MCDB and HepML, an Approach to Automation of Monte Carlo Simulation in HEP

S. Belov[1], L. Dudko[2], D. Kekelidze[1]

e-mail: Sergey.Belov@jinr.ru

[1]Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna, Russia

[2]Scobeltsyn Institute of Nuclear Physics of Lomonosov Moscow State University, Moscow, Russia

Introduction

Correct Monte Carlo simulation of complex processes requires a rather sophisticated expertise, and it also could be very resource-intensive. Simulation chain usually involves several independent program packages like Monte Carlo generators. So the information and it's detailed description should be transferred carefully through the links of this chain. Here we present a way of automation of Monte Carlo simulation in high energy physics and further use of simulated events. The tools used are HepML, dedicated XML language to describe simulated events in a pretty detailed and extensible way, and Monte Carlo Data Base (MCDB), the dedicated place to store and document Monte Carlo event samples. HepML language and corresponding libraries allow to produce self-documented event samples. Using HepML and MCDB together allows to make MC simulation chain fully automated and share well prepared MC event samples to the wide community of physicists. Web and Grid access to the data are both supported.

Monte Carlo Data Base

The LCG MCDB project has been created to facilitate communication between experts/authors of MC generators and users of the programs in the LHC collaborations. The current version of LCG MCDB [1, 2] provides flexible infrastructure to share event samples and keep the files in a reliable and convenient way. It has several interfaces, mainly Web-based, which help to carry out routine operations with the stored samples by users and authors of the samples.

The second motivation behind the project is to create a central database of MC events where stored event samples are publicly available for all groups to use and validate. Often, the same event samples are created by different experimental groups independently several times. If the samples are publicly available and equipped with corresponding and comprehensive documentation it can speed up cross checks of the samples themselves and applied physical models. In many cases it also prevents a possible waste of researcher time and computing resources.

LCG MCDB can particularly be useful in tasks where preparation of an event sample requires specific knowledge of the Monte-Carlo codes/techniques applied, significant computing power, and/or constant interaction authors of the events. For instance, this situation can arise if we use such MC programs as ALPGEN, CompHEP, GRACE, or MadGraph. Even MC generators of general purpose as PYTHIA or HERWIG sometimes require keeping of event files themselves. Examples of this sort happen in simulations of rare processes and/or with strong pre-selection cuts. In order to simplify tasks mentioned above, LCG MCDB development was initiated some time ago [3, 4, 5].

Usually it is not possible to strictly distinguish between data and meta-data, since the separation depends on situations where the data are exploited. In our concrete case we discriminate between events, as sets of particle 4-momenta (data), and information describing the events as the whole event sample (meta-data). Meta-data form the main contents of MCDB. In this sense, MCDB can hold a path to an event sample only and the sample itself can be located somewhere else. MCDB interfaces provide the means to manipulate with event meta-data. Knowledgebase is a special kind of database for knowledge management. It provides the means for the computerized collection, organization, and retrieval of knowledge. According to the definition one of the specific features of knowledgebase is that it keeps meta-data, i.e. information on data.

Comprehensive description of an event sample requires a lot of information, which should be entered to the database. However, in practice, in this specific application area a large part of the information is common for lots of samples. For example, author can re-use any pre-entered information from MCDB, or can create his/her own event description based on already entered information. The second idea behind the current design of MCDB is that MCDB is an area for interaction between two different communities, producers of events and consumers of the events. The latter users are end-users and the former users are authors. Any physicist who feels his/her sample is worthy to be kept in MCDB can make a request to open a new author account on the MCDB server. It means that MCDB does not assume to have a special team of event producers to prepare events according to end-user requests.

The standard information to describe event samples can be divided into several blocks. Each of the blocks corresponds to a definite set of parameters which are necessary to interpret a concrete event sample. The list below gives a short description of the main blocks:

- General information about a simulated event sample or a group of samples
  - Title of physical process
  - Physical Category (e.g. Higgs, Top physics or W + jets processes)
  - Abstract (short description)
  - List of authors
  - Name of an experiment and/or a group (for which the sample was prepared or intended)
  - Author comment on the sample (some additional unstructured information on the sample)
- Physical process
  - Initial state (names of beam particle, energy, etc.)
  - Final state (name of the final particles, etc.)
  - QCD scale(s)
  - Process PDF (parton distribution functions) applied
  - Information on separate subprocesses, if they are distinguished
- Event file
  - File name
  - The number of events
  - Cross section and cross section error(s)
  - Author comment
- Used MC generator
  - Name and version
  - Short description
  - Home page Web-address
- Theoretical model used to simulate the events
  - Name
  - Short description
  - A set of physical parameters and their values with the authors descriptions
- Applied cuts

