



# The Software Eco-System

Key4hep

---

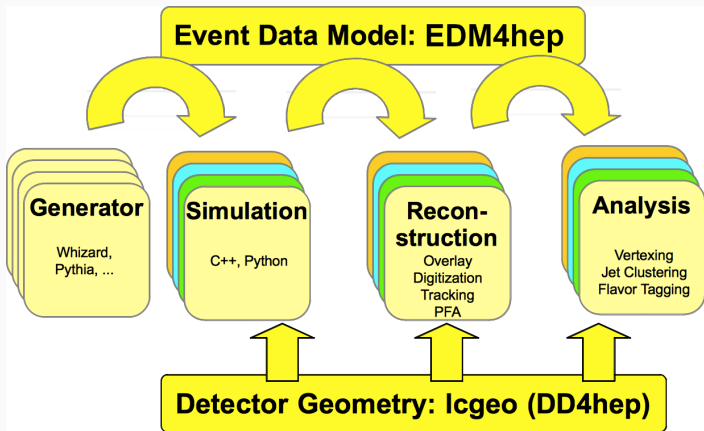


This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 101004761.

Thomas Madlener  
for the Key4hep developers  
ECFA workshop on  $e^+/e^-$   
Higgs/EW/Top Factories

Oct 6, 2022

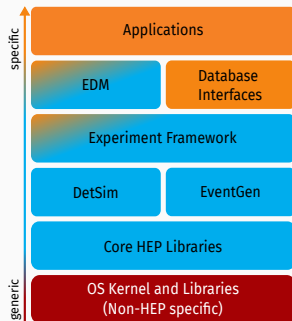
# From generation to analysis - the general workflow



- Many steps involved from generating events to analyzing them
- Hundreds of SW packages
  - Building & deploying
  - Consistency
  - Reproducibility
- Try to give an overview of the **Key4hep** SW ecosystem

# Key4hep - A (very) brief introduction

- Future detector studies rely on well maintained software for studying their potential
- Maintenance of a consistent HEP SW stack is non-trivial
  - Ecosystem of interacting components
- Sharing the burden allows everybody to reap the benefits
  - Make best use of scarce (human) resources
- **Regular contributions from ILC, CLIC, FCC, CEPC, (EIC), ...**
- Support from major R&D initiatives
  - [CERN R&D for Future Experiments](#), [AIDAinnova WP12](#), ECFA



# Key4hep goals

- Provide and maintain a consistent SW stack that allows to do physics studies for **all projects**
- Ensure interoperability of the necessary building blocks
- Reuse existing solutions where possible
  - A lot of experience from LHC experiments and LC communities
- Focus new developments on EW/Higgs factory specifics
- Share knowledge, processes, workflows and resources
  - Best practices, tutorials, documentation, ...

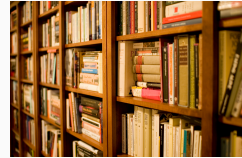
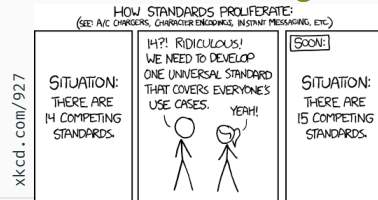


Photo by Stewart B. / [CC-BY](#)

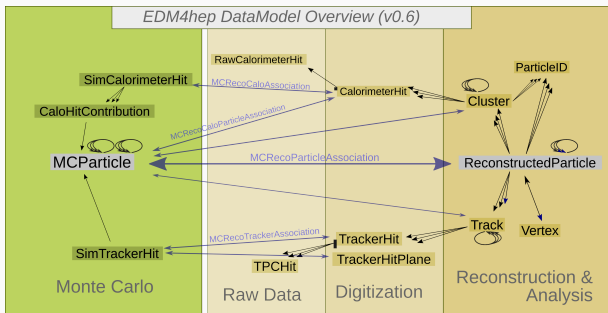



## Non-goal

- Develop and maintain project specific software and workflows



# EDM4hep - The common EDM for Key4hep



- Interoperability of different components requires a “lingua franca”
- Based on **LCIO** and **FCC-edm**
  - Focus on usability in analysis
- Generated via **podio** ()
  - Schema evolution available soon
  - Now supports **prototyping of new datatypes**
- Currently finalizing v1
  - Backwards compatible from then

