LUND
UNIVERSITY

ThePEG, Pythia7, Herwig++ and Ariadne

- Introduction
- Overview
- Status
- Current Work
- Future Plans

Zeuthen
2005.05.24
Leif Lönnblad

LUND
UNIVERSITY

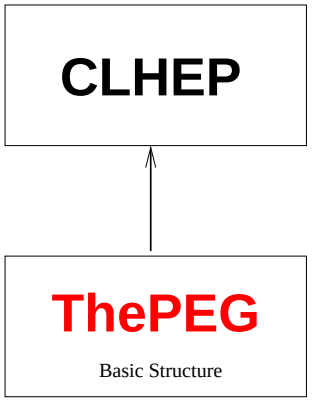
ThePEG, Pythia7, Herwig++ and Ariadne and Pythia8

- Introduction
- Overview
- Status
- Current Work
- Future Plans

Zeuthen
2005.05.24
Leif Lönnblad

What is ThePEG

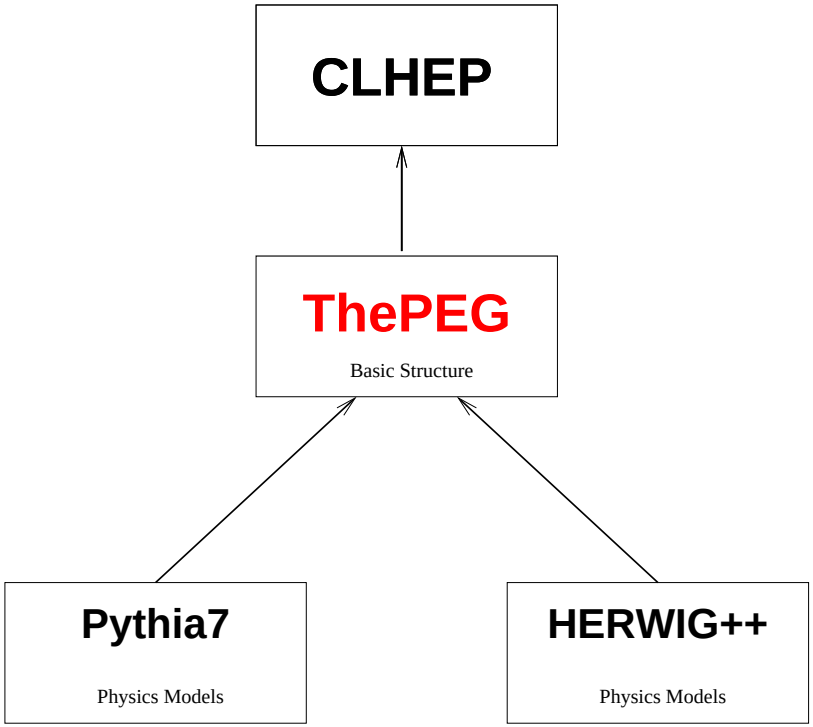
ThePEG consists of the parts of Pythia7 which were not specific to the Pythia physics models. It provides a general structure for implementing models for event generation.



What is ThePEG

ThePEG consists of the parts of Pythia7 which were not specific to the Pythia physics models. It provides a general structure for implementing models for event generation.

Both Pythia7 and Herwig++ are built on ThePEG.

