Tools for Physics Analysis in CMS

Andreas Hinzmann
RWTH Aachen University, Physikalisches Institut 3A, Otto-Blumenthal-Straße, 52056 Aachen
E-mail: hinzmann@physik.rwth-aachen.de

Abstract. The CMS Physics Analysis Toolkit (PAT) is presented. The PAT is a high-level analysis layer enabling the development of common analysis efforts across and within physics analysis groups. It aims at fulfilling the needs of most CMS analyses, providing both ease-of-use for the beginner and flexibility for the advanced user. The main PAT concepts are described in detail and some examples from realistic physics analyses are given.

1. Introduction
The Large Hadron Collider was restarted in November 2009 and has already reached an instantaneous luminosity of $10^{32}$ cm$^{-2}$s$^{-1}$. The CMS experiment[1] was already provided with millions of collisions per second, which needed to be triggered, detected, stored and analyzed in a collaboration of several thousand physicists. This huge amount of data and the complexity of the detector require a flexible data model that serves all needs of the collaboration. Therefore data are provided in different data tiers[2]. The most important ones for the user are RECO and AOD. While RECO contains a complete set of analysis relevant reconstructed object information, the Analysis Object Data (AOD) is a subset of RECO, optimized to provide information sufficient for the majority of physics analyses in CMS at a small size around 140 kb/event[3]. The corresponding data format of both of them is optimized for performance and flexibility of the reconstruction but is difficult to access for the end user’s analysis. Here the Physics Analysis Toolkit (PAT) comes into play, providing common analysis data formats and common analysis algorithms[4]. The common data formats avoid the adoption of different data formats by the various working groups, substitutes flat n-tuple approaches and simplifies controlling the event size. It facilitates transfer and comparison of analyses by common skims within physics analysis groups. Further, PAT provides physics analysis groups with a common interface to the algorithms developed by physics objects groups and defines a single entry point to information associated to physics objects. Approved algorithms and sensible defaults allow a quick start into analysis for beginners and result in synergy effects so that everybody can profit from recent developments.

2. The CMS Event Data Model
The CMS Event Data Model (EDM) is based on independent C++ plugin modules which get executed according to python configuration files. The plugins have the ability to read from and write to a flexible event content[5,6] as shown in Figure 1. The data format is based on ROOT object storage. Therefore basic reconstruction objects, for example hits or energy deposits in the tracker, are stored in independent TTrees and TBranches[7]. Other modules combine
Figure 1. The EDM event content: Independent C++ plugin modules (EDAnalyzer, EDProducer, EDFilter) can read from or write to the event content. They combine basic information to higher level analysis objects, such as photons, and store them as new collections to the event content. The basic reconstruction objects, such as energy deposits, are still referred to via persistent references.

this basic information to form more complex analysis objects. For instance hits in the tracker get combined to form tracks using different algorithms or different configurations of the same algorithm. Within this combined analysis object the basic objects are referred to via references (see Figure 1). This minimizes redundancy in persistent information. High-level analysis objects, such as electrons, muons or jets are reconstructed in several independent steps each of them represented by a different module. In this way each reconstruction step is replaceable by a different module exploiting an alternative algorithm or just a different configuration of the same algorithm. The resulting object collections may be stored in parallel in the event content. Object collections may be dropped from the event content at any state of data processing, during the reconstruction of data as well as during further physics analysis steps. A data provenance instance keeps track of all utilized modules and their relevant configuration parameters that lead to the reconstruction of each object[8]. In this way, even after several skimming and replication steps, a clear identification of events and classification of objects can be guaranteed.

In summary, the CMS Event Data Model is optimized to achieve the following[9]:

- flexibility of the reconstruction of high level analysis objects.
- traceability and reproducibility of each reconstruction step for each high-level analysis object.
- minimal redundancy of persistent event information.

3. The PAT interface to physics objects
Due to the optimization of the EDM for performance and flexibility of the reconstruction, the information in RECO/AOD is difficult to access for the end user’s analysis. Information directly related to a physics object (e.g. a jet) is often stored in different collections (e.g. b-tagging information for jets) with no direct reference. PAT facilitates access to all analysis-relevant data for the user via common data formats that serve as single entry points to all information associated to the physics objects. New data formats for high-level analysis objects like photons and jets are introduced. Another example is the pat::TriggerEvent which gathers information which is difficult to retrieve since it is scattered across many different collections in the event content as well as collections stored per luminosity block (a fixed timespan of data taking). Objects of these types are produced consistently after the reconstruction step using C++ plugin modules that can be fully configured using python-based configuration files. This additional
analysis layer allows the user to select, gather and combine all information relevant to the analysis to each of the high-level reconstruction objects.