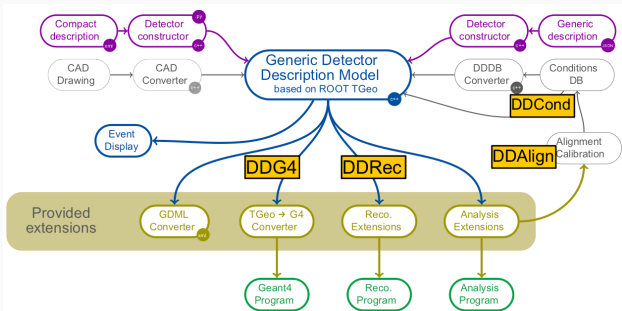
 [key4hep/EDM4hep](https://github.com/key4hep/EDM4hep)

[edm4hep.web.cern.ch](https://edm4hep.web.cern.ch)

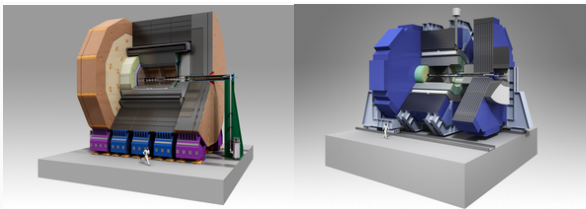
 [AIDAsoft/podio](https://github.com/AIDAsoft/podio)

# DD4hep - Detector description

See [A. Sailer's talk](#) for the details



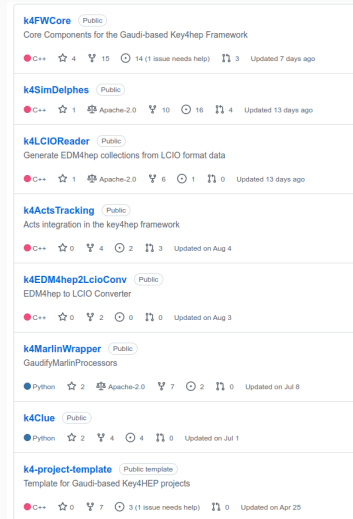
- Originally for LC but targeting all of HEP from the start (AIDA<sup>2020</sup>)
- Complete detector description
- Simulation, reconstruction, analysis
- “Industry standard”
  - ILC, CLIC, FCC, CEPC, EIC, LHCb, CMS, ...



# Experiment framework - Conducting all the different pieces

- Key4hep has adopted *Gaudi* as its experiment framework
  - Originally developed by LHCb, used by ATLAS, FCCSW
  - “Battle-proven” from LHC data taking
  - Several (legacy) “flavors”
- **k4FWCore** - core functionality
  - Data service for EDM4hep
- Dedicated packages for different tasks
- Main guideline: **Use EDM4hep for event data and DD4hep for detector description**

 [key4hep](https://github.com/key4hep) github org



The screenshot displays the GitHub repository page for the Key4hep organization. It lists several repositories with their respective languages, star counts, forks, issues, and update dates:

