RunMC: an object-oriented analysis framework to generate Monte Carlo events for current and future HEP experiments

S. Chekanov

HEP division, Argonne National Laboratory, 9700 S.Cass Avenue, Argonne, IL 60439, USA
E-mail: `chekanov@mail.desy.de`

Abstract

RunMC is a C++ object-oriented framework aimed to generate and to analyze high-energy collisions using Monte Carlo simulations. The package provides a common interface to different Monte Carlo models using modern physics libraries developed for the LHC and NLC experiments. Physics calculations can easily be loaded and saved as external project modules. This simplifies the development of complicated physics calculations for high-energy physics in large collaborations.

1 Introduction

Monte Carlo models (MC) written in FORTRAN 77 are widely used in many high-energy physics laboratories worldwide. These models are known to be fast, robust and well tested. However, the main choice for future high-energy experiments is an object-oriented programming language, either C++ (the LHC experiments at CERN) or Java (the NLC project). Some steps towards converting the FORTRAN MC models to the C++ programming language have already been undertaken [1]. However, such models written in C++ will require a thoughtful verification to insure that their predictions are consistent with the original FORTRAN-based MC programs, as well as with the physics results obtained in the past. Such verifications will go over certain time, and a tool which allows to perform such comparisons is urgently needed.

A program which allows running of both FORTRAN-coded and C++ MC models using a common C++ programming environment should be valuable. This is important not only for comparisons and verifications of these MC models. Such a C++ framework can also extend the lifetime of FORTRAN-based models especially for the LHC, NLC and TEVATRON communities, and can provide compatibility of most popular MC models with the new software to be used for current and future HEP experiments.

The RunMC package [2] is a common C++ frond-end of Monte Carlo models which provides a unified approach to generate and analyze very different MC models independent of their native codes. In this approach, the MC output (typically the HEPEVT record) is converted to C++ classes for further analysis or graphical representation (histograms). The graphical user interface (GUI) of this program helps to initialize MC models and histograms, as well as to monitor the event generation.

The RunMC program fully complies with the change in the programming paradigm of data analysis, and meets the requirements of future high-energy experiments. Instead of FORTRAN-based analysis tools, such as PAW [3] and HBOOK [4], it uses the modern CERN C++ analysis packages, CLHEP [5] and ROOT [6].

In this respect, the RunMC program is similar to the JetWeb server [7], which also provides the ability to compare the existing MC models and to confirm the physics assumptions they contain. However, in contrast to JetWeb, the RunMC program was designed as a stand-alone desktop application. Therefore, the user has full access to his calculations and to the program itself.
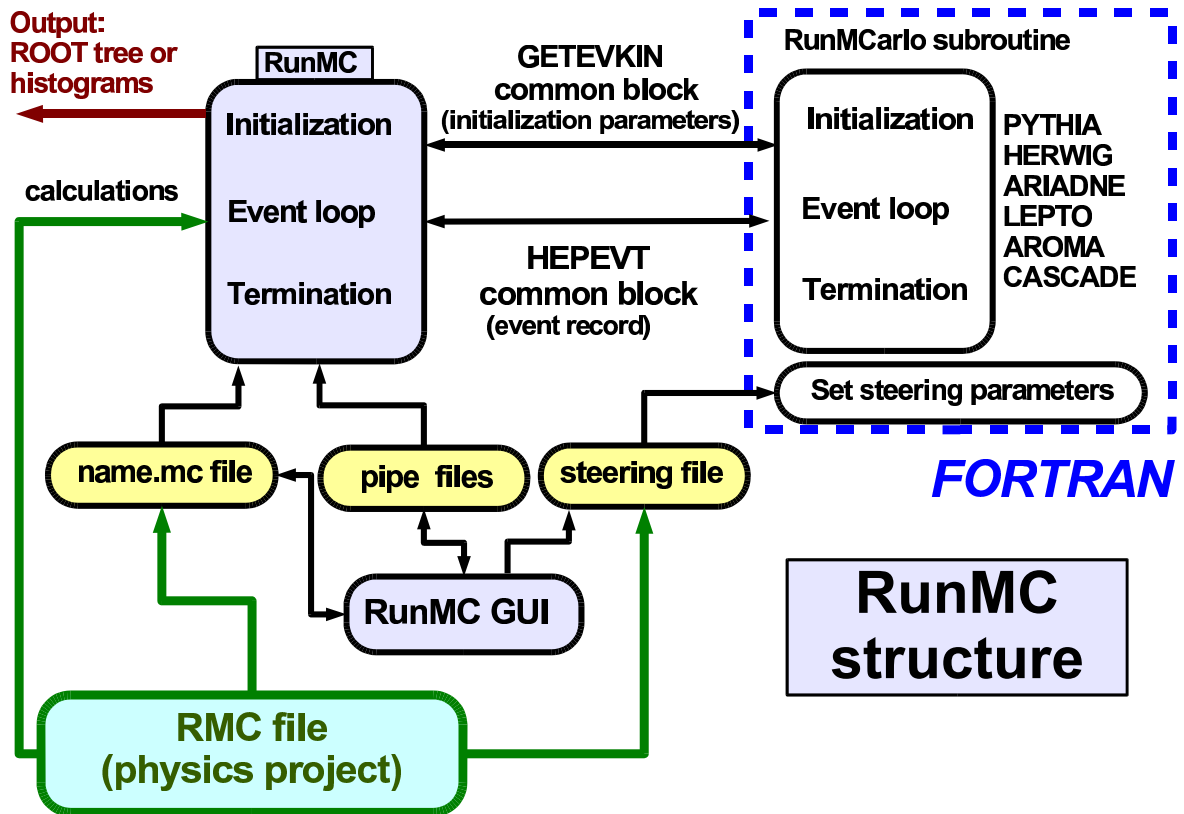
Fig. 1: The diagram shows the structure of the RunMC program.

