Monte Carlo Particle Lists: MCPL

Klinkby, Esben Bryndt

Publication date:
2016

Document Version
Publisher's PDF, also known as Version of record

Monte Carlo Particle Lists : MCPL

Neutrons cradle to grave workshop, SINE2020 GA, Coimbra, Portugal, 2016-09-06

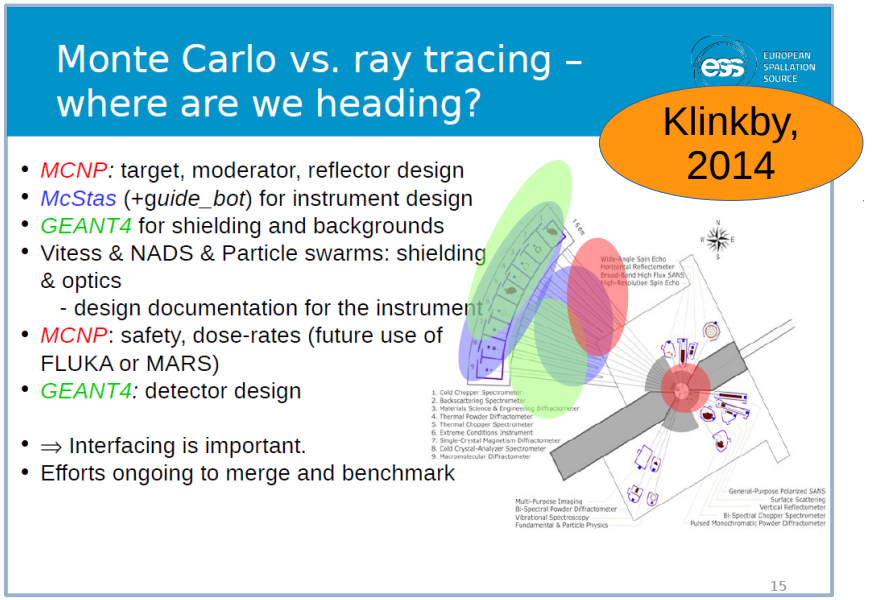
Thomas Kittelmann, ESS Detector Group
(thomas.kittelmann@esss.se)

MCPL developed with contributions from:

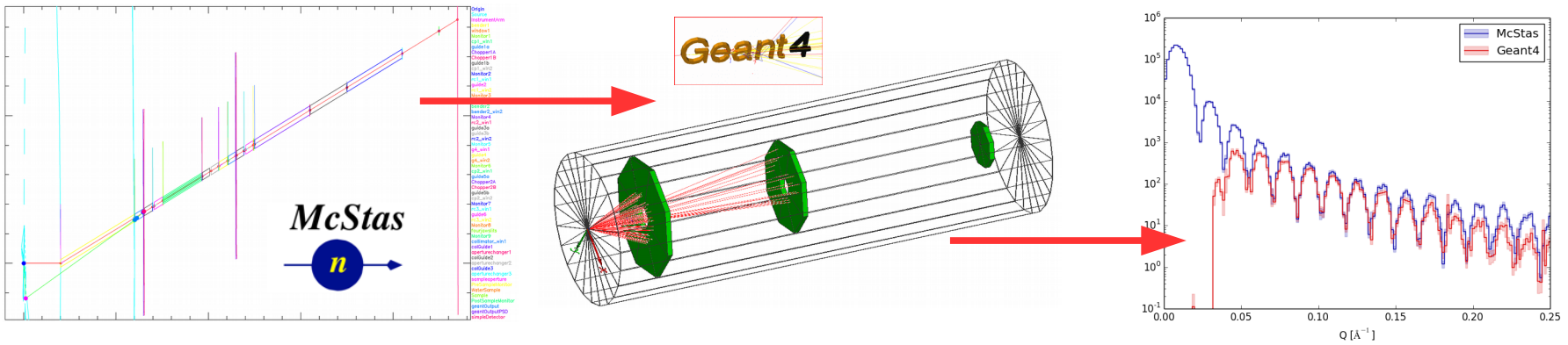
E. Klinkby (DTU), E. Knudsen (DTU), P. Willendrup (DTU, ESS), K. Kanaki (ESS), X. X. Cai (ESS, DTU)

Background / Motivation

- Many different applications in use at ESS for particle simulations.
- Desirable to be able to transfer particles between applications.
- Or reuse within a single application.



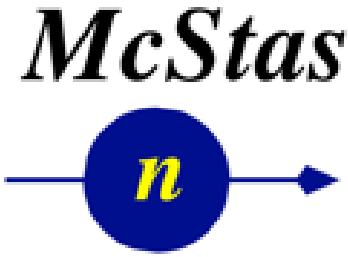
- For detector simulations in Geant4, we are interested in grabbing post-sample output of instrument simulations (usually McStas), and use those as a source.



- Or, grab background particles from MCNP or Geant4 simulations to study shielding and background issues.

How to store and transfer particles? By 2015 we had a jungle of custom solutions at ESS for just 3 apps...

NB: illustration here is surely incomplete...



- Use SSW
- Not between MCNPX & 6

- Custom McStas+MCNP code working one-way via SSW files.

Klinkby, Willendrup

- Custom McStas component?
- Reading SSW files in special format

Klinkby, Knudsen, Willendrup

- Adapted Fortran code for SSW => ASCII.
- Created custom C++ code to get into G4.
- Problems: Not all particle types worked, MCNP6 not supported, difficult work-flow.