There are several blocks in LCG MCDB, which are realized:

- Content Management System with a powerful and flexible Web interface for authors of event samples. It should have several types of templates to simplify the task of event sample description.
- A block of tree graph of physical categories with articles published by authors. This is the main part MCDB visible via Web browsers with no authentication in MCDB;
- A powerful search engine based on SQL/XML to search for contents of MCDB;
- A programming interface to CASTOR [10], which is used as a native storage of event samples;
- A block of direct uploading of files to MCDB from different locations (Web, AFS, CASTOR, EOS, Grid, Xrootd);
- Block of direct downloading of files from MCDB via Web, CASTOR, GRID, EOS, Xrootd (all by URI);
- A flexible and reliable authentication system based on CERN AFS/Kerberos logins or LCG GRID certificates;
- Backup system for all stored samples and corresponding SQL information;
- API to the LHC Collaboration software environments;
- The standard record of an event sample. The record should be encoded to a set of SQL tables;
- A unified and flexible format of event files based on the LHEF agreement and the HepML language. A programming package which supports the format.

HepML language to describe MC samples With the introduction of LHEF event files format [6], we have a common general format for representation of simulated events, which has become a generally accepted tool for the community of developers of MC generators. But LHEF has a limited power to accommodate the events with meta-data (detailed physical description of sample and generation parameters). Keeping the metadata within event samples are extremely useful for correct processing, verifying and re-usage of the event samples. Now the only project proposed unified meta-data description and storage, is HepML [7, 8].

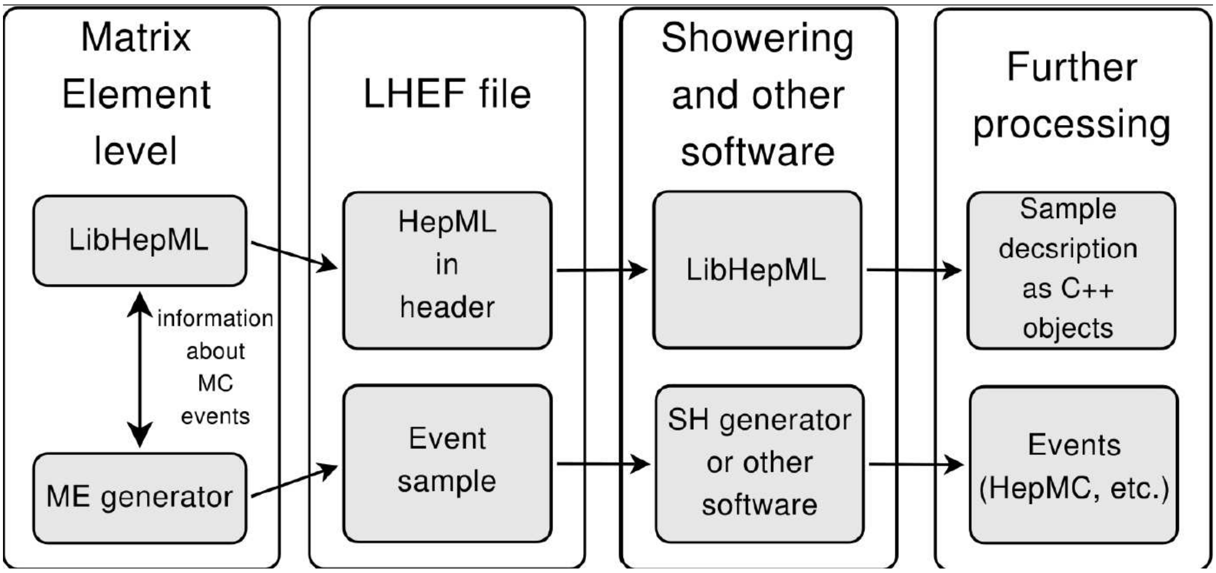

Figure 1: Place of HepML in Monte Carlo simulation chain

HepML project is an effort to state a unified extensible way of MC events description and provide program libraries to work with such meta-data. The main goal of the project is to store all possible information from MC generators in XML view, as well as to store generator input parameters and setup. In addition, HepML block is an allowed part of LHEF standard event file header, so keeping events metadata as a HepML document inside LHEF header is natural standard way. In the next sections we describe the LCG MCDB and HepML design and ideas in more detail, and briefly portray subsystems and modules of LCG MCDB.

