Computing in High Energy and Nuclear Physics, 24-28 March 2003, La Jolla, California

Physics Analysis Expert PAX: First Applications

Martin Erdmann[1], Dominic Hirschbühl, Christopher Jung, Steffen Kappler[2], Yves Kemp, Matthias Kirsch, Deborah Miksat, Christian Piasecki, Günter Quast, Klaus Rabbertz, Patrick Schemitz, Alexander Schmidt, Thorsten Walter, Christian Weiser
Institut für Experimentelle Kernphysik, Universität Karlsruhe, Wolfgang-Gaede-Str. 1,
D-76131 Karlsruhe, Germany
[1]contact *Martin.Erdmann@cern.ch*
[2]also with *CERN, EP Division, CH-1211 Geneva 23, Switzerland*

PAX (Physics Analysis Expert) is a novel, C++ based toolkit designed to assist teams in particle physics data analysis issues. The core of PAX are event interpretation containers, holding relevant information about and possible interpretations of a physics event. Providing this new level of abstraction beyond the results of the detector reconstruction programs, PAX facilitates the buildup and use of modern analysis factories. Class structure and user command syntax of PAX are set up to support expert teams as well as newcomers in preparing for the challenges expected to arise in the data analysis at future hadron colliders.

1. Motivation

Working directly on the output of detector reconstruction programs when performing data analyses is an established habit amongst particle physicists. Nevertheless, at the experiments of HERA and LEP it turned out to be an advantage to have uniform access to calorimeter energy depositions, tracks, electrons, muons etc. which requires a new level of abstraction on top of the reconstruction layer. Examples of physics analysis packages providing this level are H1PHAN[1] and ALPHA[2] of the H1 and ALEPH experiments. Noticed effects were, amongst others, that
a) users could relatively quickly answer physics questions,
b) the physics analysis code was protected against changes in the detector reconstruction layer,
c) and finally the management liked the fact, that the relevant reconstruction output had been used in the analysis.

While previous programs were designed to provide a rather complete view of the event originating from a single e^+e^- or ep scattering, a next generation package is challenged by hadron collisions with $O(20)$ simultaneous events. This implies a large number of possible interpretations of the triggered events and sometimes requires the analysis of dedicated regions of interest. The "Physics Analysis Expert" toolkit (PAX) is a data analysis utility designed to assist physicists in these tasks in the phase between detector reconstruction and physics interpretation of an event (Fig.1). The alpha-version of PAX was presented at the HCP2002 conference [1]. In this contribution we introduce the beta-version.

2. Guidelines for the PAX Design

The design of the next generation physics analysis utility PAX has been developed according to the guidelines listed below:

1. The package is a utility tool box in a sense that the user has full control of every step in the program execution.
2. The programming interface should be as simple and intuitive as possible. This minimizes the need to access the manual and thereby increases the acceptance in the community. Furthermore, simplicity enables also physicists with limited time budget or limited knowledge of object oriented programming to carry out complex physics analyses.
3. The package supports modular physics analysis structures and thus facilitates team work. The complexity of todays and future analyses makes team work of many physicists mandatory.
4. Existing physics analysis utilities can be connected. Examples are tools for fourvector gymnastics which are available in general form (e.g. in the CLHEP library[3]), other examples are histograms, fitting routines etc.
5. The physics analysis package can be used consistently among different high energy physics experiments.
6. Many use cases are to be taken care of. The following list is certainly not complete:

---

[1] H1 Collaboration, internal software manual for H1PHAN

[2] ALEPH Collaboration, "ALPHA" internal note 99-087 SOFTWR 99-001

[3] CLHEP, A Class Library for High Energy Physics, *http://proj-clhep.web.cern.ch/proj-clhep/*

