Common Turnkey Software Stack (KEY4HEP)

An AIDA++ Advanced Software Task Proposal

2019-06-17

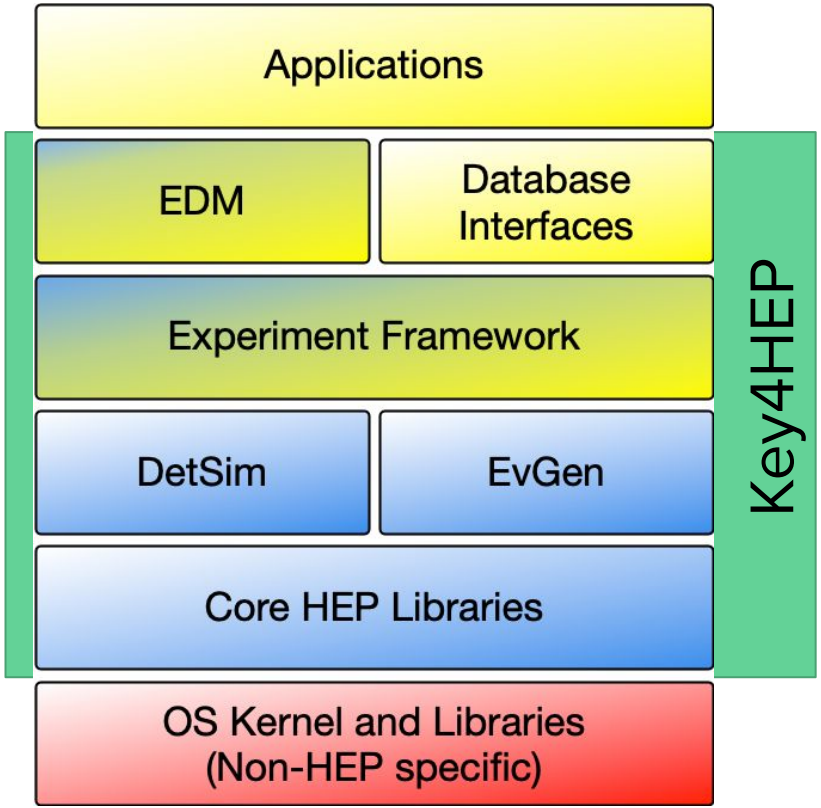
Motivation

- **Future detector studies critically rely on well-maintained software stacks** to model detector concepts and to understand a detector's limitations and physics reach
- We have a scattered landscape of specific software tools on the one hand and integrated frameworks tailored for a specific experiment on the other hand
- Aim at a low-maintenance common stack for FCC, ILC/CLIC, CEPC with ready to use "plug-ins" to develop detector concepts
- Reached **consensus among all communities** for future colliders to develop a common **turnkey** software stack

Vision

- Put together a stack of the software packages covering the different domains
  - most commonly used, avoiding as much as possible functionality overlaps
- The turnkey stack connects and extends the individual packages to enable a complete data processing framework
  - converting a set of disconnected packages into a ‘turnkey’ system
- and should be
  - easy to use: for librarians, developers, users
  - easy to set up
  - easy to deploy
  - easy to extend
  - full of functionality
  - plenty of examples for simulation and reconstruction of detectors

Content of the Key4HEP Stack



List of components:

- HEP de-facto standard: ROOT, Geant4, HepMC, ...
- New HEP libraries: VecCore, VecMath, VecGeom,...
- Externals: Boost, GSL, Eigen, ...
- EDM and Geo libraries: DD4Hep, PODIO, ...
- Rec/Tracking libraries: ACTS, ...
- Framework: Gaudi/Marlin

Event Data Model (EDM)

- Sharing a concrete EDM is a huge plus for reusing algorithms and applications
  - Very positive experience from ILC/CLIC
- Defining the EDM with a tool like PODIO is another big plus
  - Full flexibility on the implementation (the **what** versus the **how**)
- We could have a ‘base’ EDM which is mandatory to be understood by all common components and ‘extensions’ for experiment special cases
- Do we agree that we need to define a common EDM?
  - EDM4HEP (POD future collider EDM)

Framework

- The software stack **should include one and only one** data processing framework
  - Without this condition we cannot provide a ‘turnkey’ stack
- What do we need from a framework?
  - Functionally complete (conditions data, geometry, edm, ...)
  - Portable to various computing resources (Cloud, HPC, ...), architectures (x86, ARM,...) and accelerators (FPGA, GPU,...)
  - Concurrency (task-oriented)
- The best candidate so far: Gaudi

Stakeholders