At present, each MC generator supports its own output format of event files. Authors of Matrix Element tools (the term originates from [9]) provide interface programs to pass the events of a particular MC generator to the subsequent level of simulation (i.e. showering, hadronization, decays, simulation of detector response). The first step to standardize such interfaces has been described in the agreement Les Houches Accord Number One (LHA-I) [9], where a definite and strict structure of FORTRAN COMMON BLOCKS to transfer the necessary information from one code to another was fixed. The second step in this direction has been done in the agreement Les Houches Event File (LHEF) [6], where the information fixed in LHA-I is translated to the event file structure. All other information can be kept in a specific place inside the header of the event file. The standard does not apply any limitations on the extra information and the structure of the block. The next natural step is to provide a unified format to keep the necessary information within the LHEF structure. In this context the other information means the metadata described in the previous section and some other information specific to the sample (parameters of matching in different schemes, information on specific NLO approximations, jet parameters, etc.). Owing to the highly dissimilar nature of the information, the most appropriate technology for the unified representation could be XML-based format. In this case it can provide the possibility to describe the stored information in a very flexible and structured way.

The main idea behind the XML-based format is the flexibility to build and include a set of necessary parameters in an event file. For example, different MC generators may use the same tags for description of the physical parameters or they may need to keep specific information (through introduction of new dedicated tags). The new tags do not spoil the event file format and we do not need to re-write our routines which process these event files automatically. HepML is now being developed within the special LCG HepML project in collaboration with the CEDAR project [10]. More technical information is available on our wiki [11].

As the first part of HepML we have prepared several XML Schemas. The main goal of the Schemas is to provide a general and formal description of event data structures which are kept in XML files. Adapting the idea authors of MC codes can use powerful XML tools in developing of I/O routines. If the routines are consistent with the Schemas, event files generated by the routines can be read by other programs without changes in input routines of the programs.

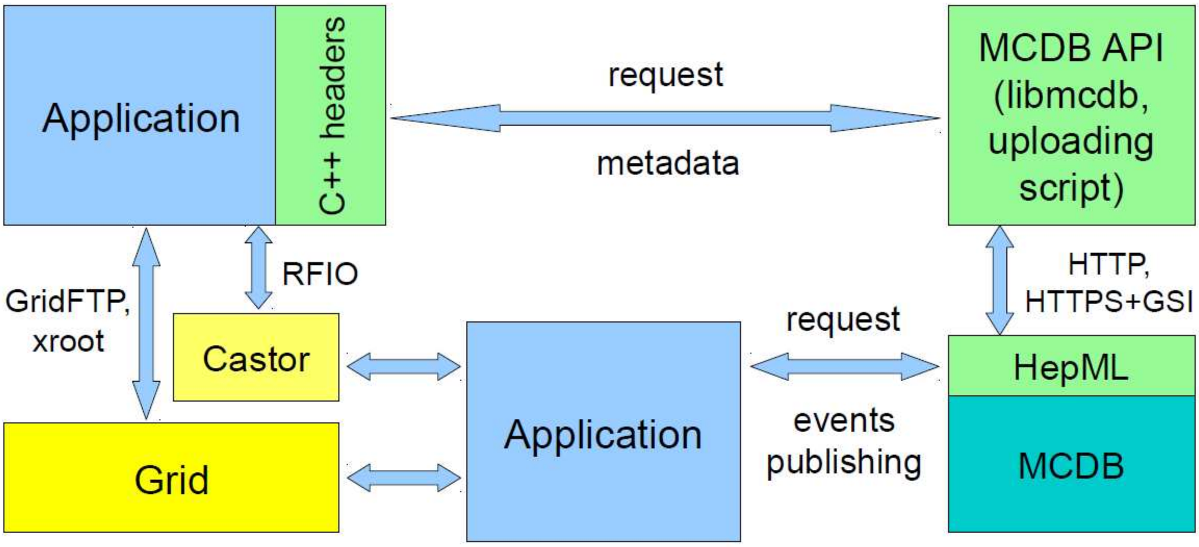

Figure 2: MCDB Application Program Interface

Also the Schemas can be used for validation of event files if the files are written according to HepML specifications. Now we have three main Schemas. The first XML Schema lha1.xsd corresponds to the whole set of parameters composing the LHA-I agreement. The other two Schemas, sample-description.xsd and mcdb-article.xsd, describe parameters, which are necessary to generate an XML data for an event sample and to form an LCG MCDB article for the sample. The CEDAR team develops other XML Schemas for other tasks arising in the problem of automatization of data processing in HEP. Now all the Schemas are unified in one general formal XML Schema hepml.xsd, which includes all the other Schemas as sub-Schemas. This solution leaves freedom to develop Schemas and software in independent groups, but to use Schemas of both groups in one software project. All the developed Schemas are available at [12].