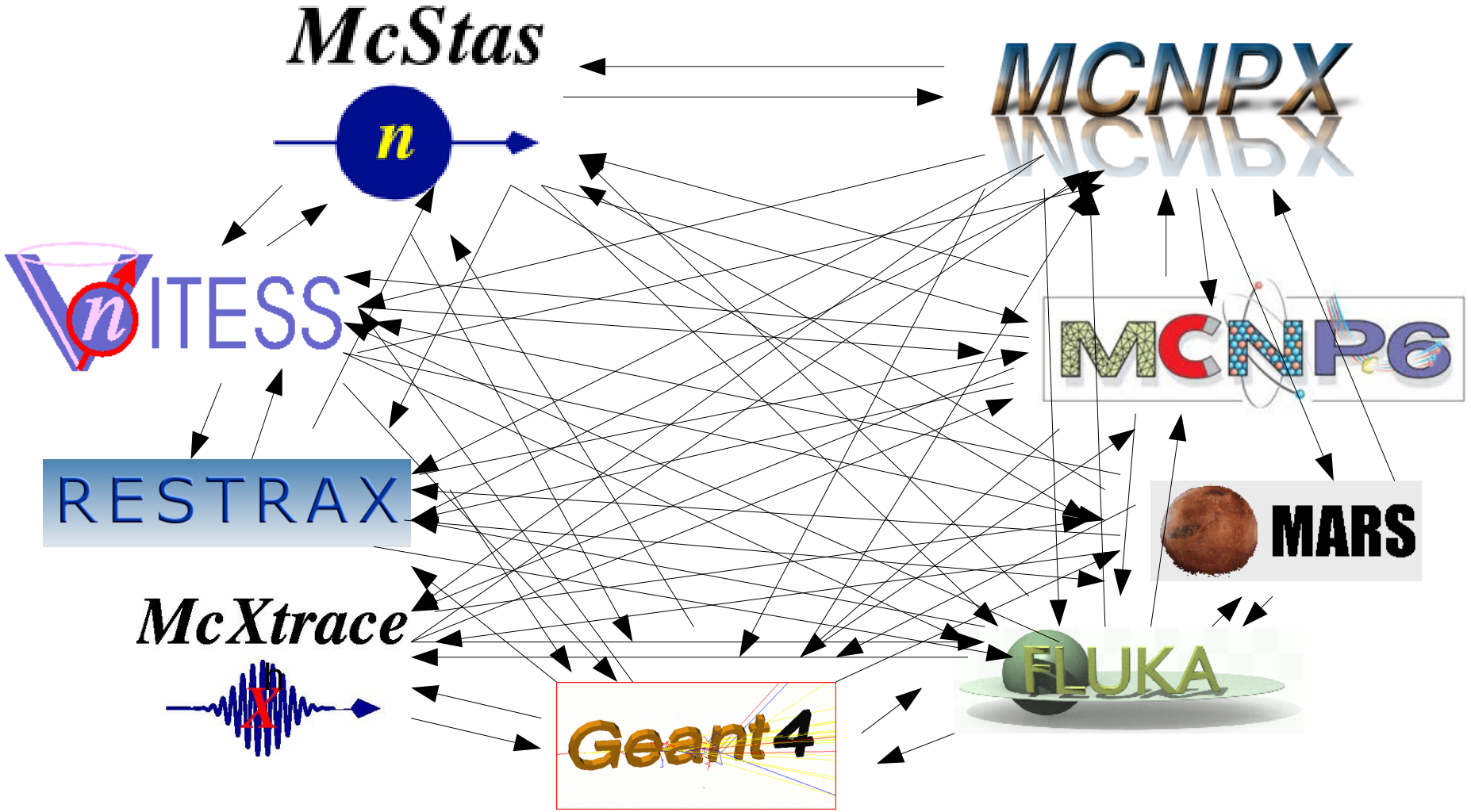
Klinkby, Kittelmann

- Used McStas MonitorND comp. => ASCII files
- Created custom C++ code to get into G4.
- Problems: MonitorND geometry, format non-constant, etc.

Kanaki, Kittelmann

- Saving particles for use within G4@ESS:
- ASCII, HDF5, GRIFF, ROOT
- Quick and dirty, non-portable, or badly suited formats...