- k4FWCore** (Public): Core Components for the Gaudi-based Key4hep Framework. C++ (4 stars, 15 forks, 14 issues). Updated 7 days ago.
- k4SimDelphes** (Public): C++ (1 star, Apache-2.0 license, 10 forks, 16 issues). Updated 13 days ago.
- k4LCIOReader** (Public): Generate EDM4hep collections from LCIO format data. C++ (1 star, Apache-2.0 license, 6 forks, 1 issue). Updated 13 days ago.
- k4ActsTracking** (Public): Acts integration in the key4hep framework. C++ (0 stars, 4 forks, 2 issues). Updated on Aug 4.
- k4EDM4hep2LcioConv** (Public): EDM4hep to LCIO Converter. C++ (0 stars, 2 forks, 0 issues). Updated on Aug 3.
- k4MarlinWrapper** (Public): GaudifyMarlinProcessors. Python (2 stars, Apache-2.0 license, 7 forks, 2 issues). Updated on Jul 8.
- k4Clue** (Public): Python (2 stars, 4 forks, 0 issues). Updated on Jul 1.
- k4-project-template** (Public template): Template for Gaudi-based Key4HEP projects. C++ (0 stars, 7 forks, 3 issues). Updated on Apr 25.

# Adoption status

- ✓ FCCSW adapted EDM4hep (switched from FCC-edm)
- ✓ CEPCSW using EDM4hep and switched from Marlin (iLCSoft) to Gaudi
- ✓ CLIC and ILD reconstruction can be run in Gaudi
  - Part of AIDAInnova WP12
- Introducing EDM4hep as output/input format to several tools ongoing
- Integration of more packages/libraries
  - ACTS tracking toolkit
- Ongoing maintenance and modernization of existing components





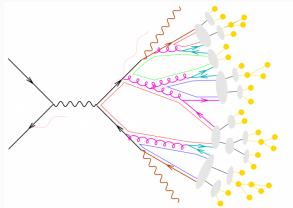
# Generators

---

# Generators in Key4hep

See [A. Siodmok's talk](#) for physics

- Generators are “just” software packages
- For inclusion in Key4hep a **spack recipe** is necessary
  - Building and installing becomes (almost) trivial
- Initial list from *LCG stacks* (mainly LHC focussed)
- Many  $e^+e^-$  additions since then
  - Including wrappers for better user experience



# Generators currently available via spack and Key4hep

- Generators

<code>babayaga</code> <sup>†</sup>	<code>baurmc</code> <sup>†</sup>	<code>bhlumi</code> <sup>†</sup>	<code>crmc</code> <sup>†</sup>	<code>evtgen</code>	<code>genie</code> <sup>†</sup>
<code>gosam</code> <sup>†</sup>	<code>guinea-pig</code> <sup>†</sup>	<code>herwig3</code>	<code>herwigpp</code> <sup>†</sup>	<code>kkmcee</code> <sup>*</sup>	<code>madgraph5amc</code>
<code>photos</code>	<code>pythia6</code> <sup>†</sup>	<code>pythia8</code>	<code>sherpa</code>	<code>starlight</code> <sup>†</sup>	<code>superchic</code> <sup>†</sup>
<code>tauola</code> <sup>†</sup>	<code>vbfnlo</code>	<code>whizard</code>			

- “Generator tools”

<code>agile</code> <sup>†</sup>	<code>alpgen</code> <sup>†</sup>	<code>ampt</code> <sup>†</sup>	<code>apfel</code> <sup>†</sup>	<code>ccs-qcd</code> <sup>†</sup>	<code>chaplin</code> <sup>†</sup>
<code>collier</code> <sup>†</sup>	<code>cuba</code> <sup>†</sup>	<code>dire</code> <sup>†</sup>	<code>feynhiggs</code> <sup>†</sup>	<code>form</code> <sup>†</sup>	<code>hepmc</code>
<code>hepmc3</code>	<code>heppdt</code>	<code>hoppet</code> <sup>†</sup>	<code>hztool</code> <sup>†</sup>	<code>lhpdf</code>	<code>lhpdfsets</code> <sup>†</sup>
<code>looptools</code>	<code>openloops</code>	<code>professor</code> <sup>†</sup>	<code>prophecy4f</code> <sup>†</sup>	<code>qd</code> <sup>†</sup>	<code>qgraf</code> <sup>†</sup>
<code>recola</code> <sup>†</sup>	<code>rivet</code>	<code>syscalc</code> <sup>†</sup>	<code>thepeg</code>	<code>unigen</code> <sup>†</sup>	<code>yoda</code>

