very successful in producing high quality physics results. But experts think it will not scale well to the larger data set expected in a not so far future.

There are several limitations in this approach. Together with added data in the ntuple also most of the event store content is replicated. Every analysis working group stores its own heterogeneous hbook and/or ROOT files in dedicate disk areas, with an evidently inefficient storage use. The connection with more detailed components of the event store is not possible to recover. Users' code based on ntuple data format does not benefit anymore from the BaBar object-oriented software design. Moreover, every analysis has to replicate code, reinventing solutions already developed elsewhere for similar cases, with the result that a lot precious manpower is spent in debugging. Finally, because of the heterogeneity of the ntuples, analysis tools and strategies sharing among different working groups is a challenge.

III. THE NEW ANALYSIS MODEL REQUIREMENTS

One of the recommendations suggested by the BaBar Computing Model review committee was to replace Objectivity based persistency technology with a ROOT-IO based one. This has been done for primary event store. The database containing the detector conditions still depends on Objectivity at the present moment, and its migration to ROOT is foreseen next year. A second important change that has been completed is the revisiting of the internal representation of the event store, with the re-implementation the "Micro" part of the event store as a sub-set of "Mini" part. In the old model the two parts were just disjoint.

Besides these two very important changes, additional functionalities have been added to the Event Store in order to improve the Analysis Model in the light of the physics analysis requirements and the recognized limitations of the current model that have been elucidated in the previous chapter.

The key aspect is having more flexibility in the event store. This can be fulfilled by supporting the persistency of composite candidates and the addition of custom data.

The role of event skimming, that is the process by which interesting events belonging to various physics categories are pre-selected and copied in dedicated events collections, has been expanded.

An header component in the event store keeps track of the physical location of all the components of the event store. This allows the production of deep-copy skims, for which the all the components are copied to files, or of pointer skims, for which some of the components, containing the basic reconstruction data, are borrowed from the original data files, in the sense that only references to the original objects are stored instead of copies, while only components the contains added data are physically written to disk.

The deep-copying of the events, which is viable only for the less abundant event categories, provides the advantage of fast read-back and easy offsite portability. The second solution, of borrowing basic data while writing only the added part provides an efficient resource usage in terms of disk space.

A possible scenario is that the analysis working groups augment their data skims with high level physics quantities and particle candidates specific for the interests of the group. Then the physics analysts read the data skim produced for their working group make their own tighter event selections and produce their private skims with their own options of added data and composite candidates. Frequent central skim

productions, with a period of three months, is foreseen to assure prompt availability of improvements, bug fixes and new skims to the users.

The support to interactive access (via ROOT/CINT) directly to the ROOT-IO based event store, to be used as a data inspection tool, is also provided.

Since BaBar is a running experiment, the introduction of the new functionalities must not to interfere with existing physics software. For this reason attention has been paid to keep the same interfaces so that it is not necessary for the users to change their code, all the necessary setup being necessary only in analysis jobs configuration.

IV. User added data design and implementation

Two new components have been added to the event store.

A "cnd" candidates component contains the necessary information to build, in the read phase, the original transient reconstruction based candidates and composite candidates.

An "usr" User Data component contains generic user added data.

The persistent composite candidate attributes are the particle genealogy, the identification code of the fitted algorithm and of the identification code of the constraints that have been used to make the original composite candidate. When reading back all the persistent candidates objects are translated into transient objects redoing the fit. This design choice has been adopted to save disk space as much as possible. This has a cost in term of performances because every candidate is fitted again when it is read back. The loss of performances may be mitigated supporting dynamic data access, i.e. implementing the reconstruction on demand of the candidates, which is currently under development.

The other aspect is support storing user defined variables. The main design requirements are the following:

- Flexibility of the interface as well as uniformity has to be guaranteed.
- Support of basic types like float and integer, and their short and long versions have to be supported as well as complex objects like three-vector and four-vectors.
- User added data may variables referring to the whole event or they can be associated to particle candidates.
- Extend the functionalities following the Open/Closed object-oriented principle. In other words, extend the functionalities without affecting existing classes interface or implementations
- Simplicity of the user interface, leaving the technological details hidden as much as possible to the user.

The requirements have been fulfilled by the systematic use of generic programming. On the transient side the type of the variable has been kept generic by means of a C++ template parameter. The user added data may be associated to the event or associated to generic unique objects. Uniqueness is required in order to objects clones have the same associated user data.

