Parameterized Simulation in the FCC framework

B. Hegner, CERN

for the FCC Experiment Software Team

29.4.2015

Driving Considerations

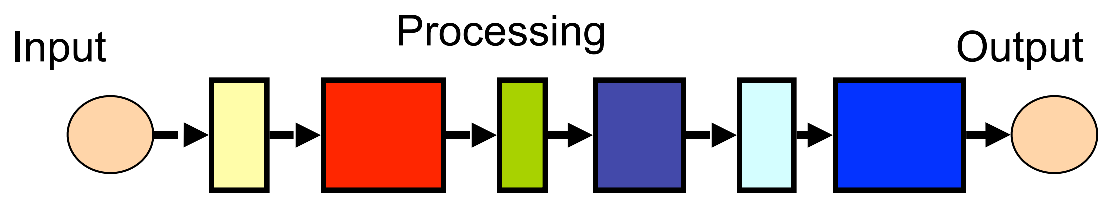
- Provide robust software to allow physics studies for CDR in 2018
- Support all **FCC-ee, -eh, and -hh** communities at the same time
  - Requires flexibility for Geometry and Simulation
- Start pragmatically
- As studies progress move to more sophisticated solutions
  - Allow components to be replaced later on
- FCC software effort relies on effort of other people
  - There is a give and take
  - Aim for, but don't blindly force, synergy with other communities

The Present

- Adapt existing solutions from LHC
  - Gaudi as underlying framework
  - ROOT for I/O
  - Geant4 for simulation
  - C++ **and** Python for user analysis
- Adapt software developments from ILC/CLIC
  - DD4Hep for detector description
- Invest in **better fast vs. full sim integration**
  - Geant4 fastsim, Atlfast (ATLAS)
- Invest in **proper future-proof data model**
  - The LHC experiments' ones are over-engineered
  - The ILC/CLIC model (LCIO) was designed before power and memory wall

Gaudi Event Processing Framework

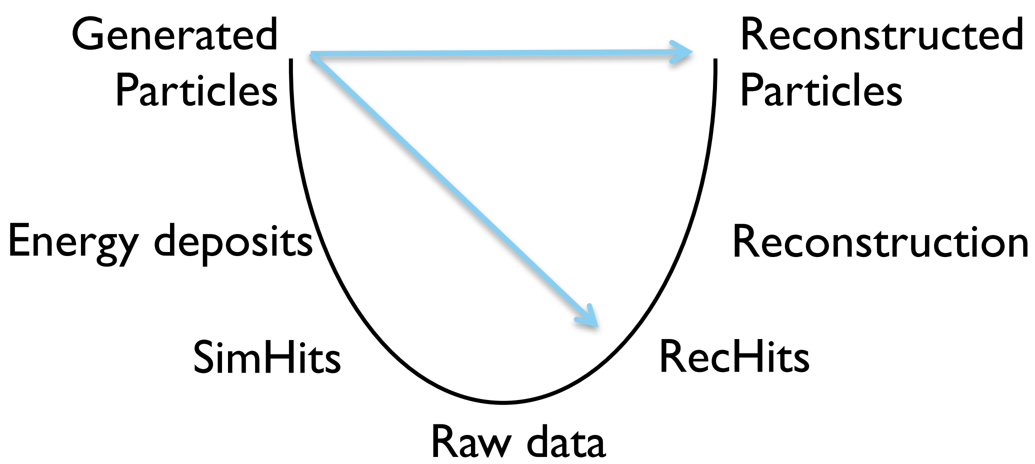
- Gaudi is an event-independent data processing framework
  - Used by LHCb, ATLAS, and a few smaller experiments
- Based on the concept of a software bus
- Work is split up in interdependent "algorithms"



- Parallelization effort with "GaudiHive" to take advantage of ever increasing hardware parallelization

FCC Simulation

- FCC Software needs to support the studies of multiple detectors
- At different stages different level of detail required
  - Smearing vs. fast sim vs. full sim



Different levels of detail =
Doing short cuts in the full workflow

FCC Simulation

- FCC Software needs to support the studies of multiple detectors
- At different stages different level of detail required
  - Smearing vs. fast sim vs. full sim
- FCC choices are
  - Delphes (*) and HepSim (**)
  - Fast simulation in Python
  - Integrated fast/full simulation with Geant4
- Should all be accessible from within the same framework
- Analysis code should be as similar as possible for these use cases.

