A software framework for FCC studies: status and plans

Javier Cervantes[1], *Jana* Faltova[2], *Gerardo* Ganis[1,*], *Clement* Helsens[1,**], *Julia* Hrdinka[1,3], *Coralie* Neubüser[1,4], *Lorenzo* Pezzotti[5], *Michele* Selvaggi[1,], *Niloufar A* Tehrani[1], *Valentin* Volkl[1,***], and *Anna* Zaborowska[1]

[1]CERN, Geneva 23, 1211 Geneva, Switzerland
[2]Charles University, Praha, Czechia
[3]Technische Universität Wien, Karlsplatz 13, 1040 Wien, Austria
[4]INFN, Trento, Italy
[5]University and INFN, Pavia, Italy

Abstract. The Future Circular Collider (FCC) is designed to provide unprecedented luminosity and centre-of-mass energies. The physics reach and potential of the different FCC options e^+e^-, pp, ep, has been studied and published in dedicated Conceptual Design Reports (CDRs) at the end of 2018. Conceptual detector designs have been developed for such studies and tested with a mixture of fast and full simulations. The investigations for all options have been conducted using a common software framework called FCCSW. In this paper, after summarising the improvements implemented in FCCSW to achieve the results included in the CDRs, we will present the current development plans to support the continuation of the physics potential and detector concept optimisation studies in view of future strategic decisions, in particular for the electron-positron machine.

1 Introduction

The Future Circular Collider (FCC) is a project designed to ultimately provide pp collisions at the largest centre-of-mass energy foreseeable, which is currently of the order of 100 TeV. Following a scenario similar to the LEP/LHC machines, the full FCC integrated program [1] features in sequence a high-luminosity e^+e^- electroweak, flavour, Higgs, and top factory, followed by a $\geq$ 100 TeV pp collider. The two FCC phases are commonly referred to as FCC-ee and FCC-hh, respectively.

A Conceptual Design Report (CDR) for the FCC project has been prepared end of 2018, and submitted as input to the 2019 Update of the European Strategy for Particle Physics. In view of the CDR, a dedicated software framework called FCCSW has been developed and used to study the physics potential of the proposed collider. For the preparation of the CDR, a total of about 250 TB of data have been simulated.

This paper reports on the status and evolution plans of FCCSW. It is organised as follows. In the next section we describe the driving considerations behind FCCSW and its core

[*]e-mail: gerardo.ganis@cern.ch
[**]e-mail: clement.helsens@cern.ch
[***]e-mail: valentin.volkl@cern.ch

components, including the event data model and the geometry description. In the following section we describe the main components of the computing workflow, including Monte Carlo generation, simulation, reconstruction and analysis. The current software infrastructure which served the CDR preparation will be presented in Sec. 4. The future developments and the relation with on-going across HEP common projects are discussed in Sec. 5. Finally, the concluding remarks will be presented in Sec. 6.

2 The FCC Core Software

FCCSW is a result of a process started in 2014 just after the FCC project kick-off. From the beginning the aim was to have one software stack to support all the collider phases (ee, hh, eh) and detector concepts, which brought the challenge to support a broad range of event complexity. The framework had to support physics and detector studies with parametrised, fast, and full simulation, allowing also a mixture of the three. It had to be modular enough to allow for evolution, allowing component parts to be improved separately. Finally it had to allow multi-paradigms for analysis, with C++ and Python at the same level. The strategy to meet these challenging requirements has been to adopt existing solutions from LHC and to look at ongoing common projects, such as those developed under the AIDA EU project [2], in particular, as it will be mentioned below, in streamlining the event data model.

The result of this process is schematically shown in Figure 1.

