

Status of IAEAphsp example

Miguel A. Cortés-Giraldo^(*), C. G. Okolinta

*Dep. Atomic, Molecular and Nuclear Physics
Universidad de Sevilla (Spain)*

^(*) miancortes@us.es

28th Geant4 Collaboration Meeting


Hokkaido University
Sapporo (Japan), September 26, 2023.



Background

➤ **IAEAphsp**: Standardized format to use phase-space files produced from different codes.

 <http://www-nds.iaea.org/phsp>

- International Advisory Committee (IAC)
 - R. Jeraj
 - I. Kawrakow
 - C.-M. Ma
 - D.W.O. Rogers
 - F. Sanchez-Doblado
 - J. Sempau
 - J. Seuntjens
 - J.V. Siebers
 - P. Andreo
- Mailing Lists
 - Send mail to all members of the IAC
 - Register to the IAEA PHSP mailing list
- Medical Portal
 - Atomic and nuclear data for medical applications
- IAEA NAPC/NDS
 - Nuclear Data Section
- IAEA NAHU/DMRP
 - 

Phase-space database for external beam radiotherapy

IAEA NAPC Nuclear Data Section
IAEA NAHU Dosimetry and Medical Radiation Physics Section

Project Officer: [Roberto Capote](#)

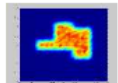
Objective: To build a database and disseminate representative [phase-space data](#) of accelerators and Co-60 units used in medical radiotherapy by compiling existing data that have been properly validated.

NEWS

[Dec 2009: Geant-4 interface to read/write the IAEA format released on December 14, 2009.](#)


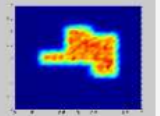


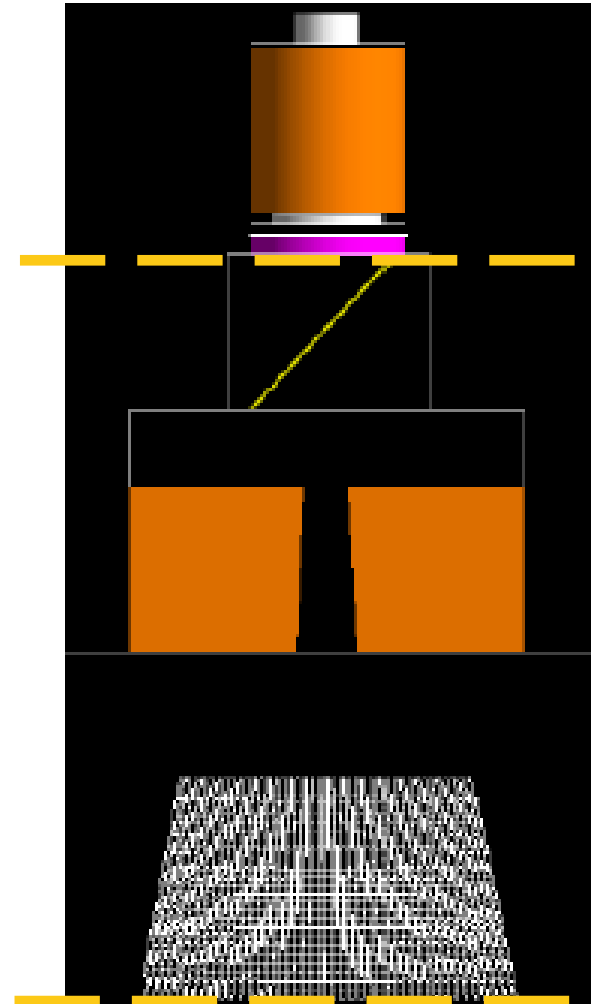
How to produce and submit phase-space data: The IAEA phsp format was designed to cover both phase-space files and event generators (see [phsp contents](#)). We have implemented the IAEA phsp format in a set of [read/write routines](#) (Updated: May 2011, see [readme file](#)). Native IAEA phsp format is available in EGSnrc and PENELOPE Monte Carlo codes. Geant4 interface to use the native IAEA phsp format is also [available](#). Once the validated phsp data is produced and documentation is published, [you may submit your phsp for review](#) using the [upload link here](#).



How to download phase-space data: You have to select a phsp data type among [Co-60 source](#), [linac electron](#) or [linac photon](#) phsps. For photon and electron PHSPs you may download the header first to decide which data you want to retrieve. Once decided you should download the PHSP data from the corresponding sub-directory. Please note that the first time access to the selected subdirectory could be slow.