The RunMC program also provides an interface to the popular HZTOOL package [8], thus many physics calculations from HERA, LEP and TEVATRAN can easily be accessed. In addition, within the RunMC approach, the concept of project modules was introduced (in fact, the HZTOOL package is one of such modules). A project file, which can contain external calculations, MC tunings, histogram definitions, etc., can be loaded to RunMC with the same ease as a document can be opened in the Microsoft Word program. The project files are small and platform independent, therefore, it is fairly simple to share complicated physics calculations between scientists in large collaborations.

2 Program structure

RunMC consists of a GUI and several RunMC MC programs. There are two implementations of the RunMC GUI: one is written using the Wide Studio C++ classes [9] and an alternative GUI is based on Java. Due to complete independence of RunMC GUI from RunMC MC programs, one can run jobs in the background without any GUI or pop-up window.

The RunMC programs integrate the C++ ROOT and CLHEP packages with native implementations of MC models. A schematic structure of RunMC is illustrated in Fig. 1. The following MC models are included to RunMC (version 3.3): PYTHIA 6.2 [10], HERWIG 6.5 [11], ARIADNE 4.12 [12], LEPTO 6.5 [13], AROMA 2.2 [14], CASCADE 1.2 [15], PHOJET 1.05 [16], RAPGAP 3.1 [17]. There are several executable RunMC MC programs corresponding to each MC model.

RunMC GUI communicates with the RunMC MC programs using pipe files located in the directory "$RUNMC/pipes". Here, "$RUNMC" denotes the installation directory which has to be defined by the user. All the directories to be discussed below are assumed to be located in this directory.

2.1 RunMC GUI

RunMC GUI allows an interaction between the user and RunMC MC programs. At present, two types of RunMC GUI are available: a user interface based on C++ (can be executed with the command "runmc") and that based on Java (the command "jrunmc"). Below we will describe only the C++ RunMC GUI.

The task of RunMC GUI is to generate the output file "project.mc", where "project" is a user-defined name of the current calculation. This file contains the most important information for MC running, such as the type of MC model, the number of output events, the type of colliding particles, their energies, RunMC output (histograms or ntuples) etc.

RunMC GUI adopts the following strategy to define the histograms: The window "Variables" contains the names of the variables (with some additional comments) defined for a certain physics project. The user should select the appropriate variable and copy it to the window "Histograms" by clicking on the corresponding variable name. If two one-dimensional histograms are defined, a two-dimensional histogram can be build from these two histograms using RunMC GUI.

The variable names are divided into the three categories: event-based variables (characterizing the event as whole), single-particle densities (filled for each particle/jet; the variable name starts with "@") and two-particle densities (filled for each particle/jet pair; the name starts with "@@"). The histograms can also be filled in the user-defined subroutine "user-run.cpp"; in this case the naming convention for the variables discussed above is unnecessary.

During the event generation, the ROOT canvas can display the output histograms (up to eight in total) with filled events. The output log from RunMC MC is written to ".analmc.log" (a symbolic link to the "project.log" file). Possible errors are redirected to the file "project.err", which is constantly monitored by RunMC GUI.

The ROOT histograms are automatically modified at the end of the fill if they are required to be normalized to the total number of events or converted to differential cross sections. Note that there is no need to wait until the end of the current run: once the histogram statistics is sufficient, one can terminate the run by clicking "Stop" on the GUI window. Histograms should be saved in the ROOT file "project.root" for further studies. The style of the histograms can further be modified using the ROOT canvas editor.