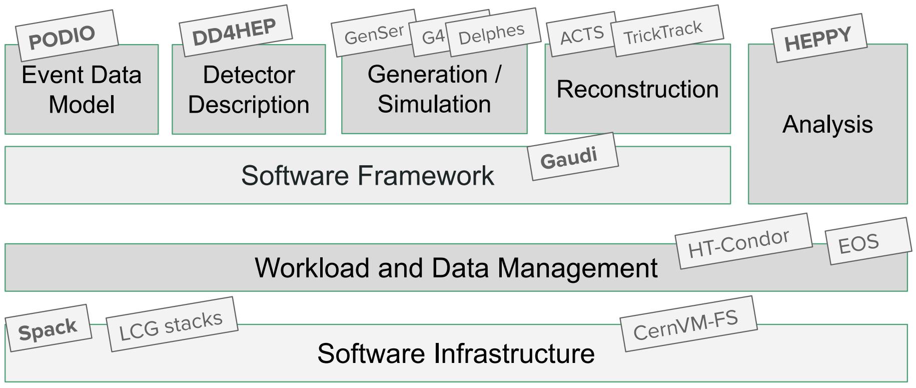

Figure 1. The components taken from existing projects included in FCCSW. See the text for a description of their role.

2.1 The Event Data Model and `PODIO`

The first FCC studies were focusing mostly on the FCC-hh phase. The event data models of the closest LHC detectors, ATLAS and CMS, were considered overly complex, and potentially unfavourable for I/O performance, in particular in the long term.

FCC decided to adopt a different approach following a then starting AIDA 2020 project, called PODIO [3]. PODIO is an Event Data Model toolkit allowing the automatic creation of *Plain Old Data* (POD) data structures starting from a high-level description of the required types. Automation allows to generate consistent and homogeneous implementations, minimising mistakes. The separation in high-level and low-level layers provides support different backends; the persistent layer as POD allows to keep the memory model simple, enabling fast I/O and efficient vectorisation.

FCC is the first project to use PODIO and the experience has been overall quite successful. However, the tool still lacks some features that are essential for a running experiment, such as schema evolution, optimisations of the memory layout and of the I/O performance. In view of the adoption of PODIO in the context of the ongoing common software stack initiative (see Sect. 5), the development of these features in PODIO has been proposed as a follow-up project in the context of the new edition of AIDA.

2.2 Detector Description: `DD4hep`

All detector geometries are implemented using the detector description toolkit DD4hep [4, 5]. This allows for runtime configuration of detector dimensions, materials and parameters, read from "compact files" in xml format. Standalone compact files are provided for each detector subsystem, and are combined in a "master" compact file to provide a full description of the detector option. The extent to which parameters can be changed at runtime depends on the specific detector implementations, but the overall dimensions can be adapted for almost all models to allow them to fit a given geometrical context. This approach allows also the "plug-and-play" interchange of existing detector subsystems; for example, the same tracking system can be evaluated with different solutions for the calorimeter technology.

2.3 The underlying software framework: Gaudi

FCCSW is based on Gaudi [6], a framework developed originally for the LHCb experiment, but which has found more widespread use in other experiments including ATLAS. Recent efforts to adapt Gaudi for concurrent data processing [7] make it a good candidate for use with evolving computing hardware.

Figure 2 shows the way Gaudi is used in the FCC software. Sec. 3 gives a more detailed account of the components used

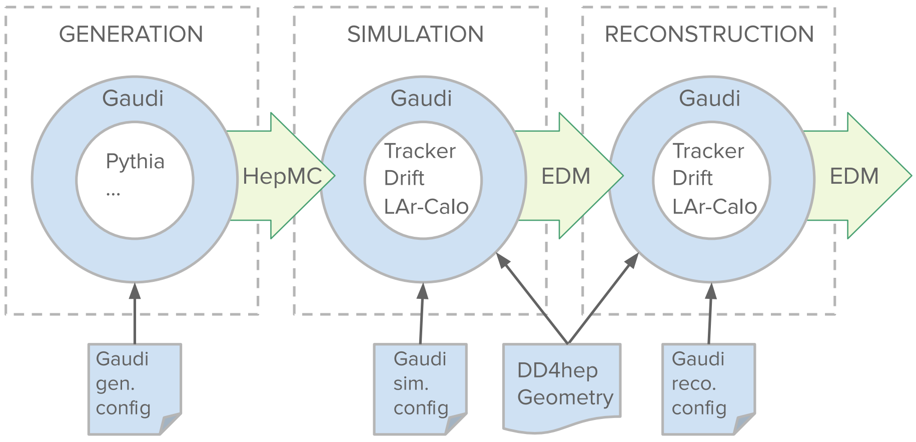

Figure 2. Schema of the organisation of Gaudi components in FCCSW, their steering and the data flow.