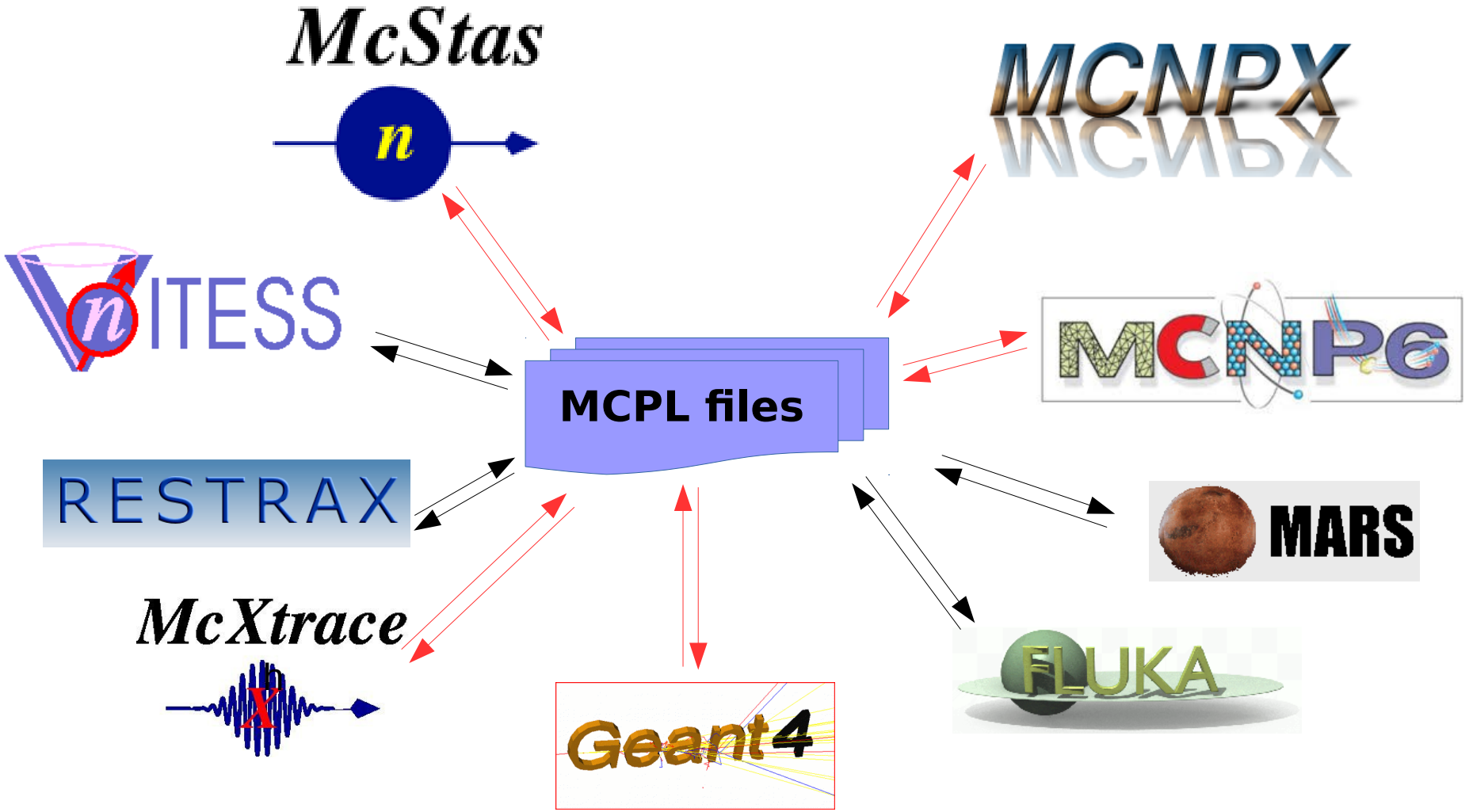
Consider more apps : The jungle gets impossibly tangled...



Disclaimer: Non-exhaustive list of applications...

The solution: A common interchange format.

MCPL: Monte Carlo Particle Lists



In red : already available now (Sep 2016).

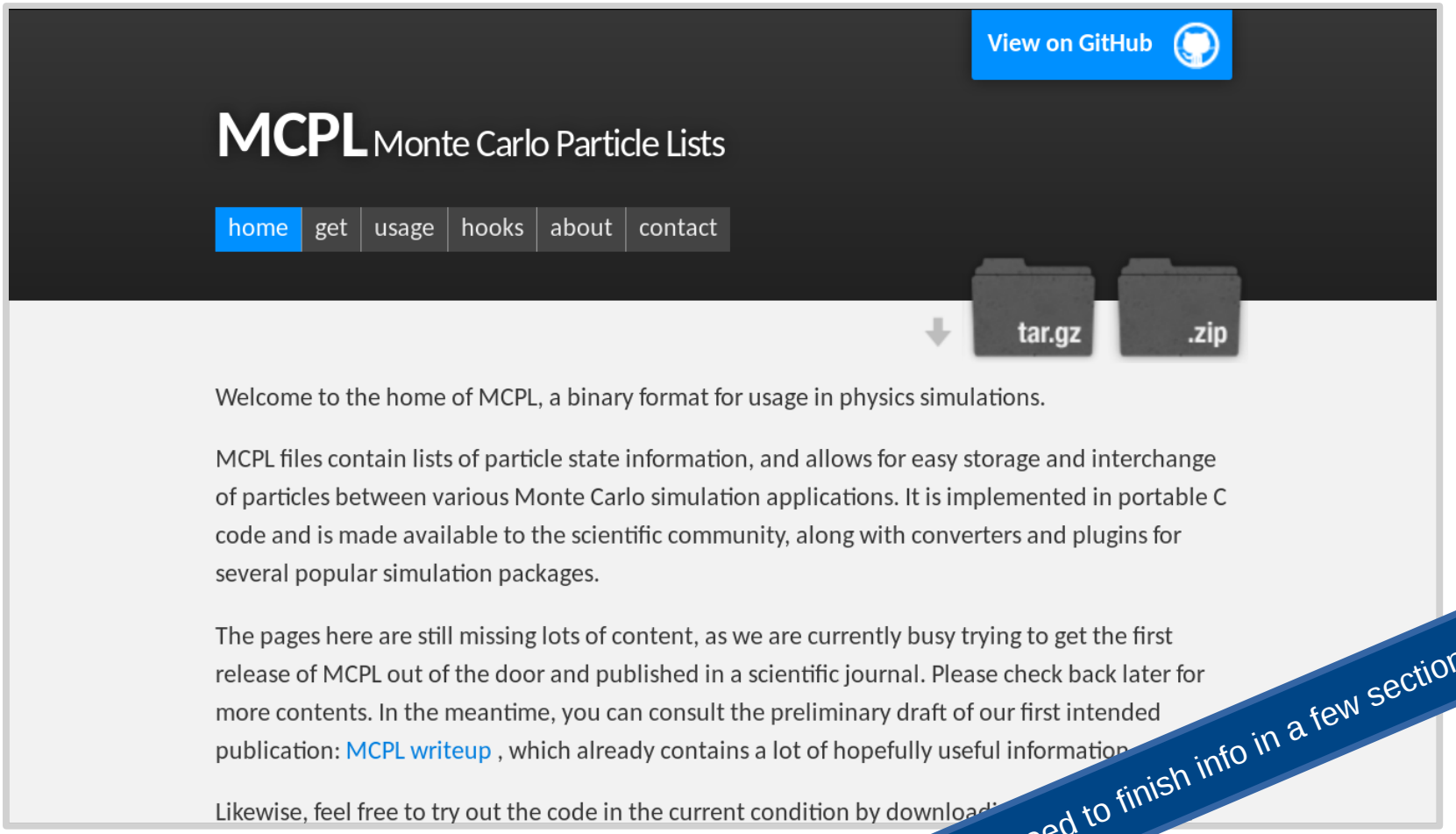
Disclaimer: Non-exhaustive list of applications...