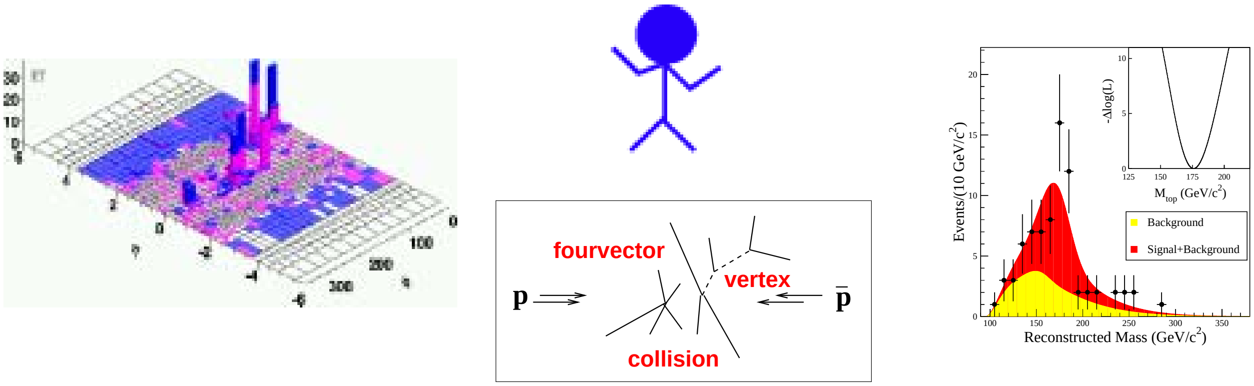

Figure 1: Application area of the PAX toolkit: in between detector reconstruction output and physics interpretation of an event [2].

(a) Access to the original data of the experiment is possible at each stage of the physics analysis. This enables detailed control of all analysis steps, and access to experiment-dependent information and related methods.

(b) When studying events of a Monte Carlo generator, relations between generated and reconstructed observables are accessible at any stage of the analysis. This allows the quality of the detector reconstruction to be studied in detail.

(c) Without significant code changes, a complete analysis chain can be tested with different input objects such as reconstructed tracks, generated particles, fast simulated particles etc.

(d) Relations between reconstructed physics objects (tracks, muons, etc.) and vertices are available, as well as hooks for separating multiple interactions.

(e) The decay chains with secondary, tertiary etc. vertices can be handled in events with multiple interactions.

(f) Information of different objects can be combined, e.g., tracks and calorimeter information.

(g) A common challenge in data analysis are reconstruction ambiguities which need to be handled. Administration of these ambiguities is supported.

(h) The user finds assistance in developing analysis factories with multiple physics data analyses carried out simultaneously.

To cope with these challenges, the advantage of using an object oriented language is obvious. For the convenience of connecting to other packages, C++ was the language of choice for the realisation of PAX.

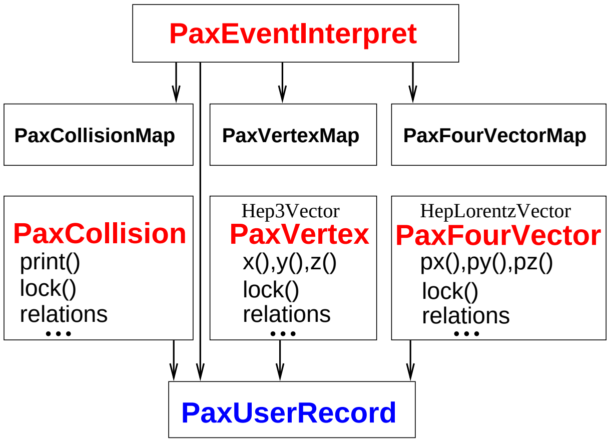

Figure 2: The basic unit in PAX: the event interpretation together with the classes for collisions, vertices, fourvectors, and user records.