Figure 2. PAT provides common data formats which combine information associated to a physics object.

Figure 2 illustrates the information that might be included typically:

- object isolation in various definitions and detector components,
- object identification (like cluster shape information for photons or electrons),
- jet energy correction factors, object resolutions or reconstruction efficiencies,
- tracks associated to a jet, jet charge or jet flavour information,
- b-tagging information for jets,
- matched generator or trigger object information,
- user defined information in C++ built-in or vector types via the definition of user functions.

Therefore via these data formats PAT provides the physics analysis groups with easy access to the output of the algorithms developed by physics objects groups, e.g. b-tagging information.

4. The PAT data format

Given the limited disk space available on local computers, it is obviously desirable to greatly reduce the size of information output from reconstruction and subsequently read during analysis. However dropping object collections implies the danger of invalid references due to the RECO/AOD event content which is optimized for minimal redundancy by making use of reference pointer arithmetic. For instance dropping space extensive collections, such as energy deposits in the calorimeters, in order to reduce disk space requirements will turn all pointer relations to these objects from within higher reconstruction objects, such as photons, invalid at the same time. Thus, it might easily lead to unwanted loss of information.

PAT addresses this problem by storing the information of interest to the higher reconstruction objects, e.g. only the energy deposits out of which the photon got reconstructed, in the PAT data formats, in so-called pat-tuples, which aim to replace common n-tuples used in former experiments. In contrast to common n-tuples, the PAT data formats profit from all advantages of the EDM format, such as provenance information. Using pat-tuples may lead to a great reduction in event size without losing information of relevance for the physics analysis.

The pat-tuples with common data formats and objects, which can be configured with well-defined parameters, facilitate transfer and comparison of analyses. Analysis groups can set up data skims with a common tool set and therefore easily exchange information. In this way, saving computing and storage resources becomes possible and the comparison of different analyses of the same physics processes is simplified. The provenance information enables an exact understanding and comparison of each algorithms and parameter used in an analysis. The proposed user analysis cycle is to create and configure a pat-tuple from a RECO/AOD input file.
that is stored at one of the Tier-2 or Tier-3 centers so that they contain all relevant information for the given analysis and to perform fast analyses on these.

A flexible design is chosen to control the event size for the pat-tuples. As shown in Figure 3, one can either save the additional information inside the object or refer to the information via a reference to the original collection. Hard copying the useful information from other collections inside the object, the so-called embedding, allows these collections to be dropped which helps to reduce the event size because only a subset of objects are referenced. Whereas saving the information by reference leaves the full information of the collection accessible. The two ways of accessing the data during later analysis are provided by the same access functions at any time.

**Figure 3.** Example showing the embedding of track information inside the muon object. In RECO format, reco::Muons reference to their corresponding track in the reco::Track collection, whereas in PAT the track can be embedded in the pat::Muon itself and the reco::Track collection can be dropped.

This fully transparent way to drop information allows effectively to reduce the event size from typically 500(140) kB/event in RECO(AOD) format to typically less than 16 kB/event for a typical top quark analysis dataset. At the same time a decrease in the time required to access the reconstructed event information is achieved as well as the goal of preserving event provenance.

5. **The PAT algorithms**

PAT provides a set of approved algorithms and standard configuration files for common analysis tasks. These include tools for

- matching (e.g. trigger objects, generator particles)
- object combination to composed objects (e.g. object pairs to J/ψ)
- object isolation (e.g. by tracker or calorimeter information)
- object selection (e.g. by momentum, id, isolation)
- object disambiguation (e.g. by seeds or by ∆R requirements)
- tag and probe methods
- and others

Supplying this collection of approved algorithms with sensible defaults, everybody can profit from recent developments.
6. The PAT configuration

PAT is set up to be configured to the specific user’s needs. This includes that object parameters need to be adapted accordingly. This can be done in the configuration file using parameters that are described in detail in documentation maintained on wiki pages. Among others one can define object parameters, choose whether to embed information and decide if one likes to make use of the matching or cleaning tools that PAT provides. Sensible defaults are chosen in case the user is not interested to make use of the full potential of PAT. A user-friendly concept to perform a clearly arranged object selection is specifying cuts as strings in the configuration file[10]. All member functions, including those of referenced objects, are applicable. For example, the module `selectedPatJets` which creates a subset of the `patJets` collection is configured using a string parameter `cut` to specify the selection criteria:

```
selectedPatJets.cut = "pt>30"
```