What is MCPL?

MCPL : Monte Carlo Particle Lists

- It is a simple file-format. Each file contains a list of particles.
- The format is flexible: can contain a lot of information if needed, or can contain only minimal information if small file-size is important.
- It is easy to make code dealing with MCPL, so it is easy to make plugins & converters for the various Monte Carlo frameworks. End-users will simply use those converters.
- MCPL files can contain meta-data. This makes it possible to tell what data is in a file, where it came from, how it should be interpreted.
- MCPL comes with tools, such as for inspecting contents.

Official website & code @ GitHub: https://mctools.github.io/mcpl/



Still need to finish info in a few sections…

Paper describing MCPL in detail about to be submitted

Monte Carlo Particle Lists : MCPL

T Kittelmann[a,*], E Klinkby[b], E Knudsen[c], P Willendrup[c], X X Cai[a,b], K Kanaki[a]

[a] *European Spallation Source ERIC, Sweden*
[b] *DTU Nutech, Technical University of Denmark, Denmark*
[c] *DTU PHYSICS, Technical University of Denmark, Denmark*

Abstract

A binary format with lists of particle state information, for interchanging particles between various Monte Carlo simulation applications, is presented. Portable C code for file manipulation is made available to the scientific community, alo
with converters and plugins for several popular simulation packages.

Draft version available on MCPL website

Lots of details!
More than most end-users will need to know or care about :-)

Browsing an MCPL file with the MCPL tool

Just run: "`mcpltool <name-of-MCPL-file>`"

```
Opened MCPL file myfile.mcpl.gz:

  Basic info
    Format             : MCPL-2
    No. of particles   : 5037156
    Header storage     : 818 bytes
    Data storage       : 181337616 bytes

  Custom meta data
    Source             : "Geant4"
    Number of comments : 8
          -> comment 0 : "Created with the Geant4 MCPLWriter in the ESS/dgcode framework"
          -> comment 1 : "MPCLWriter volumes considered : ['RecordFwd']"
          -> comment 2 : "MPCLWriter steps considered : <at-volume-exit>"
          -> comment 3 : "MPCLWriter write filter : <unfiltered>"
          -> comment 4 : "MPCLWriter user flags : <disabled>"
          -> comment 5 : "MPCLWriter track kill strategy : <none>"
          -> comment 6 : "ESS/dgcode geometry module : G4StdGeometries/GeoSlab"
          -> comment 7 : "ESS/dgcode generator module : G4StdGenerators/SimpleGen"
    Number of blobs    : 2
          -> 74 bytes of data with key "ESS/dgcode_geopars"
          -> 231 bytes of data with key "ESS/dgcode_genpars"

  Particle data format
    User flags         : no
    Polarisation info  : no
    Fixed part. type   : no
    FP precision       : single
    Endianness         : little
    Storage            : 36 bytes/particle

index     pdgcode   ekin[MeV]       x[cm]       y[cm]       z[cm]          ux          uy          uz    time[ms]      weight
    0        2112  4.0061e-08     -11.518      -2.744          40    -0.60697   -0.093797     0.78917     0.22354           1
    1        2112     2.5e-08           0           0          40           0           0           1      0.1829           1
    2          22      7.7251      7.8603     -6.7903          40    0.072796    -0.20272     0.97653     0.33498           1
    3        2112  1.8481e-08     -21.168      4.4662          40    -0.70384      0.1485     0.69466     0.24732           1
    4          22       0.511      27.191      7.7111          40     0.12641   -0.034978     0.99136     0.13778           1
    5          22       0.031     -30.093      19.067          40     0.10979     0.84395     0.52507     0.27059           1
    6          22       1.592         -50      2.7616      27.847    -0.66425     0.66981     0.33186     0.27059           1
    7          22      1.4402      16.313     -15.255          40    0.062836    -0.14628     0.98724     0.11248           1
```

Browsing an MCPL file with the MCPL tool

Just run: "`mcpltool <name-of-MCPL-file>`"

```
Opened MCPL file myfile.mcpl.gz:

  Basic info
    Format             : MCPL-2
    No. of particles   : 5037156
    Header storage     : 818 bytes
    Data storage       : 181337616 bytes

  Custom meta data
    Source             : "Geant4"
    Number of comments : 8
          -> comment 0 : "Created with the Geant4 MCPLWriter in the ESS/dgcode framework"
          -> comment 1 : "MPCLWriter volumes considered : ['RecordFwd']"
          -> comment 2 : "MPCLWriter steps considered : <at-volume-exit>"
          -> comment 3 : "MPCLWriter write filter : <unfiltered>"
          -> comment 4 : "MPCLWriter user flags : <disabled>"
          -> comment 5 : "MPCLWriter track kill strategy : <none>"
          -> comment 6 : "ESS/dgcode geometry module : G4StdGeometries/GeoSlab"
          -> comment 7 : "ESS/dgcode generator module : G4StdGenerators/SimpleGen"
    Number of blobs    : 2
          -> 74 bytes of data with key "ESS/dgcode_geopars"
          -> 231 bytes of data with key "ESS/dgcode_genpars"

  Particle data format
    User flags         : no
    Polarisation info  : no
    Fixed part. type   : no
    FP precision       : single
    Endianness         : little
    Storage            : 36 bytes/particle
```