3. PAX Class Structure Implementation

3.1. Event Interpretation

The basic unit in PAX is a view of the event which we call "event interpretation". The event interpretation is a container used to store the relevant information about the event in terms of collisions, vertices, fourvectors, their relations, and additional values needed in the user's analysis. The corresponding class is denoted *PaxEventInterpret* (Fig.2). The user books, fills, draws, copies, advances the event interpretation, and has the ultimate responsibility for deleting it. When the user finally deletes an instance of *PaxEventInterpret*, instances of objects which have been registered with this event interpretation – collisions, vertices, fourvectors, etc. – are also removed from memory.

A copy of an event interpretation is a physical copy in memory. It is generated preserving all values of the original event interpretation, and with all relations between collisions, vertices, and fourvectors corrected to remain within the copied event interpretation. In addition, the histories of the individual collisions, vertices, and fourvectors are recorded. The copy functionality simplifies producing several similar event interpretations. This is advantageous typically in the case of many possible views of the event that differ in a few aspects only. Although the event interpretations do not know of each others existence, recording the analysis history of collisions, vertices and fourvectors requires intermediate event interpretations to exist. Therefore, we recommend to delete all event interpretations together after an event has been analysed.

Besides the features mentioned, the *PaxEventInterpret* also defines an interface to algorithms such as jet algorithms, missing transverse energy calculations etc. This eases the exchange of algorithms within, or between analysis teams.

3.2. Physics Quantities

PAX supports three physics quantities: collisions, vertices, and fourvectors. Three classes have been defined correspondingly. The class *PaxCollision* provides the hooks to handle multi-collision events (Fig.2). Vertices and fourvectors are defined through the classes *PaxVertex*, and *PaxFourVector*. Since the user may need to impose vector operations on them, both PAX classes inherit from the CLHEP classes *Hep3Vector*, and *HepLorentzVector*, respectively. Their functionalities are available to the user. The *PaxVertex* and *PaxFourVector* classes contain additional functionality which mainly result from features proven to be useful in the previously mentioned H1PHAN package.

For all physics quantities the user can store additional values (data type *double*) needed by the analysis via the class *PaxUserRecord*. These values are registered together with a key (data type *string* functioning as a name), which must be given by the user.

3.3. Relation Management for the Physics Quantities

Relations between collisions, vertices, and fourvectors are handled by a separate class called *PaxRelationManager* (Fig.3). Here we followed the design pattern *Mediator* [3]. Examples for relations to be handled between physics quantities are fourvectors which originate from the primary vertex, an incoming fourvector to a secondary vertex, or connections between multiple collisions and their vertices etc.

When physics quantities are copied, the copied instance carries a pointer to the previous instance. An example would be a fourvector which is copied together with an event interpretation. In this way, the full history of the fourvector is kept throughout the analysis.

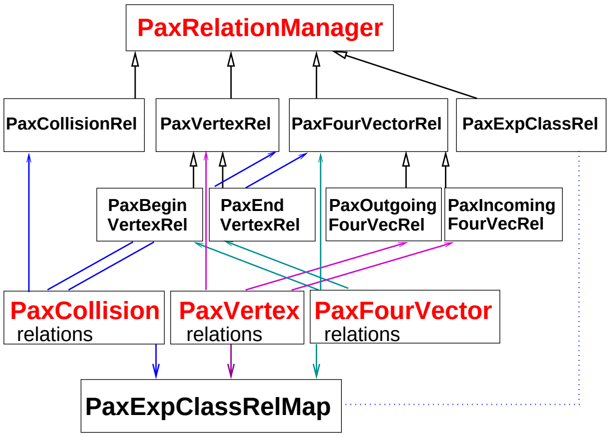

Figure 3: The relations in PAX enable storage of decay trees, records of analysis history, access to experiment specific classes, and exclusion of parts of the event from the analysis.

The relation manager also allows parts of the event to be excluded from the analysis. An example would be a lepton which needs to be preserved while applying a jet algorithm. This locking mechanism is build in the form of a tree structure which enables sophisticated exclusion of unwanted event parts. For example, locking a collision excludes the vertices and fourvectors connected to this collision (Fig.4). In the case of locking a secondary vertex, PAX will lock the decay tree starting at this vertex.