Internal adaption of LHEF and HepML formats into the most popular MC generator projects would result in a significant improvement of the MC event sample documentation and book-keeping. Such adaption of the unified standards provides the possibility to develop new standard interfaces and utilities. The LCG MCDB project already implements a part of HepML specifications in MCDB API. Example of usage of HepML in simulation chain is shown on Fig. 1.

Using MCDB and HepML together in simulation chain

Apart from the MCDB server, LCG MCDB team provides application programming interfaces (APIs) specific to the simulation environments of the LHC Collaborations. The main idea of these subsystems is to develop a set of routines for the collaboration software which would give a direct access to the LCG MCDB files during the MC production on computer farms.

The most simple way to access event samples is to use direct WEB, CASTOR or GRID path to the event samples. This way does not require any special software developments on the side of collaboration software. This way, however, does not provide any possibility for automatic access to event sample description. This is the reason we developed a more complicated interface which could be used for automatic processing of event samples and the corresponding documentation. According to our idea, MCDB team provides API based on XML representation of event sample metadata. The current version of the API is a C++ library, which can be added to collaboration software. The XML output from LCG MCDB is based on the HepML specifications (for more details, see previous section).

The current library contains C++ classes and provides routines to fill the class objects with information from a MCDB article, including paths to event files (CASTOR, GRID, HTTP, Xrootd) attached to the article. Such an interface has already been implemented in the CMS collaboration software environment. Fig. 2 reports a general scheme of an interaction between LCG MCDB to external user software via the API.

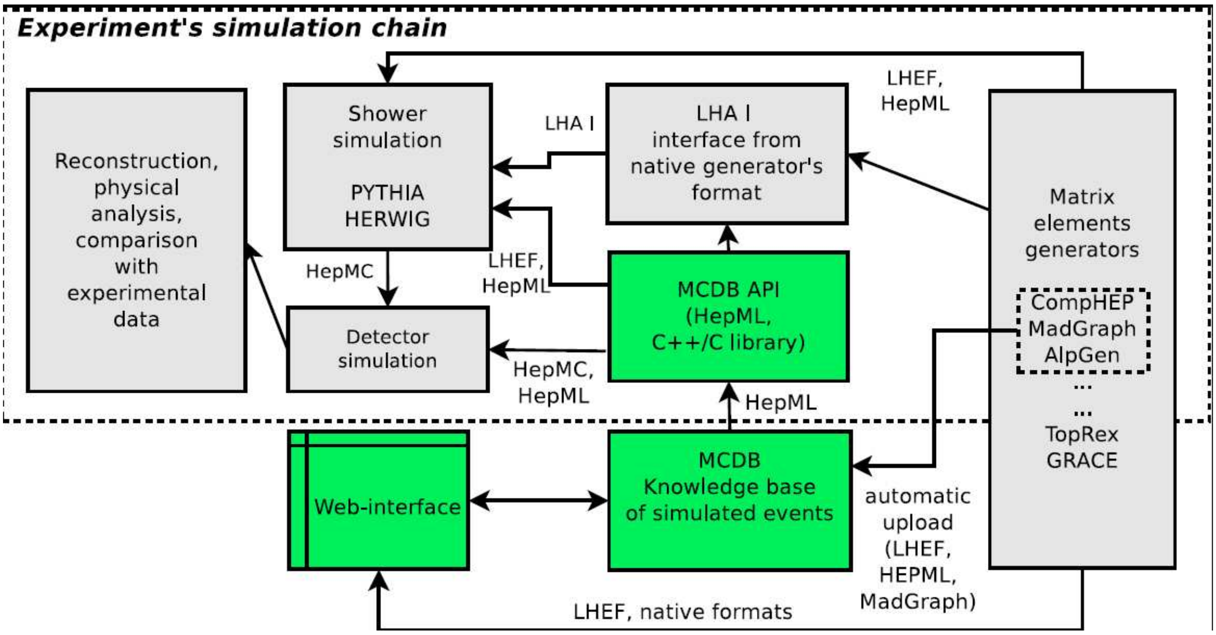
Experiment's simulation chain
Reconstruction, physical analysis, comparison with experimental data
Shower simulation
PYTHIA
HERWIG
HepMC
Detector simulation
LHA I
LHEF, HepML
HepMC, HepML
LHA I interface from native generator's format
MCDB API (HepML, C++/C library)
HepML
LHEF, HepML
Matrix elements generators
CompHEP
MadGraph
AlpGen
...
...
TopRex
GRACE
Web-interface
MCDB Knowledge base of simulated events
automatic upload (LHEF, HEPML, MadGraph)
LHEF, native formats

Figure 3: MCDB and HepML in CMS simulation chain

Other significant feature that was accomplished are extensions of API specific to the automatic uploading of HepML information and event samples to MCDB. Now it is possible to upload LHHEF file with HepML part in header (or just having a separate HepML description file) to the MCDB. Along with HepML, headers coming from MadGraph generator is also supported in MCDB's automatic uploading.

