Towards a Les Houches Events file format?

Torbjörn Sjöstrand, CERN/Lund, `torbjorn@thep.lu.se`

Clearly strong opinions — not possible to make everybody happy!

- minimal `xml` compliance?
- full-fledged `xml`-based parsing?
- flexible information content: omit some fields?
- new compulsory information?

Probably only one working solution:
1) Minimal version 1 that is strictly LHA, ready “soon”
2) Advanced version 2 with full `xml` support, ready “later”
Acceptance of version 2 “market-driven”! (What is in it for me?)

Discussion list: simple cc, dedicated, HepML?

Proposed minimal information content

Omitting blank lines, and lines beginning with an < or `#`, the file should contain *exactly* the LHA information, split across lines as follows:

1) Initialization information, given once

a) one line with process-number-independent information:

`IDBMUP(1) IDBMUP(2) EBMUP(1) EBMUP(2) PDFGUP(1) PDFGUP(2)` ⤸
⤹ `PDFSUP(1) PDFSUP(2) IDWTUP NPRUP`

b) `NPRUP` lines, one for each process `IPR` in the range `1 - NPRUP`:

`XSECUP(IPR) XERRUP(IPR) XMAXUP(IPR) LPRUP(IPR)`

2) Event information, repeated as many times as there are events

a) one line with common event information:

`NUP IDPRUP XWGTUP SCALUP AQEDUP AQCDUP`

b) `NUP` lines, one for each particle `I` in the range `1 - NUP`

`IDUP(I) ISTUP(I) MOTHUP(1,I) MOTHUP(2,I) ICOLUP(1,I) ICOLUP(2,I)` ⤸
⤹ `PUP(1,I) PUP(2,I) PUP(3,I) PUP(4,I) PUP(5,I) VTIMUP(I) SPINUP(I)`

Always blank-separated fields, with no variables omitted (but can be brief)

Proposed document structure: last week

```
# comment lines can go here
<header>
<!-- only xml tags in this section, including for comments -->
</header>
# further comments allowed between blocks
<init>
initialization information, as described previously
# further comments on initialization
</init>
# further comments allowed between blocks
<events>
event information, as described previously
# further comments for each event
</events>
```

This is not `xml` code, since needs root element! (Andy Buckley)
Too generic tags; need to attach to Les Houches origin! (Jon Butterworth)
For the future we need some versioning information! (JB, TS)

Proposed document structure: today

```
<LesHouchesEvents version="1.0">
# comment lines can go here
<header>
<!-- only xml tags in this section, including for comments -->
</header>
# further comments allowed between blocks
<init>
initialization information, as described previously
# further comments on initialization
</init>
# further comments allowed between blocks
<events>
event information, as described previously
# further comments for each event
</events>
</LesHouchesEvents>
```

While `init` and `events` blocks are compulsory, `header` block could be optional. (Fabio Maltoni, Johan Alwall)

Flexible format?

Possibility to omit some fields, to save space? For instance:
Binary code in the header to tell which items will or will not appear in the particle lines, e.g. 1010001110000 for ID, 1st mother and momentum 3-vector only. (Bryan Webber)

My opinion:

- flexible = complex = error-prone
- do not allow people not to specify colours (often nontrivial)
- when critical, people do not need to use more space then needed:
  - ⋆ do not leave more than one blank between fields (⇒ not aligned)
  - ⋆ do not use higher precision than meaningful, e.g.
    `VTIMUP(I) SPINUP(I)` = ” 0. 9.”
    (former normal in hard processes, latter allowed for undefined)
- If disk space *really* is an issue, then we *might* agree not to store energy,
  since $E = \sqrt{m^2 + p^2}$ is numerically safe,
  while e.g. $m = \sqrt{E^2 - p^2}$ is not!
  (Spacelike particles can give problems, even have $E < 0$,
  but only appear as information lines, not actually used (?).)

Sophisticated xml structure?

Alexander Sherstnev and Lev Dudko, HepML team:

We don't agree to standardize fixed places of all parameters in the discussed proposal. The solution is very limited. E.g. generators will have to store numbers with no information. We propose to keep information about a format used in the event file as a set of XML tags in header. At first stage, only LHA-I info is stored in the `<events>` and `<init>` sections. All other information can be added to `<header>`.

In HepML a part of the header defines the parameter names and their positions in the correponding event file for each event record. Based on these rules, a parser automatically generates a format string. After that a SH generator can use the format string to read events from the file and fill the LHA common blocks or structures. In our approach you simply skip the parameters (in the format description) and the parser produces a shorter format string without these parameters. Due to flexibility this approach allows new parameters in event to be stored.