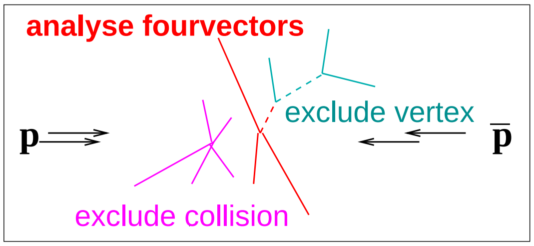

Figure 4: Excluding a collision (left) or a vertex (right) from an analysis using the lock mechanism excludes all vertices and fourvectors originating from the excluded object.

Note that when locking a fourvector f, all fourvectors related to f through its history record are locked as well. The same functionality applies to collisions and vertices. Unlocking the fourvector f removes the lock flag of all physics quantities related to f through the decay tree, or the history record at the time the unlock command is executed.

For some applications, the user may want to inquire additional information on a physics quantity which is only contained in an experiment specific class. An example is a *PaxFourVector* instance originating from a track of which the user wants to check the χ^2 probability of the track fit. The relation manager allows to register instances of experiment specific classes which led to a *PaxCollision*, *PaxVertex*, or *PaxFourVector* instance. To enable such relations, a template class *PaxExperiment*<> has been defined which allows registration of arbitrary class instances. Applying the C++ *dynamic_cast* operator, the user can recover the original instance, and access all member functions of the experiment specific class.

3.4. Container and Iterator

PAX uses the template class *map*<> from the Standard Template Library (STL) [4] to manage pairs of keys and items in a container. The user record in Fig.2 is an example of such a container for pairs of data type *string* and *double*. For accessing a certain item, optimized STL algorithms search the *map* for the corresponding key and provide access to the item.

All *PaxCollision*, *PaxVertex*, and *PaxFourVector* instances carry a unique identifier of type *PaxId* which is used as the key in the *PaxCollisionMap*, *PaxVertexMap*, and *PaxFourVectorMap* of an event interpretation (Fig.2). Pointers to the collision, vertex, and fourvector instances are the corresponding items. In this way, fast and uniform access to the individual physics quantities is guaranteed.

For users not familiar with STL iterators, we provide the *PaxIterator* class which gives a simple and unified command syntax for accessing all containers in PAX.

3.5. Documentation

The PAX user guide is available on the web [5]. In addition to a paper version of the manual, we provide a fast navigator web page which can be used as a reference guide.

4. Application within Physics Analysis of the CDF Experiment

The PAX package is explored by the Karlsruhe CDF group in top quark analyses. Example Feynman diagrams of signal and background processes relevant to top analysis in the so-called electron plus jet channel are shown in Fig.5.

4.1. Combining Results of the Detector Reconstruction Program

The CDF detector reconstruction program provides already excellent reconstruction algorithms for the calorimeter, track finding, electron identification etc.

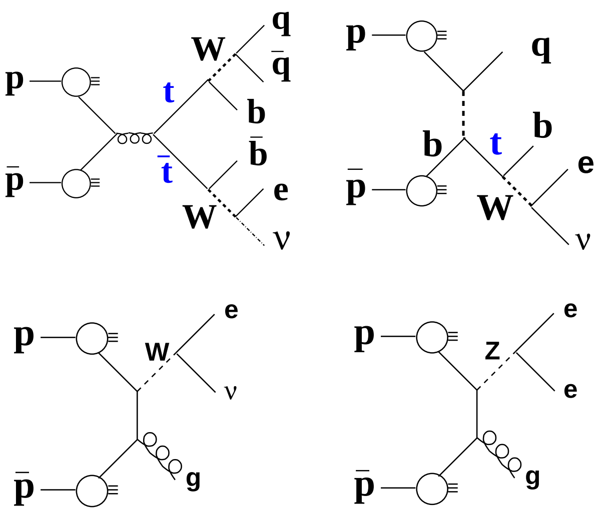

Figure 5: Example Feynman diagrams relevant to top quark analysis in proton-antiproton collisions.