- Currently the **latest version** of each package is installed in Key4hep stack

---

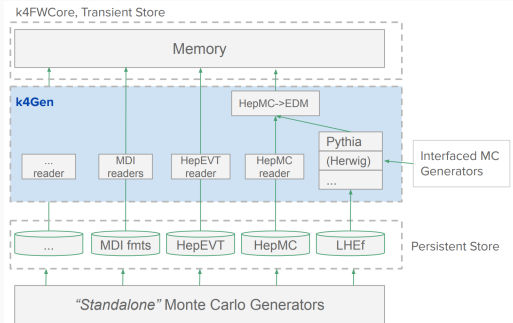
Installed with current Key4hep stack

\* Available from `key4hep-spack` repository

<sup>†</sup> Single version only

# Generator interoperability

- Majority of generators come as standalone executables
- Some have callable interfaces
  - Pythia, EvtGen, Herwig, ...
- Interoperability requires **common, well defined, data formats** or interfaces
  - Fully hadronized outputs in **HEPMC3**, **EDM4hep** for simulation
  - APIs can also be accommodated
- *k4Gen* offers several readers and tools to work on MC events
  - Particle gun, particle filters, vertex smearing, ...



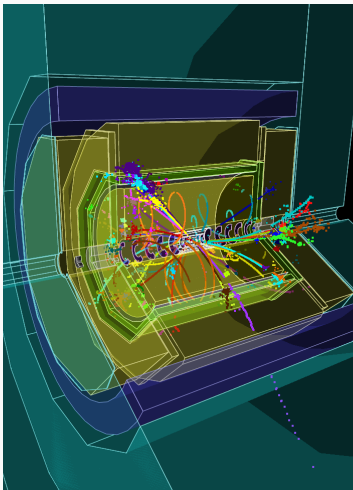
[G.Ganis](#)@ECFA generators workshop, Nov 2021

# Simulation

---





# Simulation

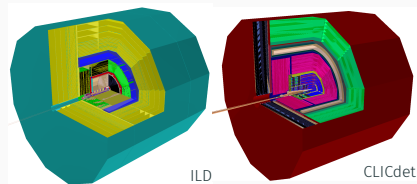
See [A. Sailer's talk](#) for the details



- Propagation of particles or decay products through detector
- Varying degrees of detail
  - **Full simulation** based on **Geant4**
  - Fast simulation
  - **Parametrized simulation (& reco)** via **Delphes**
- Simulation **requires a detector description**
  - Detailed studies need detailed simulations need detailed detector models

# Current approaches in Key4hep

- Full simulation based on **DD4hep**
- Many models already available
  -  [ilcSoft/lcgeo](https://github.com/ilcsoft/lcgeo)
  -  [HEP-FCC/FCCDetectors](https://github.com/hep-fcc/fccdetectors)
  -  [cepc/CEPCSW](https://github.com/cepc/cepcsw)
- Produce EDM4hep via standalone **ddsims**
- Framework integration
  - Consolidation of **k4SimGeant4** and **Gaussino**
-  [key4hep/k4SimDelphes](https://github.com/key4hep/k4SimDelphes) offers Delphes with EDM4hep output



# Reconstruction & Analysis

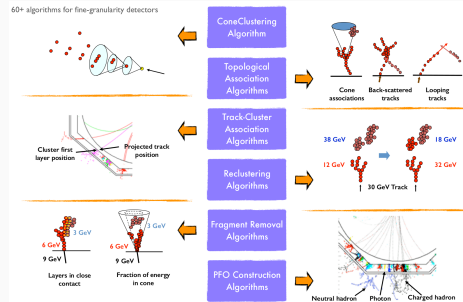
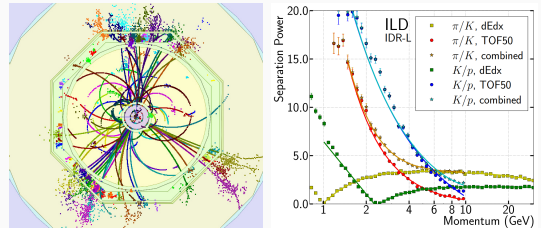
---