Here we describe a method of making a uniform Monte-Carlo simulation chain based on MCDB and HepML. Fig. 3 is used as a visual aid for this paragraph.

Events are generated with a Matrix Element MC generator, stored in MCDB, processed with a Shower and Decay MC generator, and then passed to the Detector simulation software. Handling all the data in automatic way allows a user to avoid most of human-related errors and can save researchers time and keep simulated event files in order.

Firstly, data from a Matrix Element generator are kept in the standard LHE format. Detailed samples description in HepML form is included to the LHEF header (libraries to write the HepML code are provided). After this step MC generator provides self documented event sample with the full description of simulation inside it. At the moment, CompHEP generator can provides extended information in the form of HepML code inside the standard LHEF header [6].

The next step is to store the sample in a public place. LCG MCDB allows automatic upload and documentation (for several types) of such event files.

Then, the events can be taken via a Grid interface or directly from CASTOR at CERN or through the web interface. Both URLs to the samples and its detailed description are provided by LCG MCDB in the HepML form using a unique description number (called an article number) as a reference.

After getting the files, they can be transferred to the next generation level, Showering and Hadronization, along with the full meta-data set. Then, stored, for example, in the HepMC format, the data can be processed by Detector Simulation software. By now, this conception is accomplished in CMSSW within the CMS experiment software environment. Our HepML libraries were adopted to be partially included to CMSSW. This libraries are responsible for parsing HepML response from LCG MCDB. MCDB was used for official CMS production in 2011.

Results and conclusion

MCDB is a special knowledgebase designed to keep event samples for the LHC experimental and phenomenological community. Now, a new version of the software has been finished and the server is ready for use by the community. Some new important features are implemented in the software. The features simplify and improve the process of documentation of event samples.

At present, LCG MCDB is a stable software package and ready to use for the LHC community. A dedicated web server is deployed at CERN. There is an application programming interface for program access to MCDB content using HepML. An XML

Schema and a program tool to handle (create, validate and parse) HepML documents are publicly available.

In addition to the server, MCDB team has prepared an API for the LHC Collaboration software environments. Implementation of the API to the software environments could give a possibility to use MCDB as a native storage in large-scale productions in collaborations. In 2011, MCDB was the official storage for Monte Carlo samples in CMS MCD production.

Acknowledgements

The work of S. Belov was partially supported by JINR grant No. 13-603-01.

References

[1] S. Belov et al., *LCG MCDB - a Knowledgebase of Monte Carlo Simulated Events*, Computer Physics Communications, Volume 178, Issue 3, 1 February 2008, p. 222 [hep-ph/0703287]

[2] MCDB project: http://mcdb.cern.ch

[3] M. Dobbs et al., *QCD/SM Working Group: Summary Report, Les Houches*, Physics at Tev Colliders 2003 [arXiv:hep-ph/0403100]

[4] P. Bartalini, L. Dudko, A. Kryukov, I.V. Selyuzhenkov, A. Sherstnev and A. Vologdin, *LCG Monte-Carlo data base: LCG generator services subproject* [hep-ph/0404241]

[5] L. Dudko, A. Sherstnev, *CMS MCDB*: http://cmsdoc.cern.ch/cms/generators/mcdb

[6] J. Alwall et al., *A standard format for Les Houches event files*, Comput. Phys. Commun. 176, 300 (2007) [arXiv:hep-ph/0609017]

[7] S. Belov et al., *HepML, an XML-based format for describing simulated data in high energy physics*, Comput.Phys.Commun. 181 (2010) pp. 1758-1768 [arXiv:1001.2576]

[8] A. Buckley et al., *CEDAR HepML Web Page*: http://projects.hepforge.org/hepml

[9] E. Boos, et al., *Generic user process interface for event generators*, [arXiv:hep-ph/0109068]

[10] J.M. Butterworth, S. Butterworth, B.M.Waugh,W.J. Stirling, M.R. Whalley, *The CEDAR Project*, [arXiv:hep-ph/0412139]

[11] CEDAR HepML Wiki: http://projects.hepforge.org/hepml/trac/wiki

LCG HepML Wiki, http://twiki.cern.ch/twiki/bin/view/Main/HepML

[12] S. Belov, L. Dudko, A. Sherstnev, HepML XML schema, http://mcdb.cern.ch/hepml/schemas/