Columns of particle data (1 row = 1 particle)
In this file: No *userflags* or *polarisation*

index	pdgcode	ekin[MeV]	x[cm]	y[cm]	z[cm]	ux	uy	uz	time[ms]	weight
0	2112	4.0061e-08	-11.518	-2.744	40	-0.60697	-0.093797	0.78917	0.22354	1
1	2112	2.5e-08	0	0	40	0	0	1	0.1829	1
2	22	7.7251	7.8603	-6.7903	40	0.072796	-0.20272	0.97653	0.33498	1
3	2112	1.8481e-08	-21.168	4.4662	40	-0.70384	0.1485	0.69466	0.24732	1
4	22	0.511	27.191	7.7111	40	0.12641	-0.034978	0.99136	0.13778	1
5	22	0.031	-30.093	19.067	40	0.10979	0.84395	0.52507	0.27059	1
6	22	1.592	-50	2.7616	27.847	-0.66425	0.66981	0.33186	0.27059	1
7	22	1.4402	16.313	-15.255	40	0.062836	-0.14628	0.98724	0.11248	1

Browsing an MCPL file with the MCPL tool

Just run: "`mcpltool <name-of-MCPL-file>`"

```
Opened MCPL file myfile.mcpl.gz:

  Basic info
    Format             : MCPL-2
    No. of particles   : 5037156
    Header storage     : 818 bytes
    Data storage       : 181337616 bytes

  Custom meta data
    Source             : "Geant4"
    Number of comments : 8
          -> comment 0 : "Created with the Geant4 MCPLWriter in the ESS/dgcode framework"
          -> comment 1 : "MPCLWriter volumes considered : ['RecordFwd']"
          -> comment 2 : "MPCLWriter steps considered : <at-volume-exit>"
          -> comment 3 : "MPCLWriter write filter : <unfiltered>"
          -> comment 4 : "MPCLWriter user flags : <disabled>"
          -> comment 5 : "MPCLWriter track kill strategy : <none>"
          -> comment 6 : "ESS/dgcode geometry module : G4StdGeometries/GeoSlab"
          -> comment 7 : "ESS/dgcode generator module : G4StdGenerators/SimpleGen"
    Number of blobs    : 2
          -> 74 bytes of data with key "ESS/dgcode_geopars"
          -> 231 bytes of data with key "ESS/dgcode_genpars"

  Particle data format
    User flags         : no
    Polarisation info  : no
    Fixed part. type   : no
    FP precision       : single
    Endianness         : little
    Storage            : 36 bytes/particle
```

index	pdgcode	ekin[MeV]	x[cm]	y[cm]	z[cm]	ux	uy	uz	time[ms]	weight
0	2112	4.0061e-08	-11.518	-2.744	40	-0.60697	-0.093797	0.78917	0.22354	1
1	2112	2.5e-08	0	0	40	0	0	1	0.1829	1
2	22	7.7251	7.8603	-6.7903	40	0.072796	-0.20272	0.97653	0.33498	1
3	2112	1.8481e-08	-21.168	4.4662	40	-0.70384	0.1485	0.69466	0.24732	1
4	22	0.511	27.191	7.7111	40	0.12641	-0.034978	0.99136	0.13778	1
							0.84395	0.52507	0.27059	1
							0.66981	0.33186	0.27059	1
							-0.14628	0.98724	0.11248	1

Columns of particle data (1 row = 1 particle)
In this file: No *userflags* or *polarisation*

PDG codes: 2112 = neutron, 22 = gamma
More at http://pdg.lbl.gov/2015/reviews/rpp2015-rev-monte-carlo-numbering.pdf

Browsing an MCPL file with the MCPL tool

Just run: "mcpltool <name-of-MCPL-file>"

```
Opened MCPL file myfile.mcpl.gz:

  Basic info
    Format              : MCPL-2
    No. of particles    : 5037156
    Header storage      : 818 bytes
    Data storage        : 181337616 bytes

  Custom meta data
    Source              : "Geant4"
    Number of comments : 8
          -> comment 0 : "Created with the Geant4 MCPLWriter in the ESS/dgcode f
          -> comment 1 : "MPCLWriter volumes considered : ['RecordFwd']"
          -> comment 2 : "MPCLWriter steps considered : <at-volume-exit>"
          -> comment 3 : "MPCLWriter write filter : <unfiltered>"
          -> comment 4 : "MPCLWriter user flags : <disabled>"
          -> comment 5 : "MPCLWriter track kill strategy : <none>"
          -> comment 6 : "ESS/dgcode geometry module : G4StdGeometries/GeoSlab"
          -> comment 7 : "ESS/dgcode generator module : G4StdGenerators/SimpleGe
    Number of blobs     : 2
          -> 74 bytes of data with key "ESS/dgcode_geopars"
          -> 231 bytes of data with key "ESS/dgcode_genpars"

  Particle data format
    User flags          : no
    Polarisation info   : no
    Fixed part. type    : no
    FP precision        : single
    Endianness          : little
    Storage             : 36 bytes/particle
```

Custom meta-data

- This file is from ESS-DG Geant4
- Comments reminding us of setup used to create file
- Binary "blobs" keep more complete configuration details (here ESS-DG geo/gen parameters, could be McStas instrument file or MCNP input deck).