# Reconstruction & Analysis

- Everything after simulation
- Digitization, Overlay
- Reconstruction
  - Tracking, Clustering, ...
- “High level” reconstruction
  - Particle Flow (*Pandora*)
  - Particle Identification using TOF, dE/dx
  - Flavor tagging, etc.
- Physics analysis

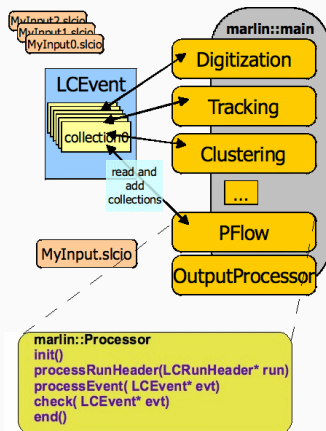
See [E. Brondolin's](#) and [J. Smiesko's](#) talks for more



PandoraPFA

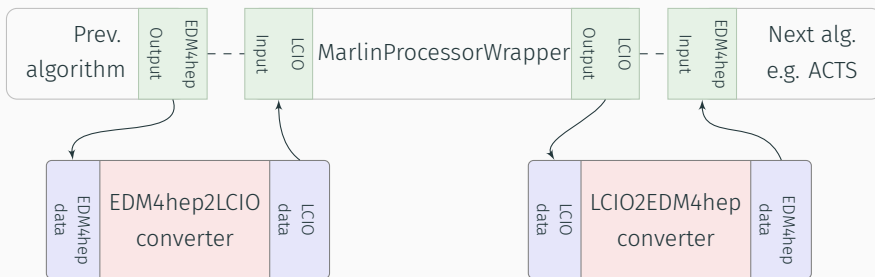
# iLCSoft - The ecosystem within the ecosystem

- Spiritual predecessor of Key4hep
- Offers a complete framework (*Marlin*)
  - Digitization, background overlay
  - Reconstruction
  - High Level analysis
- Thoroughly tested by LC projects (and **CLD**)
- **Ready to be used now**
- Working horse for ILD
  - Will be maintained for foreseeable future
- Modernization via evolution instead of revolution



# k4MarlinWrapper

- Wraps **Marlin processor** in a Gaudi algorithm and allows to **run them unchanged**
- Automatic, on-the-fly conversion between LCIO and EDM4hep
- **Allows to “mix and match” existing reconstruction algorithms with new developments**

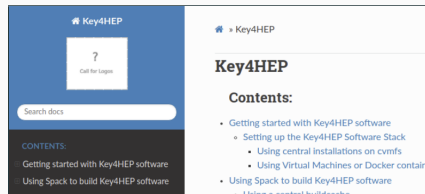


# Key4hep resources

- (Rolling) latest release of the complete Key4hep software stack

```
source /cvmfs/sw.hsf.org/key4hep/setup.sh
source /cvmfs/ilc.desy.de/key4hep/setup.sh
```

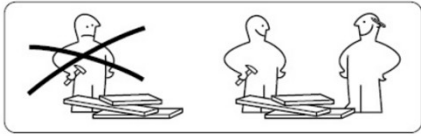
- Built via *spack* package manager for CentOS7
- **Release early and release often**
  - Solicit feedback as early as possible
- Documentation available at [key4hep.web.cern.ch](https://key4hep.web.cern.ch)
- Active weekly meetings (~ 10 – 15 attendees)
  - <https://indico.cern.ch/category/11461/>
- Feedback and contributions are greatly appreciated