To further advance these results we fill them into instances of the *PaxEventInterpret* class (Fig.2). Examples are the calorimeter energy measurements, the tracking output, jet searches, and electron, muon, and photon identification. For the graphical representation of the different event interpretations in Fig.6 we use the ROOT package [6]. The lines indicate the direction of the fourvectors, and the lengths correspond to their energies. In order to optimize the energy measurements and take into account the advanced particle identification algorithms of the reconstruction software, we combine different results into a single event interpretation.

While combining the calorimeter output with the electrons is relatively straight forward, an algorithm to combine the calorimeter and track information needs to treat the energy which is measured in both subdetectors in order to avoid double counting of energy. In Fig.6 the quality of our combined energy measurement is tested. Using $t\bar{t}$ events of the Herwig Monte Carlo generator [7], the histograms vertical axis shows the reconstructed transverse energy sum as a function of the true transverse energy sum. The latter was determined from the generated hadrons and leptons, excluding neutrinos. The algorithm provides a good measurement of the event total transverse energy.

4.2. Top Quark Analysis Factory

To optimize separation of signal from background processes, we set up an “analysis factory” based on the PAX event interpretation concept. In the top quark factory, every event is examined with respect to different processes which include electroweak and strong production of top quarks, as well as W- and

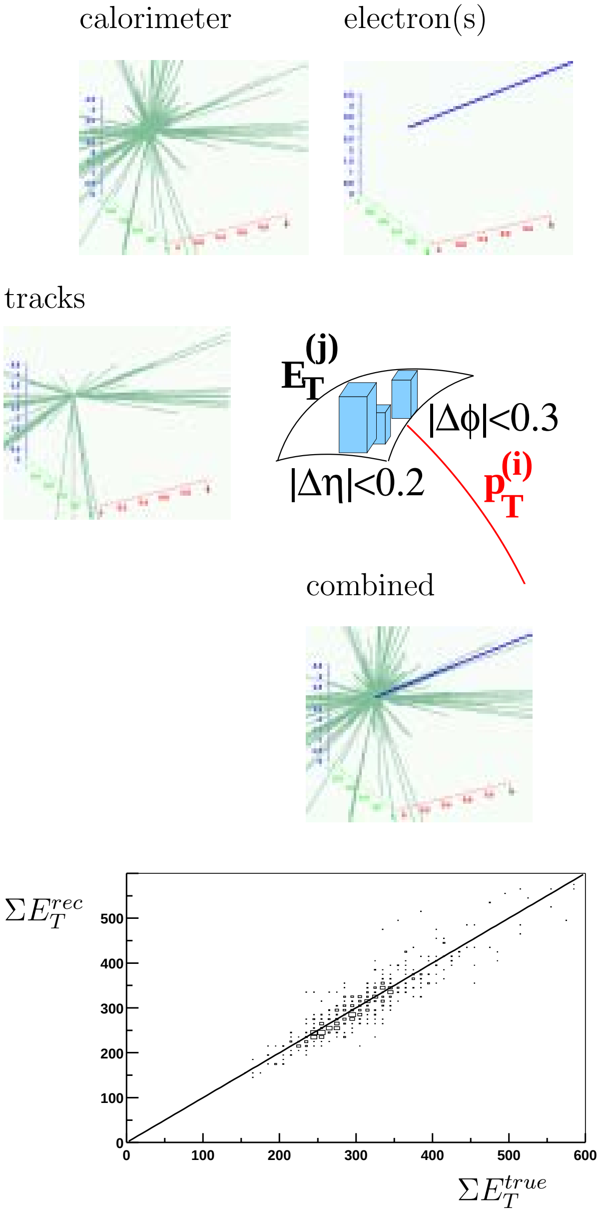

Figure 6: Combining different output of the detector reconstruction program results in a good reconstruction quality of the transverse energy in $t\bar{t}$ events of the Herwig Monte Carlo generator.