The object identity is provided by a functor operating on the objects and returning an unique identifier. In the generic class that manages user data associations with objects the functor is a template as well. In summary the interface and the core functionalities have being defined for using generic variables type associable to generic objects. References to objects are maintained by means of unique identifier.

Nevertheless the advanced fully templated design, the user interface is extremely simple and uniform. Users have to deal with few concepts and classes. A class UsrVariable<T>, represents a variable of generic type T.

The generic user variables interface has been provided with all the required methods in order to use it as a regular variable. Users may instantiate UsrVariable<float> or UsrVariable<int> specializations and then use the variables as a regular floats or ints.

Blocks of variables aggregate the variable values and the associations with objects. An UsrEventBlock class aggregates variables associated to the whole event, while an UsrCandBlock class aggregates objects references and their associate variables. Two static get and put functions are provided to retrieve from or to save into the event store a block.

In practice, the user may save and get back from the event store custom quantities in the same fashion he is used to do when writing in ntuples, but with a much greater flexibility and usability.

V. Current Status

In December 2002 the design of the new Analysis Model has been finalized and the requirements have been set. The implementation has been developed setting various intermediate milestones during the first half of the 2003. Two tests productions on a limited data set have been completed in April and in July 2003. They have been used to test the development achievements at that point, as well as to validate the core functionalities in different physics use cases.

The persistent class structure is in its final shape so that the persistent schema has been frozen. A mechanism for schema evolution is supported in the ROOT-IO in the case of minor modifications in the future might be needed. Development will continue in several areas not related with core persistency implementation. In particular for the short term, efforts are ongoing for the implementation of the load on demand, the data inspection tools for interactive access, and for improvements in performances.

The systematic skim production for new data and conversion of the full data set and Monte Carlo to the new format will begin on November 1st. Full skim productions will follow with a period of three months. The first physics results

using the new model are expected to come in the summer 2004.

VI. Conclusions

The new Analysis Model of the BaBar experiment addresses the limitations of the old one, introducing a new data format for the Event Store, which beside a more efficient internal structure presents improved functionalities in terms of extensibility of the data contents and flexibility in usage, to fulfill the new requirements dictated by the current physics use cases and the expected resources available in the future.

The persistency technology for the primary event store is based on ROOT. A complete phase out of Objectivity implementation, including also the detector conditions database, is foreseen for the next year.

The new data format has been engineered to be compatible with interactive access directly from the ROOT-IO based Event Store by means of CINT.

The new system is, moreover, backward compatible with existing physics code, because the interfaces have been preserved and therefore the changes in data format are completely transparent to the users.

Flexibility and extensibility of the event store, together with a central role of the event skims production frees the analysis working groups from the need a large PAW ntuples productions, considerably reducing the amount of resources need in terms of CPU time, disk space and manpower.

The natural approach of the New Analysis Model consists in performing a progressive data reduction, with respect to number of events, and enrichment, with respect to data contents in the single event, by means of iterative application of the skimming procedure. This allows the final users to postpone the writing of their results in a private format (like the ntuples) only at the end of their analysis chain, taking fully advantage of the object oriented design of the BaBar experiment software.

Moreover, the new model favors common efforts among the working groups in the development of physics analysis tools and strategies.

VII. Acknowledgment

The authors wish to recognize the achievements of the handful of BaBar people who participated to the design and the implementation of the new Analysis Model of the experiment, succeeding in the terrific effort of deploying such a complex and delicate system in a very tight time schedule.

VIII. References

[1] BaBar Collaboration, B.Aubert et al., Nucl. Instrum. Methods A479, 1 (2002).

[2] *PEP-II: An Asymmetric B Factory*, Conceptual Design Report, SLAC-418, LBL-5379 (1993).

[3] Objectivity Inc., http://www.objectivity.com

[4] D. N. Brown, *The BaBar Mini*, Proceedings of CHEP 2003, hep-ex/0305085.

[5] A. Adesanya, J. Becla and D. Wang, *The Redesigned BaBar Event Store: Believe the Hype*, Proceedings of CHEP 2003, ePrint cs.DB/0306023.

[6] D. Kirkby, *KANGA(ROO): Handling the micro-DST of the BaBar experiment with ROOT*, presentation given at CHEP 2003.