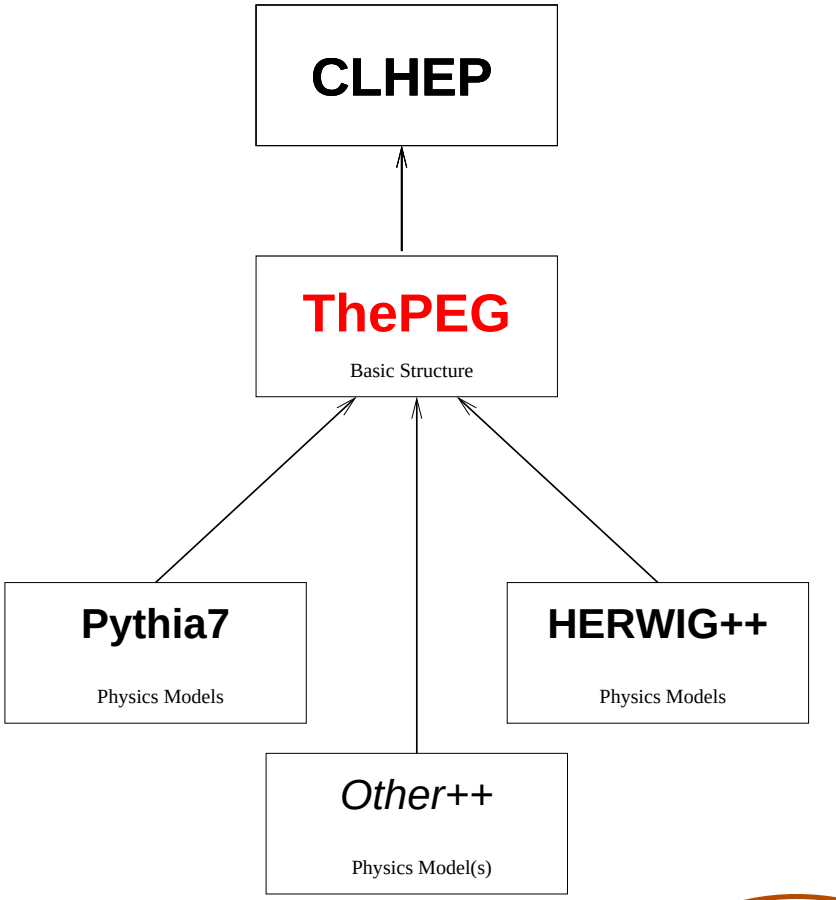


What is ThePEG

ThePEG consists of the parts of Pythia7 which were not specific to the Pythia physics models. It provides a general structure for implementing models for event generation.

Both Pythia7 and Herwig++ are built on ThePEG.

But it is open for anyone...



The components of ThePEG

- Basic infrastructure: Smart pointers, extended type information, object persistency, Exceptions, Dynamic loading, ...
- Kinematics: Extra utilities on top of CLHEP vectors, 5-vectors, flat n-body decay, ...
- Repository: Manipulation of interfaced objects. Setting of parameters and switches and connecting objects together.
- Handler classes: to inherit from to implement a specific physics model.
- Event record: Used to communicate between handler classes.
- Particle data: particle properties, decay tables, decayers etc...

ThePEG defines a set of abstract Handler classes for hard partonic sub-processes, parton densities, QCD cascades, hadronization, etc. . .

These handler classes interacts with the underlying structure using a special Event Record and a pre-defined set of virtual function definitions.

The procedure to implement e.g. a new hadronization model, is to write a new (C++) class inheriting from the abstract `HadronizationHandler` base class, implementing the relevant virtual functions.

When implementing models for event generation there is typically a number of parameters and options available (in addition to the parameters of the Standard Model).

ThePEG defines a uniform way of interacting with the handler classes. The sub-classes may define a set of `InterfaceBase` objects corresponding to parameters, switches or references to objects of other `Interfaced` classes.

These are then used by the `Repository` to manipulate the corresponding member variables in the handler classes.

How to use ThePEG

Running ThePEG is separated into two phases.

- Setup:

  A setup program is provided to combine different objects implementing physics models together to build up an `EventGenerator` object. Here the user can also change parameters and switches etc.

  No C++ knowledge is needed for this. In the future we would like a nice GUI so that the user can just click-and-drag.

  The Repository already contains a number of ready-built `EventGenerators`. It is also possible to specify `AnalysisHandler` object for an `EventGenerator`.

  In the end the built `EventGenerator` is saved to a file.

```
cd /Defaults/EEGenerators
set SimpleLEPHandler:HadronizationHandler stdLundFragHandler
set stdLundFragHandler:FlavourGenerator:BaryonMode 1
set stdLundFragHandler:ZGenerator:a 0.24
set SimpleLEPGenerator:EventHandler SimpleLEPHandler
set SimpleLEPGenerator:NumberOfEvents 10000
saverun SimpleLEP SimpleLEPGenerator
```

- Running:

  The saved `EventGenerator` can be simply read in and run using a special slave program. If `AnalysisHandlers` have been specified, this is all you have to do.

  Alternatively the the file with the `EventGenerator` can be read into any program which can then use it to generate events which can be sent to analysis or to detector simulation.

  The `ThePEG::Events` can, of course, be translated into `HepMC::GenEvents` or whatever.