3 Computing workflow components

3.1 Monte Carlo generation and MDI

Monte Carlo Generators are provided by the GenSer project, and mostly used standalone, using common data formats to produce files readable by FCCSW. An exception is Pythia8

[8], the main tool to simulate hard scatter events and hadronisation for FCC-hh. Gaudi components for Pythia8, conversion to and from HepMC [9] and user-configurable single particle input to simulation have been added to FCCSW. All of these components can make use of additional tools to smear vertex positions with any given spatial and temporal distribution.

The different FCC options have different sources of background: FCC-hh will have extreme levels of pileup collisions ($< \mu >= 1000$), while the main concern for FCC-ee are beam backgrounds. FCCSW provides components that can handle and overlay such background data, eliminating the need to generate and simulate large events on the fly.

3.2 Simulation

In order to address the different needs within the design study, different simulation libraries are integrated as components in FCCSW. Delphes [10] is used for fast, parametrised simulations of the detector response. A Delphes card parametrising the response of FCC-hh is included with FCC software.

Geant4 is used for detailed studies of the detector response. Full detector simulation is computationally expensive, but the FCC software infrastructure allows mixing of full and fast simulation based on detector regions [11]. Gaudi components to specify user actions, physics lists and outputs are also available in FCCSW.

3.3 Reconstruction

Reconstruction software tends to be experiment-specific, and avoiding code duplication for the different FCC options poses some challenges. Nevertheless, efforts have been made to use and develop generic reconstruction libraries in FCC. "A Common Tracking Software" (ACTS) [12], developed within ATLAS, is one such library the encapsulates tracking code that used to be experiment-specific. Similarly, TrickTrack makes part of the CMS pixel track seeding code available in an experiment-independent library. For calorimeter reconstruction, a Sliding-Window and Topo-Clustering algorithm have been implemented, adapted from ATLAS approaches.

3.4 Analysis

The analysis and n-tuple production of simulated data was initially done using the Heppy framework [13], written in Python. While flexible, the performance of the Python implementation in processing the large volume of data required was problematic. With the RDataFrame classes, part of the ROOT library, [14] there is however a strong alternative concept that provides the performance of compiled C++ code with the flexibility of an interpreter. In order to use RDataFrame for FCC analyses, the analysers and I/O routines of Heppy were ported to C++.

4 Software infrastructure

A central experiment software provides many benefits to developers and user as it reduces the overhead of much necessary infrastructure. All FCC packages follow best practices established by the HEP Software Foundation (HSF), make use of modern CMake build configurations and C++ features where applicable. Releases and nightly builds are built with the package manager tool Spack and deployed in a dedicated CVMFS repository (`/cvmfs/fcc.cern.ch`). A dedicated website (`https://cern.ch/fccsw`) lists resources, technical documentation and tutorials for users.

5 Future developments and `Key4hep`

The publication of the CDR marks a new phase for the FCC design studies, for which the current organisation of software developments has provided a good foundation. Further studies leading up to a technical design report will require more detailed and comprehensive simulation and reconstruction workflows. In order to enhance collaborations and efficiently use R&D resources, a new common project called Key4HEP has been agreed upon with other future collider projects, including CLIC, ILC and CEPC. This aims to establish a common, ready-to-use ("turnkey") software stack to which all participating studies can contribute [15]. Figure 3 shows the proposed re-organisation of packages in experiment-specific, inter-experiment and completely generic parts.