(*) http://delphes.hepforge.org
(**) http://atlaswww.hep.anl.gov/hepsim/

FCCSW and Delphes

Delphes is a standalone package that just works out-of-the-box

- Simple and flexible

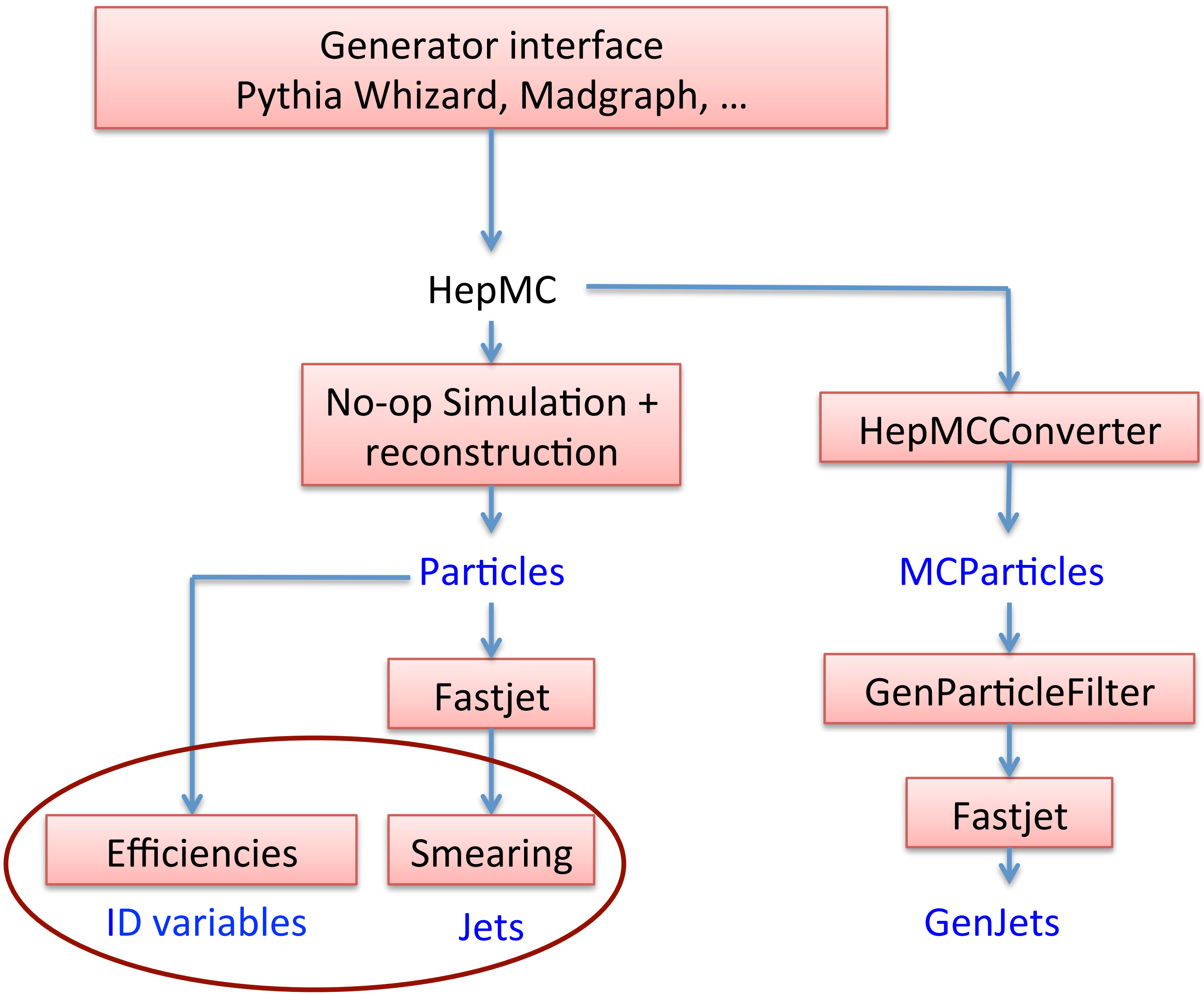
Not foreseen to be integrated into other frameworks and software stacks