2.2 The RunMC MC programs

The RunMC MC programs integrating Monte Carlo models with ROOT C++ classes have the generic names "analmc.MCname", where "MCname" denotes the MC name. The main C++ function of RunMC MC is located in the file "analmc.cpp" (in the directory "main/src"). The C++ code accesses the HEPEVT common block of a given MC program via a C-like structure. The RunMC MC program receives the initial parameters via the symbolic link ".analmc.ln" pointing to the file "project.mc".

Each MC model has its own FORTRAN subroutine "runmcarlo" which provides an interface to the native MC code. This interface program (in the file "RUNMC-MCname.f") is located in the directory "main/mcarlo/MCname". The task of the subroutine "runmcarlo" is to fill the HEPEVT common block. In addition, some initial settings are done by accessing a C/C++ structure with the initial parameters defined in the "project.mc" file. The main function in "analmc.cpp" calls this interface subroutine and fills the C/C++ structure which represents the complete HEPEVT event record. The output is copied to the class "HEPLIST" which can be accessed by external calculations. The HEPLIST class consists of several vectors based on the LorentzVector vector class (from the CLHEP library) which represents four-momentum of a particle (or a jet). The definition of the HEPLIST class, as well as other include files, can be found in the "main/inc" directory. Note that the user still can access more elementary event records (such as FORTRAN HEPEVT common block) which can be used to transform them to other event classes and physics calculations.

There are several physics packages available inside RunMC MC to transform the original four-momentum vector of particles/jets to the required observable:

– the transformations provided by the physics vector class "LorentzVector" from CLHEP can be used, since a particle or a jet is represented as a general four-vector based on this class;

– the event-shape calculations are available using the package developed by M. Iwasaki [18];

– the longitudinally-invariant k_T algorithm as implemented in C++ [19] can be used for the jet reconstruction. In addition to this package, the JADE and Durham jet algorithms are implemented according to M. Iwasaki [18];

– the Breit frame calculations were implemented for ep deep inelastic scattering.

The physics packages and their documentation are located in the directory "main/physics".

3 User calculations

For a new physics calculation, the directory "proj" should be modified. This user directory can contain external calculations, steering cards for MC initializations, as well as the standard RunMC functions which are necessary to initialize and fill the histograms.

The user directory should always contain the file "project.mc" created by RunMC GUI. This file can be edited manually without the RunMC GUI program using any text editor. On Linux/Unix, one can load this file to RunMC GUI by executing the command "runmc project.mc" from the shell prompt (or using the option "Projects→read MC" of RunMC GUI).

The directory "proj" can contain steering files "MCname.cards" to redefine initial MC parameters. Such files can be created via RunMC GUI ("MC settings" option). For more flexibility, the MC initialization parameters can also be overwritten by FORTRAN-coded subroutines located in the directory "proj/ini". If this is not done, the default MC parameters will be selected according to the RunMC option.

To define histograms, user-defined variables should be calculated in the file "proj/user_afill.cpp". The output of this function is a pointer. The output variable name should always be associated with this pointer. The variable names should be specified in the file "user-name.txt". It includes the variable names to the list "Variables" accessed by RunMC GUI. Finally, to compile the source codes and to rebuild all RunMC MC programs to take into account changes made in the project source files, one should type "make" in the "proj" directory. All MC programs will be recompiled and RunMC GUI will be updated with new histograms. Then, the command "runmc" (or "jrunmc") should be executed from the directory "proj" to start RunMC GUI. The main advantage of this approach is that once a necessary variable is defined, new histogram definitions do not require the MC recompilation.

RunMC histograms can also be filled using the conventional method, i.e. in the function located in "user-run.cpp". In this case, the initialization of histograms is not required, as long as the file "project.mc" defines which histograms should be filled and what presentation style should be used to fill the histograms. The histograms can be initialized in the file "user-init.cpp" using the standard ROOT procedure.

4 Physics calculations as external RMC projects

In order to share complicated analysis calculations or to store them for future use, the directory "proj" can be packed into an external RMC file with the extention "rmc". For example, "project.rmc" is the RMC file which has the user-defined name "project". The "proj" directory inside of it has a file "project.mc" with RunMC GUI settings, user-defined external functions, libraries, make files MC steering files, etc.

RunMC GUI can automatically load and recompile such project RMC files (see details in [2]). The user can also save his/her calculations into a RMC file for future analysis. As it was mentioned, the project files are compact and platform independent, therefore, it is fairly simple to share physics calculations between the users, as long as the RunMC package is installed.

At present, several RunMC project files are available on the Web [2] (they are also included in the directory "archive" of RunMC):

- the default project. Only pre-installed variables can be included in the calculations.
- HERA kinematic variables (Q^2, x, etc.);
- jets at HERA and LHC using the longitudinally-invariant k_T algorithm in the Breit frame. In addition, the ratio of jet cross sections at the parton and hadron levels are calculated (the so-called hadronisation corrections);
- D^* cross sections in ep collisions at HERA;
- cross sections for strange-particle production in ep collisions at HERA;
- the HZTOOL package [8];
- the event-shape variables in e^+e^- at NLC energies;
- several examples of how to visualize tracks and k_T jets in 3D for a single MC event (e^+e^-, ep, pp collisions).