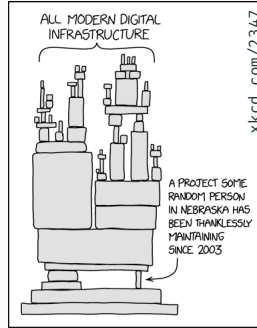
# Summary


- Key4hep provides a common software stack for **all future collider projects**
- Very successful in **bringing together communities** and **focusing on common approaches**
  - Common **EDM4hep** format with increasing maturity and adoption
  - **DD4hep** for detector description
  - Shared tools for building, developing and deploying software stack
- **Key4hep is ready to be used for future collider studies now**
- **Still a lot of room for your contributions**
  - Now is the ideal time to get on board

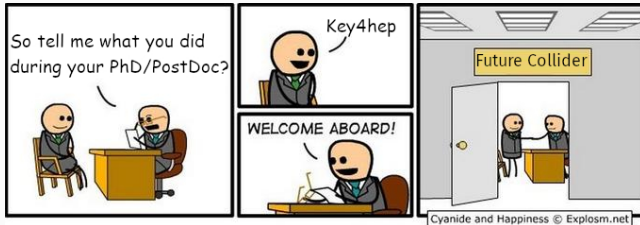
# A few convincing arguments



Collaboration is “The Right Thing”™

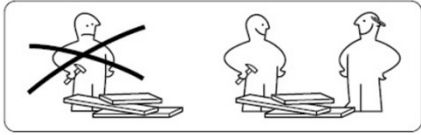


- Managing large SW stacks
- Research Software Engineering
  - Continuous Integration (CI)
  - Containerization
  - ML/AI (and running it in production SW)
-  \*more buzz words

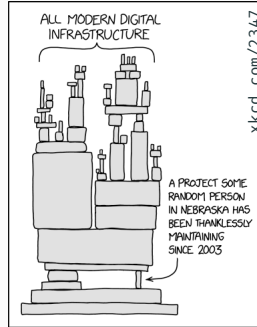



- More work than people
  - Create your own area of work
- A lot of visibility
- Simply a cool project