- Work on build procedures and interfacing completed

Delphes uses a very specific data model - ExROOTAnalysis

- ExROOTAnalysis was used in CMS as stop-gap solution after problems in computing challenge in '07.
- Not completed step is mapping this data model into the FCC data model
  - FCCSW efforts limited by manpower, not by technology

MC level analysis in FCCSW



Your chance to contribute w/ smart ideas and code :-)

Detector Description

Detector Description in LHC experiments is a not-well organized environment

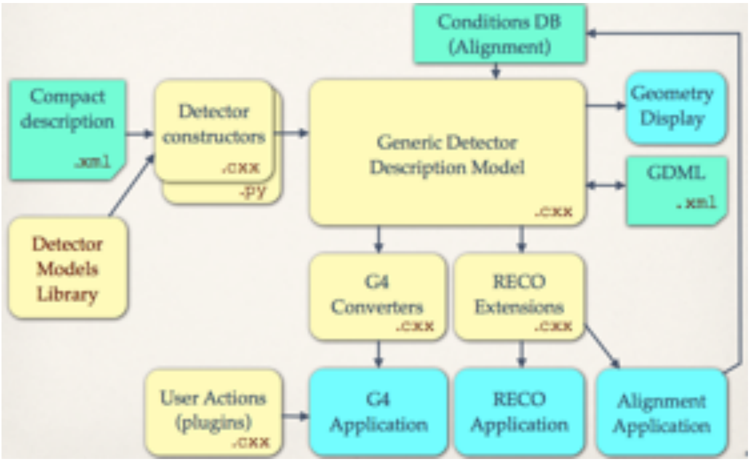
- Detectors modeled long ago and expertise largely gone
- Work on upgrade reveals weaknesses
- Heterogeneous setups even within experiments

ILC/CLIC efforts triggered the project DD4hep (*)

- Covering simulation, display, alignment in a consistent way

FCC joined these efforts of DD4hep

- Good support by developers!
- Working on first test-detector



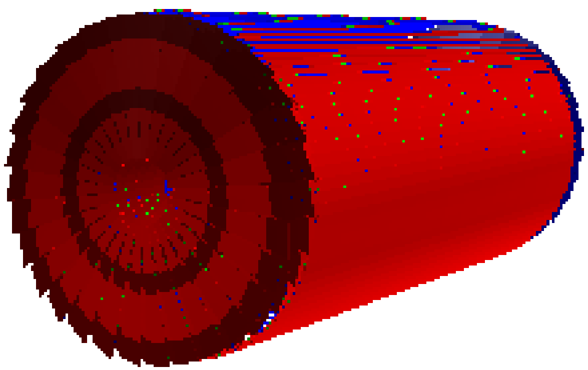
Existing Detector Descriptions

In DD4hep there currently exist a few prototypes FCC-hh could start with

- ILD and CLIC_SiD
- LHeC / FCC-eh detector
- FCC silicon tracker example

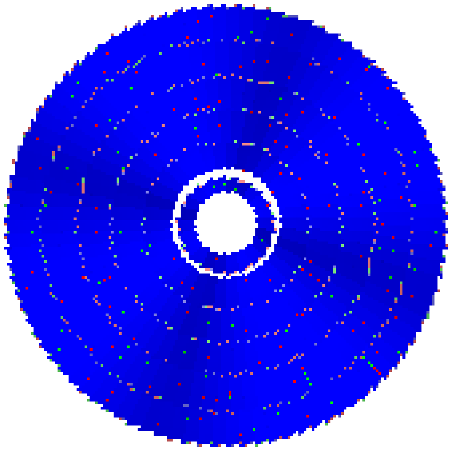
Important for FCC-hh would be to define detector hierarchy for people to get started building up sub-detector geometries

- Which of the concepts in Werner's presentation do we want to tackle first?