Columns of particle data (1 row = 1 particle)
In this file: No *userflags* or *polarisation*

index	pdgcode	ekin[MeV]	x[cm]	y[cm]	z[cm]	ux	uy	uz	time[ms]	weight
0	2112	4.0061e-08	-11.518	-2.744	40	-0.60697	-0.093797	0.78917	0.22354	1
1	2112	2.5e-08	0	0	40	0	0	1	0.1829	1
2	22	7.7251	7.8603	-6.7903	40	0.072796	-0.20272	0.97653	0.33498	1
3	2112	1.8481e-08	-21.168	4.4662	40	-0.70384	0.1485	0.69466	0.24732	1
4	22	0.511	27.191	7.7111	40	0.12641	-0.034978	0.99136	0.13778	1
							0.84395	0.52507	0.27059	1
							0.66981	0.33186	0.27059	1
							-0.14628	0.98724	0.11248	1

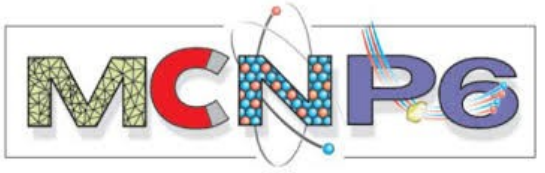
PDG codes: 2112 = neutron, 22 = gamma
More at http://pdg.lbl.gov/2015/reviews/rpp2015-rev-monte-carlo-numbering.pdf

Using MCPL with Geant4

- Provided as C++ classes extending G4 interfaces, since that is the usual M.O. for working with Geant4.
  - MCPL as input through custom G4VUserPrimaryGeneratorAction (G4MCPLGenerator).
  - MCPL as output through Custom sensitive detector (G4MCPLWriter) capturing particles entering selected volumes.
  - Many possibilities for fine-tuning behaviour.
- Users of the ESS detector group Geant4-framework don't need to deal with C++ classes, but can simply specify desired input/output behaviour with a few lines of python or at the command line.

More info on MCPL website & in section 3.1 of writeup!

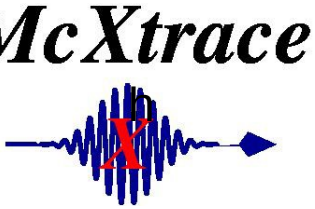
Using MCPL with MCNP



- Provided as two dependency-free command-line applications written in portable C, for converting between MCNP Surface Source Read/Write files (aka SSW files aka WSSA files) and MCPL:
  - **mcpl2ssw** and **ssw2mcpl**
- For instance run: **`ssw2mcpl <my-ssw-file> output.mcpl`**
- Easy to get access to one of those commands: Download a single file from the MCPL website and compile it into the executable.
- Supports MCNP5, MCNPX & MCNP6 (despite incompatible SSW formats).

More info on MCPL website & in section 3.2 of writeup!

Using MCPL with McStas or McXtrace



- MCPL_output and MCPL_input components were already included upstream.
- For output, just add two lines in your instrument file at the appropriate position (for instance, right after the sample component):

```
COMPONENT mcplout = MCPL_output(filename="myfile")
AT(0,0,0) RELATIVE PREVIOUS
```

- This captures into myfile.mcpl.gz the full state of all neutrons as they leave the previous component (with coordinates relative to that component).
- Using particles in an MCPL files as a *source* in McStas is equally simple.
- Also works when running McStas with MPI.
- Example instruments using are included with McStas:
  - mcstas-comps/examples/Test_MCPL_output.inst
  - mcstas-comps/examples/Test_MCPL_input.inst

More info on MCPL website & in section 3.3 of writeup!

NOTE: The MCPL code is already part of McStas 2.3, but a few bugs were fixed late, so need to copy a fixed version of MCPL_output.comp into your rundir. From McStas 2.4 and McXtrace 1.3, everything will work out of the box.

C-code for reading MCPL file

Note: This is shown in case someone is wondering if they could implement converters for their own application. End-users should normally just activate pre-written converters & plugins for their applications

Listing 1: Simple example for looping over all particles in an existing MCPL file

```c
#include "mcpl.h"

void example()
{
  mcpl_file_t f = mcpl_open_file("myfile.mcpl");

  const mcpl_particle_t* p;

  while ( ( p = mcpl_read(f) ) ) {

    /* Particle properties can here be accessed
       through the pointer "p":

       p->pdgcode
       p->position[k] (k=0,1,2)
       p->direction[k] (k=0,1,2)
       p->polarisation[k] (k=0,1,2)
       p->ekin
       p->time
       p->weight
       p->userflags
    */

  }

  mcpl_close_file(f);
}
```