The `EventGenerator` class is the main class administrating an event generation run.

It maintains global information needed by the different models: The `ParticleData` objects to be used, a `StandardModel` object with couplings etc, a `RandomGenerator`, a list of `AnalysisHandlers` etc.

It also has an `EventHandler` object to administer the actual generation.

Status

ThePEG version 1.0α exists and is working. Snapshots of the current development code is available from http://www.thep.lu.se/ThePEG.

Pythia7 is now based on ThePEG. Version 1.0α exists and is working. Snapshots of the current development code is available from http://www.thep.lu.se/Pythia7.

Herwig++ is also based on ThePEG. Version 1.0 exists and is working. Can be obtained from http://www.hep.phy.cam.ac.uk/theory/Herwig++/.

PYTHIA7/THEPEG includes some basic $2 \to 2$ matrix elements, a couple of PDF parameterizations, remnant handling, initial- and final-state parton showers, Lund string fragmentation and particle decays.

HERWIG++ includes a new parton shower algorithm, improved cluster fragmentation. Mainly $\mathrm{e}^+\mathrm{e}^-$.

Current Work

The code documentation has been converted into Doxygen format. Soon to start with reference and user manual also using Doxygen.

The plan is to have many *Howto* examples to which the user community is welcome to contribute.

Previously the `EventHandler` structure was a bit too rigid. Although it allowed for alternatives through inheritance, it was not really intended or optimized for that.

Now the `EventHandler` class has been remodeled into an abstract base class allowing for different strategies of handling initial sub-processes - the handling of cascades, hadronization is still the same.

Currently there is two concrete classes, one for the standard THEPEG sub-process handling (`StandardEventHandler`) and one for general external Matrix Element generators via the Les Houches accord (`LesHouchesEventHandler`).

Portability is a bit tricky since ThePEG relies heavily on the ability to dynamically load libraries.

Currently ThePEG runs on any platform,

Portability is a bit tricky since ThePEG relies heavily on the ability to dynamically load libraries.

Currently ThePEG runs on any platform, as long as it is Linux with gcc version 3 or later.

The build process is being remodeled using `libtool` to facilitate portability.

PYTHIA7 ⇒ PYTHIA8

- Development of PYTHIA7 has stopped
- Instead Torbjörn Sjöstrand has gone off by himself to build PYTHIA8 which will NOT be based on THEPEG.
- Hopefully it will still be possible to call PYTHIA8 modules (?) from within the THEPEG framework.
- PYTHIA8 will be a rewrite of Fortran PYTHIA in C++.

Future Plans

- ThePEG: Documentation
- ThePEG: Java GUI
- ThePEG: Spin and Helicity stuff ready (Richardson), but could be expanded to HELAS-like ME generation
- ThePEG: CKKW ME/PS matching

Future Plans

- ThePEG: Documentation
- ThePEG: Java GUI
- ThePEG: Spin and Helicity stuff ready (Richardson), but could be expanded to HELAS-like ME generation
- ThePEG: CKKW ME/PS matching
- Herwig++: Initial state PS (with CKKW)
- Herwig++: SUSY/BSM stuff
- Herwig++: All the rest...

Future Plans

- ThePEG: Documentation
- ThePEG: Java GUI
- ThePEG: Spin and Helicity stuff ready (Richardson), but could be expanded to HELAS-like ME generation
- ThePEG: CKKW ME/PS matching
- Herwig++: Initial state PS (with CKKW)
- Herwig++: SUSY/BSM stuff
- Herwig++: All the rest...
- Ariadne: Dipole shower with CKKW.
- Ariadne: LDC model with multiple interactions.

Manpower

- ThePEG: L.L., Stefan Gieseke, Alberto Ribon, Peter Richardson.
- Pythia8 interface: (was Pythia7) L.L.
- Herwig++: Stefan Gieseke, David Grellscheid, Alberto Ribon, Peter Richardson, Mike Seymour, Phil Stephens, Bryan Webber.
- Ariadne: L.L. Nils Lavesson

Conclusions

ThePEG was intended to be the toolkit/platform for event generators for the LHC era.

Conclusions

ThePEG was intended to be the toolkit/platform for event generators for the LHC era.

Now, it will be one among a handful of platforms (together with Sherpa, Pythia8, ...).