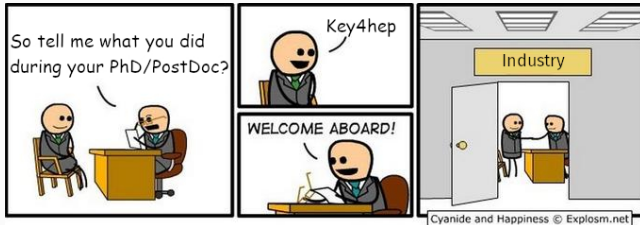
# A few convincing arguments



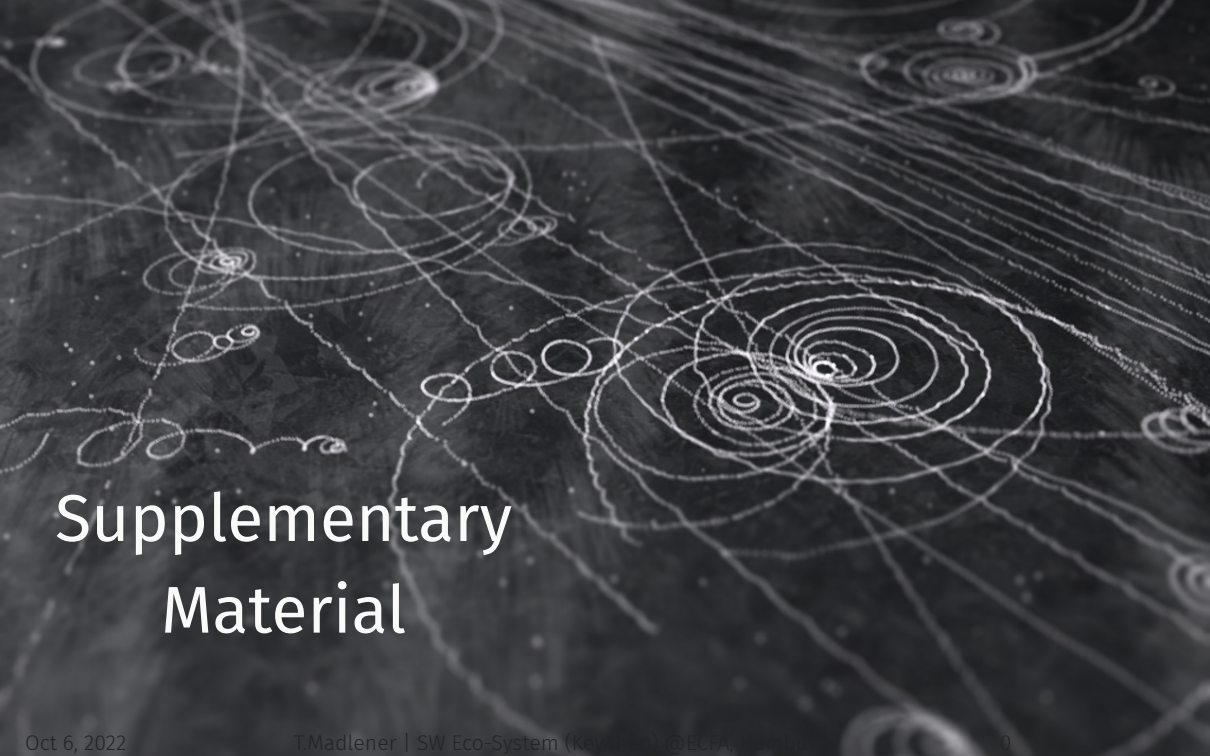
Collaboration is “The Right Thing”™



- Managing large SW stacks
- Research Software Engineering
  - Continuous Integration (CI)
  - Containerization
  - ML/AI (and running it in production SW)
-  \*more buzz words



- More work than people
  - Create your own area of work
- A lot of visibility
- Simply a cool project



# Supplementary Material



# Pointers to software (re)sources

- Key4hep

[key4hep.github.io/key4hep-doc](https://key4hep.github.io/key4hep-doc)

 [key4hep](https://github.com/key4hep) - github organisation

- EDM4hep

 [key4hep/EDM4hep](https://github.com/key4hep/EDM4hep)

[cern.ch/edm4hep](https://cern.ch/edm4hep)

- DD4hep

 [AIDASoft/DD4hep](https://github.com/AIDASoft/DD4hep)

[dd4hep.web.cern.ch](https://dd4hep.web.cern.ch)

- iLCSoft

 [iLCSoft](https://github.com/iLCSoft) - github organisation

[ilcsoft.desy.de](https://ilcsoft.desy.de)










- FCCSW

 [HEP-FCC](https://github.com/HEP-FCC) - github organisation



[xkcd.com/138](https://xkcd.com/138)

# Key4hep packages

- **k4FWCore**  [key4hep/k4FWCore](https://github.com/key4hep/k4FWCore)
  - Core Key4hep framework providing core functionality, e.g.
    - Data Service for EDM4hep inputs
    - Overlay for backgrounds
- **k4SimDelphes** for Delphes fast simulation  [key4hep/k4SimDelphes](https://github.com/key4hep/k4SimDelphes)
- **k4MarlinWrapper** Marlin proc. wrapper  [key4hep/k4MarlinWrapper](https://github.com/key4hep/k4MarlinWrapper)
- Many packages migrated from FCCSW to Key4hep
  - **k4SimGeant4** for Geant4 simulation integration  [HEP-FCC/k4SimGeant4](https://github.com/HEP-FCC/k4SimGeant4)
  - **k4Gen** for generic generator interface  [HEP-FCC/k4Gen](https://github.com/HEP-FCC/k4Gen)
  - ...
- Ongoing work to integrate more components
  - ACTS tracking framework  [acts-project/acts](https://github.com/acts-project/acts) |  [key4hep/k4ActsTracking](https://github.com/key4hep/k4ActsTracking)
  - CLUE fast clustering algorithms  [.cern.ch/kalos/CLUE](https://cern.ch/kalos/CLUE) |  [key4hep/k4CLUE](https://github.com/key4hep/k4CLUE)