Z-production (Fig.5). We attempt a complete reconstruction of the partonic scattering process.

Some aspects of the event analysis of a $t\bar{t}$ event are demonstrated in Fig.7. The first picture shows the situation after applying a jet finding algorithm to the combined reconstruction information shown in Fig.6. The electron candidate has been preserved using the locking mechanism. The lines indicate the fourvectors of the 4 jets, the electron, and one fourvector which includes all remaining unclustered energy depositions.

In the second row, a W-boson decaying into an electron and a neutrino is reconstructed. From the W-mass constraint two possible solutions can be deduced for the longitudinal neutrino momentum which correspond to two W event interpretations. Combining the W with one of the jets leads to top quark solutions, two of which are shown in the bottom row of Fig.7. The reconstructed top quark candidates point into different directions. In this four jet event, 24 $t\bar{t}$ solutions can be constructed. Although the number of remaining fourvectors is relatively small, the full information of the original O(1000) calorimeter energy depositions, tracks etc. can still be accessed from each of the event interpretations.

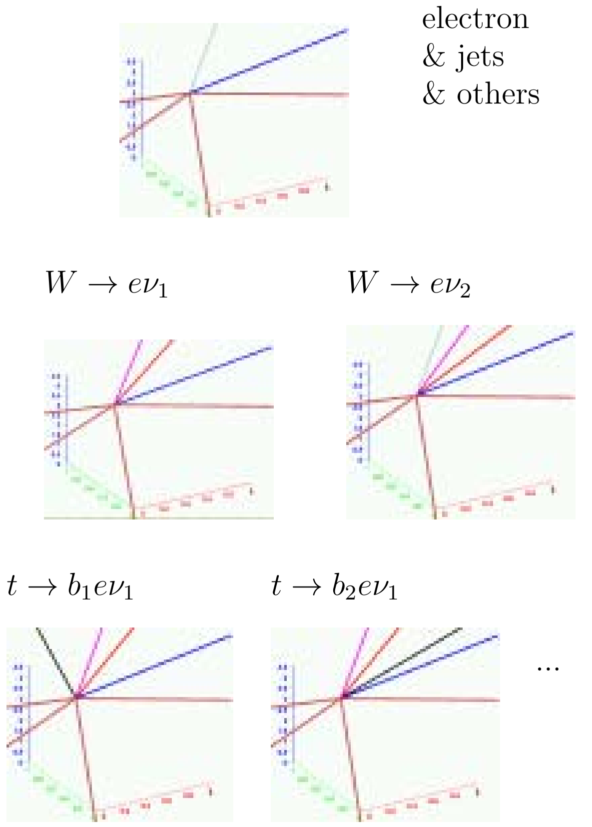

Figure 7: Full reconstruction of the partonic scattering process of a Herwig $t\bar{t}$ event.

We select the most likely $t\bar{t}$ event interpretation by first demanding a non-zero bottom quark probability for one of the jets of the top candidate. Fig.8a shows that the number of remaining event interpretations is still relatively large. We select one of these solutions by using a simple χ^2 test on the reconstructed W-boson and top quark masses. In Fig.8b, the reconstructed mass of the top quark in the electron plus jet decay channel is shown for all events (histogram). The symbols indicate the number of events in which the correct top quark candidate was found.