The RMC project files discussed above only illustrate how to set up and to develop new physics calculations in the RunMC framework. For practical applications, these examples should be modified.

5 RunMC ROOT tree analyzer

In addition to the standard functionality of the MC event simulation, RunMC GUI can also use ROOT trees as the input for physics calculations.

The ROOT tree can be generated by selecting the option "HEPEVT" or "RUNMC", in addition or instead of the ROOT histogram option. Then, the MC events should be generated as usual, but this time a ROOT tree with the extension ".rtup" or ".htup" will be created. Then, RunMC can run over this ROOT tree if, instead of the MC model, the option "RUNMC" or "HEPEVT" is selected. Several ROOT trees can automatically be included in the analysis, as long as they are in the same directory. The analysis of the ROOT trees is very similar to the standard run over MC events. External RMC files can be used to include new calculations, variables and histograms.

The main advantage of the RunMC ROOT tree analyzer is that physics calculations can be validated significantly faster than when RunMC is used to generate events and to fill histograms at the same time. In case of the ROOT trees, RunMC can fill histograms by a factor of $\sim$10–15 faster, thus the RMC project files can be validated and analyzed more efficiently.

With this additional functionality, the RunMC program can also be used to analyze experimental data if the event record is converted to the appropriate ROOT tree. The data analysis can be performed using exactly the same RMC project files as for the usual MC simulation runs.

References

[1] M. Bertini, L. Lönnblad and T. Sjöstrand, Comput. Phys. Commun. **134**, 365 (2001);
L. Lönnblad, THEPEG: *Toolkit for High Energy Physics Event Generation* (unpublished),
available on `http://www.thep.lu.se/ThePEG/`.

[2] S. Chekanov, RUNMC - C++ *object-oriented framework for monte carlo models*. Preprint
hep-ph/0411080, Comm. Phys. Comm. (in press), available on
`http://hepforge.cedar.ac.uk/runmc/`
`http://www.hep.anl.gov/chakanau/runmc`.

[3] Application-Software-Group, PAW: *Physics Analysis Workstation*, available on
`http://wwwasd.web.cern.ch/wwwasd/paw/`.

[4] R. Brun and D. Lienart, CERN-Y250 (1987).

[5] L. Lönnblad, Comput. Phys. Commun. **84**, 307 (1994);
M. Fischler and A. Pfeiffer, CLHEP - *new developments and directions*, in *The proceedings of
International Conference on Computing in High Energy Physics and Nuclear Physics (CHEP
2000), Padova, Italy, 7-11 Feb 2000*. Available on
`http://wwwasd.web.cern.ch/wwwasd/lhc++/clhep/`.

[6] R. Brun and F. Rademakers, Nucl. Instrum. Meth. **A389**, 81 (1997);
R. Brun, F. Rademakers, P. Canal and M. Goto, ECONF **C0303241**, MOJT001 (2003).

[7] J.M. Butterworth and S. Butterworth, Comput. Phys. Commun. **153**, 164 (2003).

[8] J. Bromley et al., HZTOOL: *a package for* MONTE CARLO - *data comparisons at hera*, in *Future
Physics and HERA*, eds. Buchmuller, W. and Ingelman, G., Vol. 2, p. 611. Hamburg, Germany,
1996-1997, available on `http://hztool.hep.ucl.ac.uk/`.

[9] T. Hirabayashi, *The Wide Studio project* (unpublished), available on
`http://www.widestudio.org`.

[10] T. Sjöstrand, L. Lönnblad and S. Mrenna, *Pythia 6.2: Physics and manual*. Preprint
hep-ph/0108264, 2001.

[11] G. Corcella et al., JHEP **0101**, 10 (2001).

[12] L. Lönnblad, Comput. Phys. Commun. **71**, 15 (1992).

[13] G. Ingelman, A. Edin and J. Rathsman, Comput. Phys. Commun. **101**, 108 (1997).

[14] G. Ingelman, J. Rathsman and G. Schuler, Comput. Phys. Commun. **101**, 135 (1997).

[15] H. Jung, Comput. Phys. Commun. **143**, 100 (2002).

[16] R. Engel, PHOJET (unpublished), 1997, available on
`http://www-ik.fzk.de/~engel/phojet.html`.

[17] H. Jung, Comm. Phys. Commun. **86**, 147 (1995).

[18] M. Iwasaki, *Jet-EventShape-Finders package* (unpublished).

[19] J.M. Butterworth, J.P. Couchman, B.E. Cox and B.M. Waugh, Comput. Phys. Commun.
153, 85 (2003).