See `https://twiki.cern.ch/twiki/bin/view/Main`
`/ExampleOfHepMLLHAIEventFileOfME-SHLevelOfSimulation`

```
<?xml version="1.0" encoding="UTF-8"?>
<EventFile>
<header>
<run type=’lha1’>
    <beams>
        <beam number=’1’>
            <beamParticleTypeId name=’proton’ PDGid=’2212’/>
            <energy value=’980’ unit=’GeV’/>
            <PDF type=’collinear’>
                <pdfGroupId value=’0’/>
                <pdfSetId value=’7’/>
            </PDF>
        </beam>
        <!-- same for beam2 -->
    </beams>
    <Processes>
        <NumberOfProcesses value=’2’/>
        <Process number=’1’>
            <CrossSection unit=’pb’ value=’5.220106E+00’/>
            <CrossSectionerror unit=’pb’ value=’5.384128E-01’/>
            <MaxWeight value=’1.000000E+00’/>
            <ProcessListing value=’81’/>
        </Process>
        <!-- same for process 2 -->
    </Processes>
</run>
```

```
<rules type=’event’ version=’0.1’ type=’pythia’>
     <Empty Npositions=’1’/>
     <nup frm=’I6’/>
     <ProcessId frm=’I6’/>
     <EventWeight frm=’e14.6’/>
     <QCDScale frm=’e14.6’/>
     <QCDCoupling frm=’e14.6’/>
     <QEDCoupling frm=’e14.6’/>
     <EndOfLine/>
     <particle>
        <Empty Npositions=’1’/>
        <ParticleID frm=’I8’/>
        <ParticleStatusCode frm=’I5’/>
        <Mother N=’1’ frm=’I5’/><Mother  N=’2’ frm=’I5’/>
        <ColorChain>
            <ColorChainEnd type=’color’ frm=’I5’/>
            <ColorChainEnd type=’anticolor’ frm=’I5’/>
        </ColorChain>
        <momentum>
            <Px frm=’e18.10’/>
            <Py frm=’e18.10’/>
            <Pz frm=’e18.10’/>
            <E  frm=’e18.10’/>
            <M  frm=’e18.10’/>
        </momentum>
        <vtime frm=’e14.6’/>
        <spin frm=’e12.4’/>
        <EndOfLine/>
     </particle>
</rules>
</header>
```

Personal opinion on xml

- Industry-standard technology with many emerging tools . . .
  . . . but not really needed here.
- MCDB has separate system of articles maintained in `xml`;
  no competition or contradiction with that.
- If we raise the bar for newcomers too high
  there will be no useful agreement.
- Many people would want to see a proposal completely without `xml`,
  i.e. only with # comments; the current proposal is already
  a compromize in favour of the `xml` enthusiasts.
- Standard 1.0 should be to enforce *only* existing LHA structure.
- A version 2.0 could come later, with rich selection of `xml` tags.
  (Including phase space cuts, ME matching, . . . ?)
- Then let market decide whether people see the advantages.

Other issues

Wish to store pdf values event-by-event. (Sergey Slabospitsky, CMS)
Could introduce a few half-standardized “reserved” comment formats, e.g.

```
#pdf id1 id2 x1 x2 scalePDF xpdf1 xpdf2
```

e.g. `#pdf 21 2 1.234E-04 0.2314 10.13 15.32 0.9435`

Write information array by array, rather than particle by particle, so more efficient array memory access. (Steve Mrenna)
No, because:
1) C++ should not store array by array, but particle by particle
2) Array filling time negligible compared to generation time.

Changes to the existing format MUST be backwards compatible. New things can be added (such as new fortran common blocks), but files that were made with the old format must be still useable. In particular a variable's meaning cannot be allowed to change. (Ian Hinchliffe)
Yes, definitely; the meaning of all variables is unchanged!

Future

Need a few more rounds of discussion:

- simple mail among people who expressed interest?
- new mail list?
- HepML mailing list? (`project-hepml@cern.ch`)

Suggestion: for version 1.0: use simple mail + cc to HepML;
final version to whole mc4lhc mailing list.

Currently: Bryan Webber, Lynn Garren, Andy Buckley, Joey Huston, Stephen Mrenna, Fabio Maltoni, Johan Alwall, Peter Skands, Sergey Slabospitsky, Ian Hinchliffe, HepML team.