Both the PHSP data and header should be present for the PHSP data to be accessible !

- Tech. Report IAEA-NDS-0484
 - 
- PHSP format
 - List of PHSP variables
- PHSP Header
 - How to fill header ...
- PHSP upload
 - Upload files
- PHSP to review
 - Files to review
- PHSP database
 - 
 - 1. Co-60 phsps
 - 2. Photon phsps
 - 3. Electron phsps



M.A. Cortés-Giraldo et al., IJRB 88: 200-8 (2012)

IAEAphsp code – current features

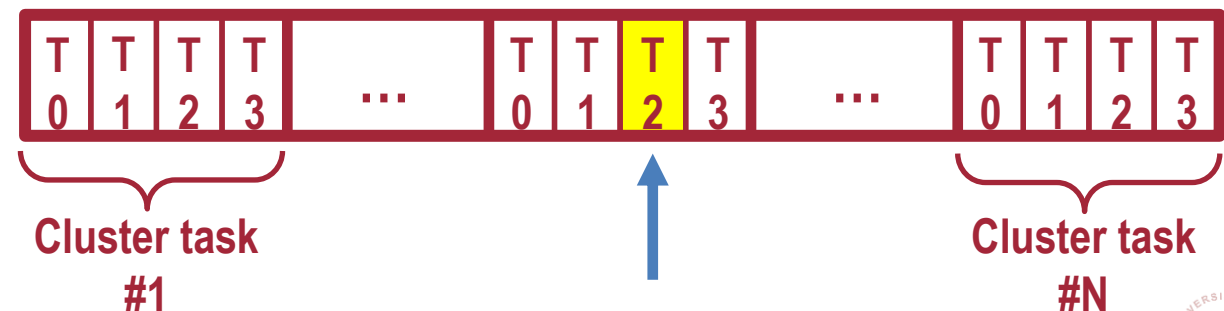
3

- **Focus:** to show **clearly** how to work with IAEAphsp files.
 - Very simple geometry & scoring (mostly to show that this works!)
- A controlled IAEAphsp file incorporated for testing purpose.
 - Just 1000 particles (200 of each kind: e-, e+, gamma, neutron, proton)
 - Dynamic variables are given values calculated from position within the file.
 - Energy, position, incremental history number (*n_stat*), etc.
- Compiled with latest version of IAEAphsp routines (sep-2013):

<code>h</code> iaea_config.h	
<code>h</code> iaea_header.h	<code>C++</code> iaea_header.cc
<code>h</code> iaea_phsp.h	<code>C++</code> iaea_phsp.cc
<code>h</code> iaea_record.h	<code>C++</code> iaea_record.cc
<code>h</code> utilities.h	<code>C++</code> utilities.cc
- **G4IAEAphspReader** class finished and tested.
 - **Messenger** class is also available.

G4IAEAphspReader MT class

- Derived from **G4VPrimaryGenerator**.
 - G4IAEAphspReader are **thread-local** objects.
- Object dynamically created in **ActionInitialization::Build()**
 - This was (for me!) the cleanest solution.
 - The file name has to be set prior object creation (no “empty” phsp reader is allowed).
- Each thread-local object defines a **source_id** defined internally in IAEA routines’. (**MAX_NUM_SOURCES=30**)
 - Currently, up to 30 threads are admitted.
 - But each **fstream** is associated to a **source_id** (either **reading** or **writing** a phsp file!).
 - The limit is lower when using a sources “slot” to write a phsp file.



Current status & next steps

5

- Currently working on **G4IAEAphspWriter** class.
 - Cleaning up code and redesigning to make it MT-compliant.
- To decide between these approaches:
 1. Consider the G4IAEAphspWriter objects as **thread local**.
 - Each thread produces a file, merge them at end of run.
 - Remember that each G4IAEAphspWriter object will 'eat' one source_id.
 2. Consider a **shared** G4IAEAphspWriter object.
 - Particles dumped in bunches from worker threads to G4IAEAphspWriter.
 - G4IAEAphspWriter object receives **N** particles to dump it into the phsp file.
 - The bunch size should be set by user. Optimal value of **N** depends on use case.



This work receives funding from Grant PID2021-123879OB-C21 funded by MCIN/AEI/10.13039/501100011033 and by “ERDF A way of making Europe”, European Union.



@ miancortes@us.es