Complex and common work flow adjustments can be coded into Python functions (called PAT tools), so that users need only to extend their configuration file with few simple lines to profit from those common tools. For example, the creation of the `patJets` collection, which involves retrieving of jet correction parameters and extra information such as b-tagging, can be configured using a single call of a tool:

```
switchJetCollection(
    process, # process in which the PAT configuration is defined.
    cms.InputTag('ak5CaloJets'), # label of the input jet collection.
    doJTA = True, # do a jet track association and determine the jet charge.
    doBTagging = True, # contract b tag information into the pat::Jet collection # or not.
    jetCorrLabel = ('AK5Calo', ['L2Relative', 'L3Absolute']), # correction labels # as a pair of a string and a vector of strings of the type.
    doType1MET = True, # use the specified jet energy scale corrections to perform # METType1 corrections from them.
    genJetCollection = cms.InputTag("ak5GenJets"), # generator level jet collection # to be used for generator matching as edm::InputTag.
    doJetID = True, # embed jetId variables to the pat::Jet collection.
    jetIdLabel = "ak5" # prefix of the value map of the jet id variables to be # embedded to the pat::Jet.
)
```

An editor with a graphical user interface (ConfigEditor) was developed to facilitate the creation of configuration files for the convenience of physicists [11,12].

7. Use cases in physics analysis

The PAT is widely used in CMS analyses from many fields. Two examples making use of different features of the PAT are the top physics analysis group and the SUSY physics analysis group.

In particular the top group has a large number of subgroups with analyses and crosscheck analyses in different channels contributing to the same final measurement. Therefore the top group has converged on the common object interface of PAT and a common set of algorithms in PAT and its top quark analysis specific extension, called Top Quark Analysis Framework (TQAF). Common selection criteria are defined via the PAT object interface allowing for continuous synchronization and comparison of the different analyses.

The SUSY group hosts analyses using a variety of group specific object and observable definitions. In order to keep track of this large number of object and observable definitions, they have decided to create a common PAT configuration defining all necessary objects, observables
and selection criteria used in the group. They provide recipes on how to create group wide analysis PAT tuples containing the necessary set of objects and observables for specific analyses in a common dataformat.

8. Summary
The PAT is a completely configurable tool set providing common data formats and algorithms, and serves as single entry point for all kinds of analyses in CMS. It is designed for the creation of small-size analysis specific datasets within the EDM framework, taking advantage of provenance information and facilitating interchangeability of analyses. Since a period of 2 years, regular tutorials are held to spread the knowledge on these common tools[13], which resulted in wide use during the 2009–2010 data taking period in analyses in CMS, making them transparent and comparable.

References
[1] CMS Collaboration (2008): The CMS experiment at the CERN LHC. JINST 0803:S08004, pp361
[2] CMS Collaboration (2004): The CMS computing model, CERN-LHCC-2004-035/G-083
[3] CMS Collaboration (2006): CMS Physics TDR, Volume 1, CERN-LHCC-2006-001
[4] Y. Küessel (2010): PAT: The CMS Physics Analysis Toolkit, Proc. XXX. Physics in Collision 2010 (PIC 2010), Karlsruhe, Germany
[5] CMS Collaboration (2005): CMS Computing TDR, CERN-LHCC-2005-023
[6] CMS Collaboration (2005): LCG Computing TDR, CERN-LHCC-2005-024
[7] R. Brun, F. Rademakers (1996): ROOT - An Object Oriented Data Analysis Framework, Proc. ALHENP’ 96 Workshop, see also root.cern.ch
[8] C.D. Jones et al (2009): File Level Provenance Tracking in CMS, Proc. 17th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2009), Prague, Czech Republic, J. Phys.: Conf. Ser. 219 032011
[9] R. Wolf (2009): A Tour Through the CMS Data Model, Proc. Ninth International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2009), Rhodes, Greece
[10] C.D. Jones, L. Lista and G. Petrucciani (2009), Expression and cut parser for CMS event. Proc. 17th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2009), Prague, Czech Republic, J. Phys.: Conf. Ser. 219 032050
[11] A. Hinzmann (2009): Visualization of the CMS Python Configuration System, Proc. 17th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2009), Prague, Czech Republic, J. Phys.: Conf. Ser. 219 042008
[12] A. de Cosa (2010): CMS Configuration Editor: GUI based application for user analysis job, these proceedings
[13] K. Lassila-Perini (2010): Planning and organization of an e-learning training program on the analysis software in CMS, these proceedings