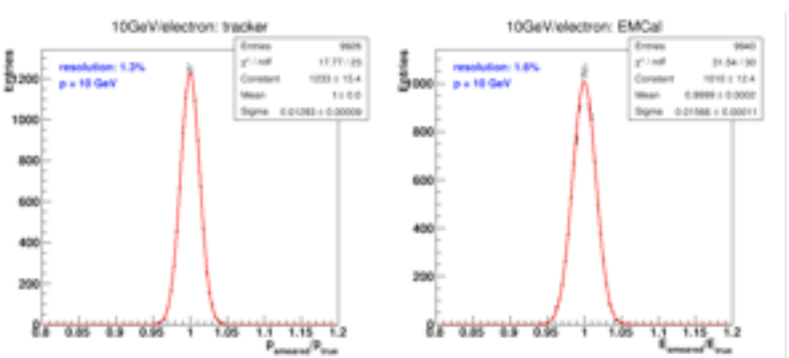
Fast Simulation

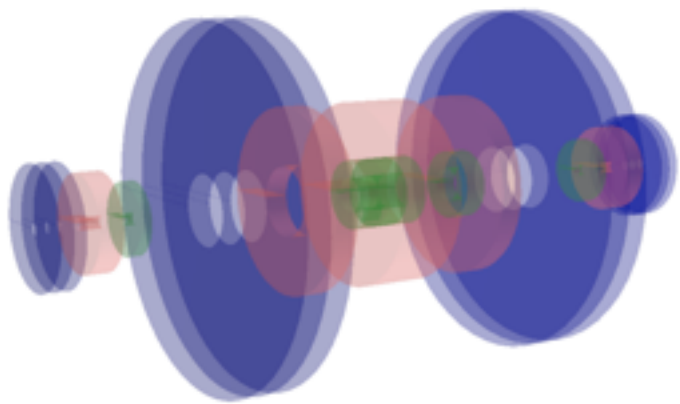
- Goal is to have a **combined fast and full simulation**
  - Decide at the config level where to do what
- (Semi-) automatic extraction of fast simulation parameters from full simulation
  - To be able to do fast-sim for any detector design
- Though not re-inventing the wheel, we are heavily re-designing it



Simulation Status

- First development phase was focussed on producing a demonstrator
  - Using expertise from ATLAS and Geant4 developers
  - Chosen approach worked out nicely
  - Results now being integrated into Geant4 and Gaudi

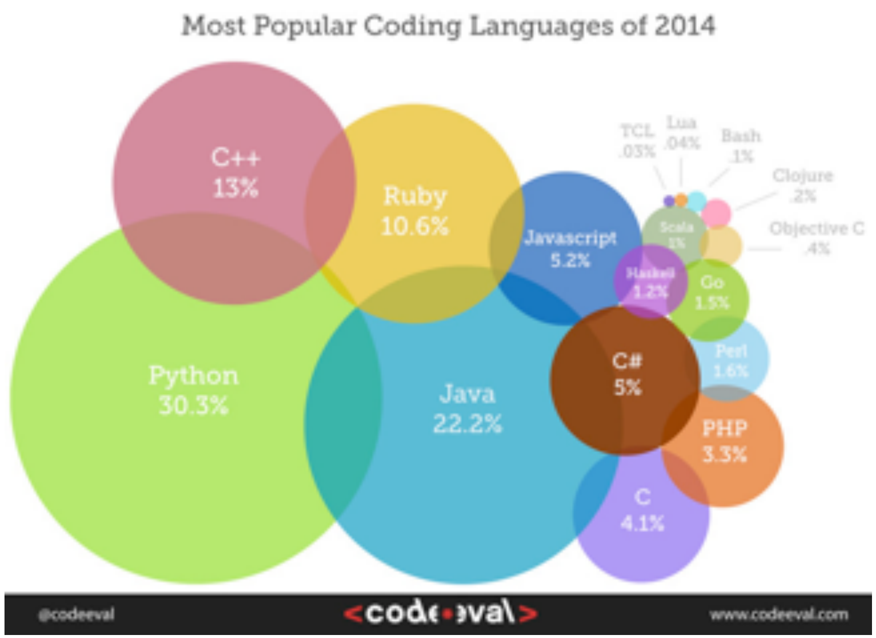




nota bene: main aim is faster simulation after first detector description is there