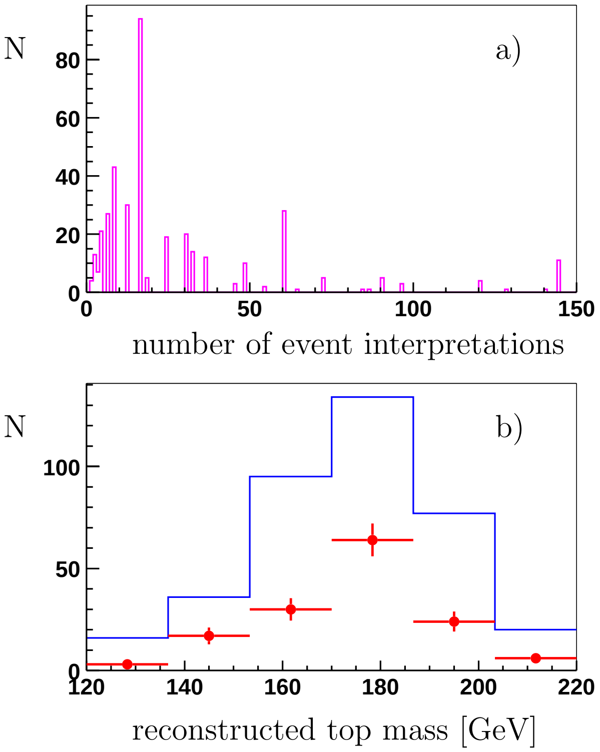

Figure 8: a) Multiplicity of event interpretations, and b) reconstructed mass of the top quark decaying into a bottom quark and a W-boson which subsequently decays into an electron and a neutrino. The symbols represent the events where the selected event interpretation was the correct one.

5. Application within Physics Analysis of the CMS Experiment

The Karlsruhe CMS group uses PAX within Higgs search studies. The applications shown here are benchmark tests where the Higgs boson decays into ZZ* with subsequent decays into four muons (Fig.9). We simulated this process with the Monte Carlo generator PYTHIA [8], assuming a hypothetical Higgs mass of $m_H = 130$ GeV. As background process we considered Z$b\bar{b}$ production, which we generated using COMPHEP [9] followed by the LUND string fragmentation model within PYTHIA.

Whereas in the Z$b\bar{b}$ process the muons result from the Z boson and two bottom quark jets, in Higgs events the four muons are the decay products of the Z and Z*. Thus, to reconstruct the Higgs, all muons of a generated event are filled into an event interpretation and, with the help of a likelihood method, a Z and a Z* are reconstructed. Combining of Z and Z* then results in the Higgs mass spectrum, shown in Fig.10a.

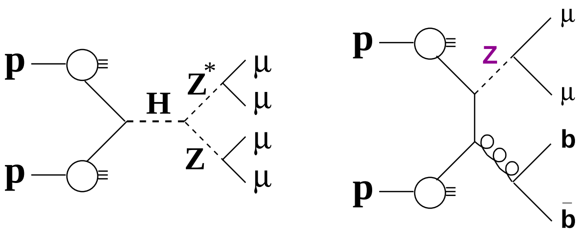

Figure 9: Example Feynman diagrams of Higgs production and the Z$b\bar{b}$ background process.