C-code for creating MCPL file

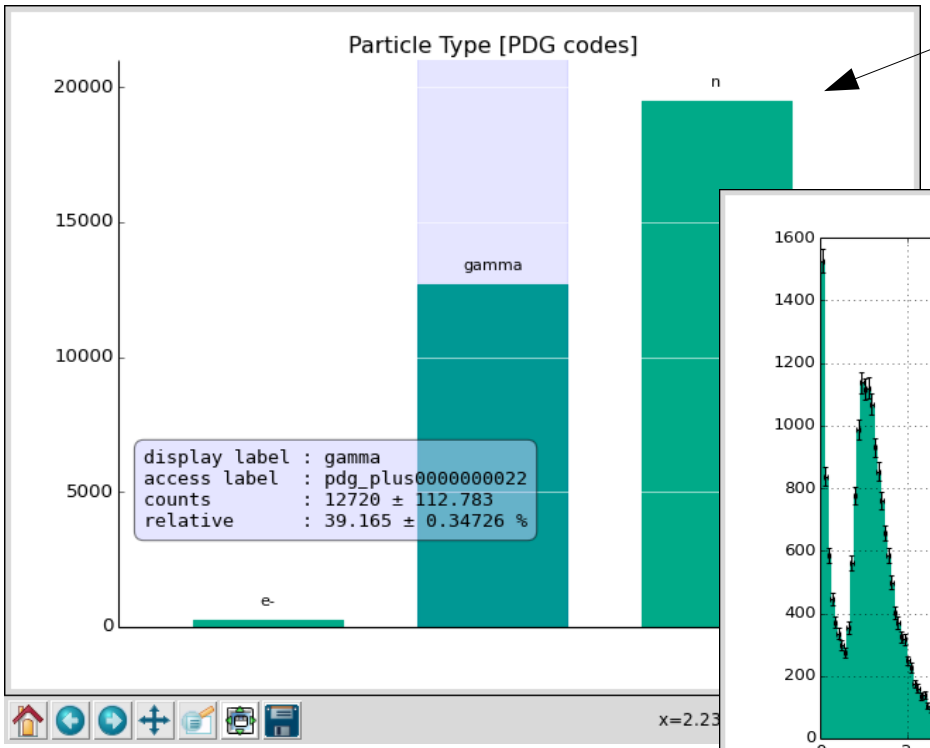
Note: This is again shown in case someone is wondering if they could implement converters for their own application...

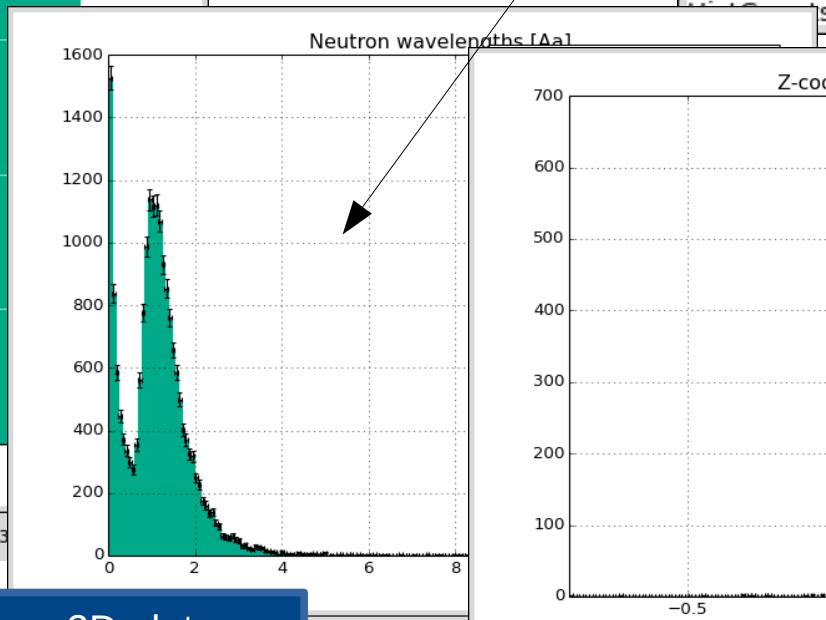
Listing 2: Simple example for creating an MCPL file with 1000 particles.

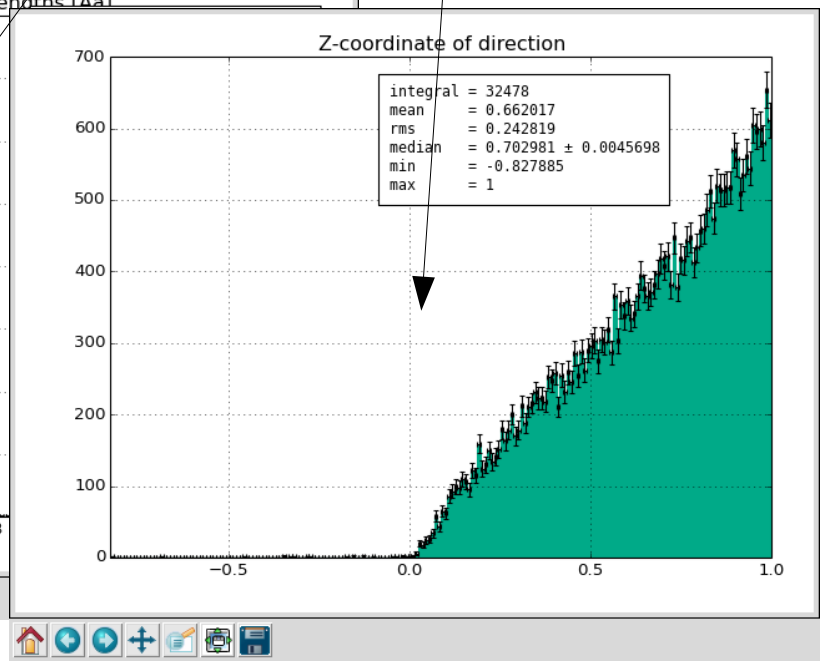
```
#include "mcpl.h"

void example()
{
  mcpl_outfile_t f = mcpl_create_outfile("myfile.mcpl");
  mcpl_hdr_set_srcname(f,"MyAppName-1.0");

  /* Tune file options or add custom comments or
     binary data into the header:

     mcpl_enable_universal_pdgcode(f,myglobalpdgcode);
     mcpl_enable_userflags(f);
     mcpl_enable_polarisation(f);
     mcpl_enable_doubleprec(f);
     mcpl_hdr_add_comment(f,"Some comment.");
     mcpl_hdr_add_data(f,"mydatakey",
                       my_datalength, my_databuf)
  */

  mcpl_particle_t* p = mcpl_get_empty_particle(f);

  int i;
  for ( i = 0; i < 1000; ++i ) {

    /* The following particle properties must
       always be set here:

       p->position[k] (k=0,1,2)
       p->direction[k] (k=0,1,2)
       p->ekin
       p->time
       p->weight

       These should also be set when required by
       file options:

       p->pdgcode
       p->userflags
       p->polarisation[k] (k=0,1,2)
    */

    mcpl_add_particle(f,p);
  }

  mcpl_close_outfile(f);
}
```