# Ongoing work (selection)

## ACTS integration

- ACTS can now digest DD4hep detectors (with annotations)
- Minimal EDM4hep I/O support
  - More general solution under discussion
- Major effort with significant personpower requirements

## Gaudi modernization

- Switch towards more modern Gaudi approach (*Gaudi Functional*)
  - “Thread safe by default”
- Missing documentation is a major hurdle

## “Framework independent” algorithms

- EIC chose Jana2 over Gaudi
- Can “the hard part” still be shared?

# Spack for Key4hep

- [Spack](#) is a package manager
  - Independent of operating system
  - Builds all packages from source
- Originally developed by the HPC community
  - Emphasis on dealing with **multiple configurations** of the same package
- Basic building block is a formalized build procedure → **spack recipe**
  - Build instructions, dependencies, versions and location of source code
  - ~ 6650 packages currently available from spack
  - Key4hep maintains repository with additional packages
- The whole Key4hep software stack can be built from scratch using spack  

```
spack install key4hep-stack
```



# Spack recipe

```
class Evtgen(CMakePackage):
    """EvtGen is a Monte Carlo event generator that simulates
    the decays of heavy flavour particles, primarily B and D mesons."""

    homepage = "https://evtgen.hepforge.org/"
    url = "https://evtgen.hepforge.org/downloads?f=EvtGen-02.00.00.tar.gz"

    tags = ["hep"]

    maintainers = ["vvolkl"]

    version("02.00.00", sha256="02372308e1261b8369d10538a3aa65fe60728ab343fcb64b224dac7313deb719")
    # switched to cmake in 02.00.00
    version(
        "01.07.00",
        sha256="2648f1e2be5f11568d589d2079f22f589c283a2960390bbdb8d9d7f71bc9c014",
        deprecated=True,
    )

    variant("pythia8", default=True, description="Build with pythia8")
    variant("tauola", default=False, description="Build with tauola")
    variant("photos", default=False, description="Build with photos")
    variant("hepmc3", default=False, description="Link with hepmc3 (instead of hepmc)")

    patch("g2c.patch", when="@01.07.00")
    patch("evtgen-2.0.0.patch", when="@02.00.00 ^pythia8@8.304:")

    depends_on("hepmc", when="~hepmc3")
    depends_on("hepmc3", when="+hepmc3")
    depends_on("pythia8", when="+pythia8")
```

Build system

Where to find source code

Available versions

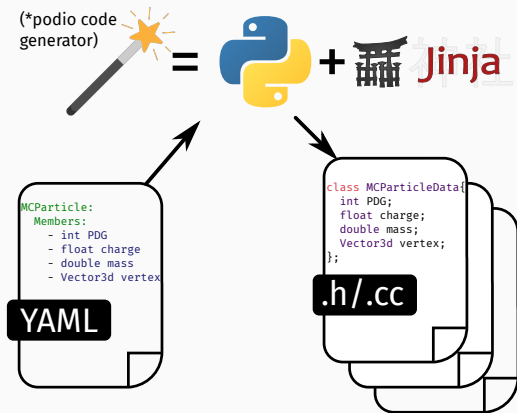
Variants / build options

On-the-fly patches

Dependencies

# podio as generator for EDM4hep

- Traditionally HEP c++ EDMs are heavily Object Oriented
- Use **podio** to generate thread safe code starting from a high level description
- Provide an easy to use interface to the users



 [AIDASoft/podio](https://github.com/AIDASoft/podio)

# podio supports different I/O backends

- Default **ROOT** backend
  - POD buffers are stored as branches in a **TTree**
  - Files can be interpreted **without EDM library(!)**
  - Can be used in **RDataFrame** or with **uproot**
- Alternative **SIO** backend
  - Persistency library used in **LCIO**
  - Complete events are stored as binary records
- Adding more I/O backends is possible