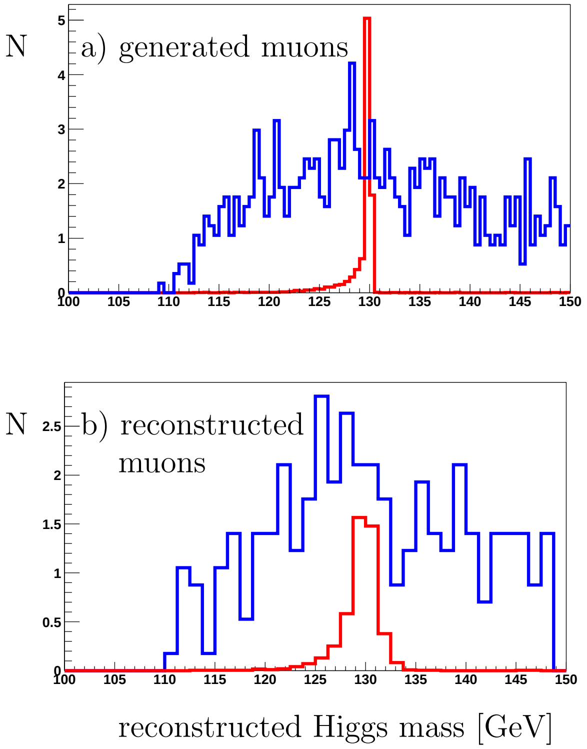

Figure 10: Reconstructed Higgs mass for an integrated luminosity of 20 fb^{-1} using a) generated and b) reconstructed muons of generated Higgs signal events and Z$b\bar{b}$ background events.

The same analysis has been used in a full simulation study, where the detector response was simulated with CMSIM[4] and the muons reconstructed with the CMS reconstruction software ORCA[5]. As shown in Fig.10b, the quality of the reconstructed Higgs mass spectrum is still satisfactory. Please note that both results – based on generated and reconstructed muons – were obtained by using the identical analysis code. This is easily possible due to the new level of abstraction which is provided by the PAX toolkit.

---

[4] CMSIM – CMS Simulation and Reconstruction Package, *http://cmsdoc.cern.ch/cmsim/cmsim.html*

[5] ORCA – Object-oriented Reconstruction for CMS Analysis, *http://cmsdoc.cern.ch/orca/*

6. Conclusion

The experiences gained at HERA and LEP show the advantages of performing physics analyses not directly on the output of detector reconstruction software, but using a new level of abstraction with uniform access to reconstructed objects. This level is provided by the presented data analysis toolkit PAX (Physics Analysis Expert). The design of PAX was guided by the experience of earlier experiments together with the demands arising in data analyses at future hadron colliders. Implemented in the C++ programming language, PAX provides a simple and intuitive programming interface to work within experiment specific C++ frameworks, but also on Monte Carlo generators. Event interpretation containers hold the relevant information about the event in terms of collisions, vertices, fourvectors, their relations, and additional values needed in the analysis. This enables the user to keep different interpretations of one event simultaneously and advance these in various directions. As PAX supports modular analysis and even the buildup of analysis factories, it facilitates team work of many physicists. PAX is suited for expert teams, physicists with limited time budget for data analyses, as well as newcomers. Groups within the experiments CDF and CMS are using PAX successfully for their analyses.

Acknowledgments

We wish to thank the German Bundesministerium für Bildung und Forschung BMBF for financial support. One of us, M.E., wishes to thank the organizers for a very interesting and pleasant conference.

References

[1] M. Erdmann et al., “Physics Analysis Expert”, Proceedings of the 14th Topical Conference on Hadron Collider Physics, HCP2002, Karlsruhe, Germany (2002)

[2] CDF Collaboration, F. Abe et al., Phys. Rev. Lett. 80 (1998) 2767

[3] E. Gamma et al., “Design Patterns”, Addison Wesley, ISBN 0-201-63361-2 (1994)

[4] B. Stroustrup, “The C++ Programming Language”, Addison Wesley, ISBN 0-201-88954-2 (1997)

[5] M. Erdmann et al., “Physics Analysis eXpert Users Guide”, IEKP-KA/2003-11 (2003) *http://www-ekp.physik.uni-karlsruhe.de/ ∼pax*

[6] R. Brun et al., “ROOT, an object oriented data analysis framework”, Proceedings of the 23rd CERN School of Computing , Marathon, Greece (2000)

[7] G. Corcella et al., JHEP 0101 (2001) 010, hep-ph/0201201 (2002)

[8] T. Sjöstrand et al., Computer Phys. Commun. 135 (2001) 238

[9] A. Pukhov et al., Preprint INP MSU 98-41/542, hep-ph/9908288, and *http://theory.sinp.msu.ru/comphep/*