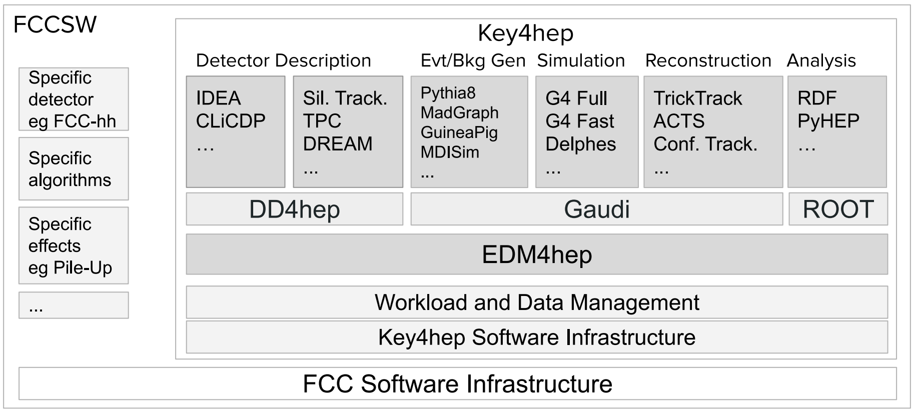
FCCSW
Specific detector eg FCC-hh
Specific algorithms
Specific effects eg Pile-Up
...
Key4hep
Detector Description
Evt/Bkg Gen
Simulation
Reconstruction
Analysis
IDEA CLiCDP …
Sil. Track. TPC DREAM ...
Pythia8 MadGraph GuineaPig MDISim ...
G4 Full G4 Fast Delphes ...
TrickTrack ACTS Conf. Track. ...
RDF PyHEP …
DD4hep
Gaudi
ROOT
EDM4hep
Workload and Data Management
Key4hep Software Infrastructure
FCC Software Infrastructure

Figure 3. Diagram showing the proposed use of the Key4HEP software stack in the FCC experiment.

6 Conclusions

FCC software has effectively supported the studies leading up the CDR, emphasising collaboration and re-use of existing packages. With the start of a new phase for FCC, more detailed studies, in particular for e^+e^- will be required. To this end, developers will closely follow and participate in new common activities under the Key4HEP organisation.

References

[1] M. Benedikt, A. Blondel, O. Brunner, M. Capeans Garrido, F. Cerutti, J. Gutleber, B. Goddard, P. Janot, J.M. Jimenez, M. Klein et al., Tech. Rep. CERN-ACC-2019-0007, CERN, Geneva (2019), `https://cds.cern.ch/record/2653673`

[2] T.A.. Collaboration, Tech. Rep. AIDA-2020-NOTE-2018-002, CERN, Geneva (2018), `https://cds.cern.ch/record/2628353`

[3] F. Gaede, B. Hegner, P. Mato, J. Phys. Conf. Ser. **898**, 072039 (2017)

[4] M. Frank, F. Gaede, C. Grefe, P. Mato, J. Phys. Conf. Ser. **513**, 022010 (2014)

[5] M. Petrič, M. Frank, F. Gaede, A. Sailer, EPJ Web Conf. **214**, 02037 (2019)

[6] G. Barrand et al., Comput. Phys. Commun. **140**, 45 (2001)

[7] M. Clemencic, B. Hegner, C. Leggett, J. Phys. Conf. Ser. **898**, 042044 (2017)

[8] T. Sjostrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C.O. Rasmussen, P.Z. Skands, Comput. Phys. Commun. **191**, 159 (2015), `1410.3012`

[9] M. Dobbs, J.B. Hansen, Comput. Phys. Commun. **134**, 41 (2001)

[10] M. Selvaggi, J. Phys. Conf. Ser. **523**, 012033 (2014)

[11] A. Zaborowska, J. Phys. : Conf. Ser. **898**, 042053. 8 p (2017)

[12] C. Gumpert, A. Salzburger, M. Kiehn, J. Hrdinka, N. Calace (ATLAS Collaboration), Tech. Rep. ATL-SOFT-PROC-2017-030. 4, CERN, Geneva (2017), `https://cds.cern.ch/record/2243297`

[13] C. Bernet, Tech. Rep. 1st DAWG Technology and Innovation Survey, CERN, Geneva (2019), `https://indico.cern.ch/event/789007/contributions/3317130`

[14] E. Guiraud, Tech. Rep. CHEP 2018, CERN, Geneva (2018), `https://indico.cern.ch/event/587955/contributions/2937534`

[15] A. Sailer, G. Ganis, P. Mato, G.A. Stewart, to appear in J. Phys. Conf. Ser. (2020)