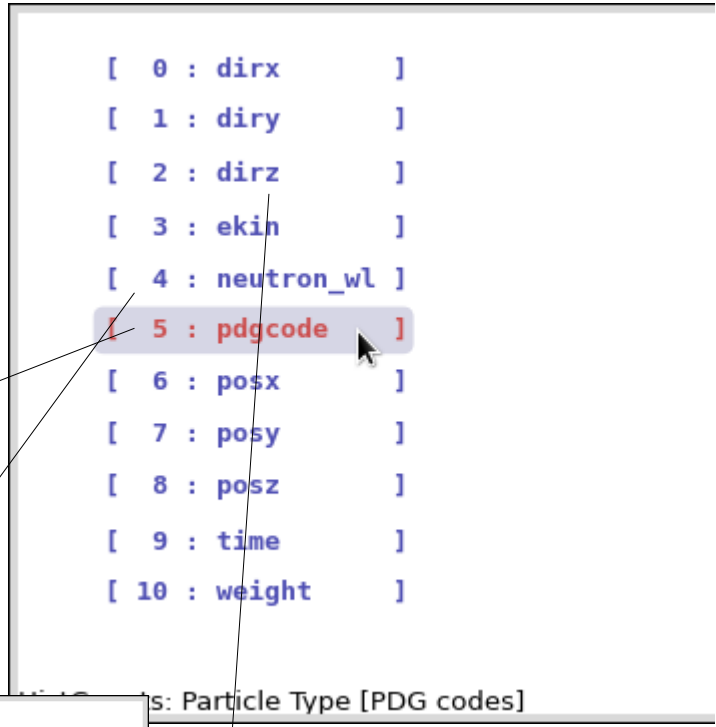
In the ESS detector group, we have many extra MCPL tools which we hope to make available externally as well… Example shown here is a "browser".

```
$> ess_mcplextra_browse myfile.mcpl.gz
```









Can also produce 2D plots of variable correlations...

And impose filters, to study subset of particles

ESS DG framework reference:
Kittelmann, et al. CHEP 2013.
doi:10.1088/1742-6596/513/5/052022

Summary and outlook

- Collaboration between ESS detector group (focus:Geant4), McStas developers & the ESS target group (focus:MCNP), have resulted in a new standard particle interchange format.
- It can be (and is) used for serious studies already now!
- We hope to be able to provide more MCPL tools in the future.
- Still a few loose ends to tidy up:
  - Several sections on MCPL website needs more contents.
  - Submit publication (this week!)
- We welcome any application-specific experts who might be interested in extending the list of MCPL-aware applications from the current (G4+MCNP+McStas). **Get in touch if you are interested!**

Additional material

Meta-data in MCPL header

File header information	
Field	*Description*
File type magic number 0x4d43504c ("MCPL")	All MCPL files start with this 4-byte word.
Version	File format version.
Endianness	Whether numbers in file are in little- or big-endian format.
Number of particles in file	64 bit integer.
Flag : Particles have polarisation info	If false, all loaded particles will have polarisation vectors (0,0,0).
Flag : Particles have "userflags" field	If false, all loaded particles will have userflags 0x00000000.
Flag : Particle info use double-precision	If true, floating points storage use double-precision.
Global pdgcode	If this 32 bit integer is non-zero, all loaded particles will have this pdgcode.
Source name	String indicating the application which created the MCPL file.
Comments	A variable number of comments (strings) added at file creation.
Binary blobs	A variable number of binary data blobs, indexed by keys (strings). This allows arbitrary custom data to be embedded.

Table 1: Information available in the header section of `MCPL` files.

Reference: C-code for extracting subset of particles from one MCPL file into a new one

Listing 3: Example extracting low-energy neutrons (pdgcode 2112) from an MCPL file.

```c
#include "mcpl.h"

void example() {

  /* open files, transfer meta-data, add comment */

  mcpl_file_t fi = mcpl_open_file("myfile.mcpl");
  mcpl_outfile_t fo = mcpl_create_outfile("new.mcpl");
  mcpl_transfer_metadata(fi, fo);
  mcpl_hdr_add_comment(fo,"Extracted neutrons with ekin<0.1MeV");

  /* transfer selected particles */

  const mcpl_particle_t* particle;
  while ( ( particle = mcpl_read(fi) ) ) {
    if ( particle->pdgcode == 2112 && particle->ekin < 0.1 )
      mcpl_add_particle(fo,particle);
  }

  /* finish up */

  mcpl_closeandgzip_outfile(fo);
  mcpl_close_file(fi);
}
```