- Analysis should be easy and powerful
- Lesson from LHC experiments and ILC/CLIC
  - If setup is too complex, physicists stop using common software and create their own mini-frameworks
- Physicists will join from different experiments and we need to make the transition as easy as possible
- Need to allow multiple paradigms to do analysis
  - **C++ and Python**

Analysis in Python

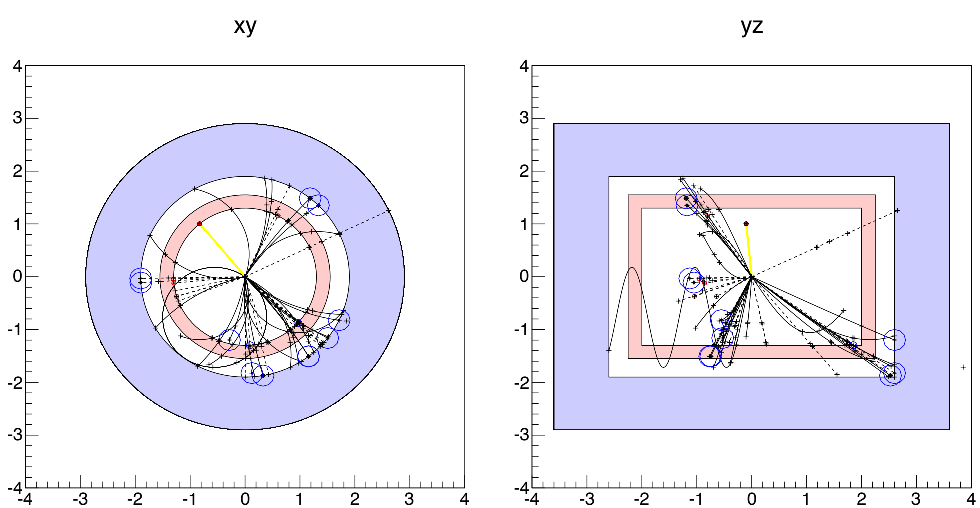


- Very large user base
- Super easy to learn
- Light & short code
- Good performance
  – usually wraps C or C++ modules
- « Batteries included »
  – massive and easy-to-use standard library
- Dynamic typing
  – good for multichannel analyses
  – code highly reusable
- Dynamic object modification
  – Can attach new attributes (or methods) to an existing object
- Productivity x 5-10 w/r C++
- A lot of fun!

- Supporting this with the **heppy** package originating from CMS

PAPAS - Fast Simulation with Python

- Fast simulation for physics studies and detector design studies
  - Written in Python it allows quick turn-around
- Uses the same data model as the C++ framework
  - Lowers the bar when moving to production code



More details tomorrow:

https://indico.cern.ch/event/390497/

Where are we now?

- Established common FCC experiment software project
- Multiple approaches to simulation
- First phase of pick & chose is finished
- Integrated fast/full sim design validated
- C++ and Python based analysis environment provided

- All being presented here is being discussed / designed in the FCC weekly software meeting:

  https://indico.cern.ch/category/5666/

Volunteers needed!

• Framework – Core event data model, Gaudi integration, Software stack	Bernet, Hegner
• Generators – Integration	Pilicer, **People needed**
• Simulation infrastructure – Geant4 (fast & full)	Carminati, Dell'aqua, Hrdinka, Salzburger, Zaborowska
– Delphes integration	De Gruttola, Hegner
• Reconstruction	**People needed**
• Analysis tools – python & C++ framework	Bernet
• Validation – testing and performance	Hegner , **People needed**
• Computing – sample production and management	**People needed**

Many thanks to our hard working team!