




Review and outlook of accelerator-related codes and their interplay with the experiments software

Manuela Boscolo^{1,2}, Helmut Burkhardt², Gerardo Ganis^{2,a} , Clément Helsens²

¹ INFN, Frascati, Italy

² CERN, Geneva, Switzerland

Received: 16 June 2021 / Accepted: 20 November 2021

© The Author(s) 2021, corrected publication 2022

Abstract Powerful flexible computer codes are essential for the design and optimisation of accelerator and experiments. We briefly review what already exists and what is needed in terms of accelerator codes. For the FCC-ee, it will be important to include the effects of beamstrahlung and beam–beam interaction as well as machine imperfections and sources of beam-induced backgrounds relevant for the experiments and consider the possibility of beam polarisation. The experiment software Key4hep, which aims to provide a common software stack for future experiments, is described, and the possibility of extending this concept to machine codes is discussed. We analyse how to interface and connect the accelerator and experiment codes in an efficient and flexible way for optimisation of the FCC-ee interaction region design and discuss the possibility of using shared data formats as an interface.

1 Introduction

The international Future Circular Collider (FCC) study aims to design p-p, e^+e^- and e-p colliders to be built in a new 100 km tunnel in the Geneva region. The e^+e^- collider (FCC-ee) has a centre of mass energy range between 91.2 and 365 GeV with instantaneous luminosities as high as $2.3 \times 10^{36} \text{cm}^{-2} \text{s}^{-1}$ and $1.55 \times 10^{34} \text{cm}^{-2} \text{s}^{-1}$, respectively [1]. The design of the interaction region is crucial to reach such unprecedented energies and luminosities.

The main characteristics of the interaction region optics design is determined by the crab-waist scheme with a local chromatic correction system and a horizontal crossing angle of 30 mrad at the interaction point. A description of the main challenges of the interaction region and machine detector interface (MDI) design can be found in Ref. [2]. The baseline optics for the FCC-ee double-ring collider is described in Ref. [3]. The total synchrotron radiation power is limited by design at 100 MW for the two beams, and consequently, the stored current per beam varies from 1.4 A at Z to 5.4 mA at the $\bar{t}\bar{t}$ data taking stage. Following the LEP-2 experience where the highest local critical energy was 72 keV for photons emitted at 260 m from the IP [4], the FCC-ee optics design maintains critical energies from bending magnets below 100 keV starting at 100 m from the interaction point; the critical energy from the first bend after the interaction point is higher, at 691 keV for the $\bar{t}\bar{t}$ threshold. An asymmetric optics has been designed to meet these critical energy goals in the interaction region. Synchrotron radiation mask tips are placed in the horizontal plane just in front of the first final focus

^a e-mail: gerardo.ganis@cern.ch (corresponding author)

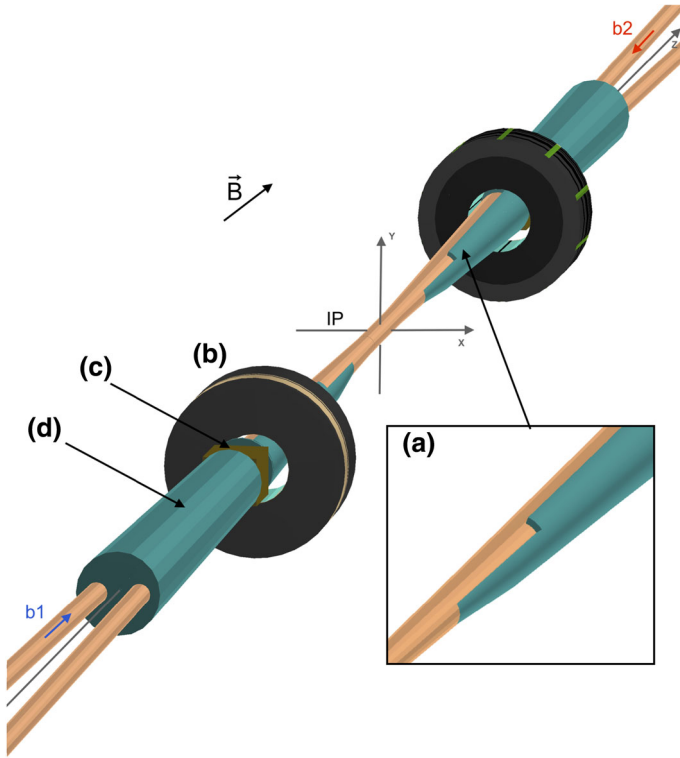


Fig. 1 Sketch of the FCC-ee interaction region. (a) Detail of the shielding; (b) luminometer; (c) HOM absorber; and (d) thick tungsten shielding [1]

quadrupole at 2.1 m from the interaction point. The free length between the interaction point and the first final focus quadrupole is 2.2 m, which is inside the detector. Figure 1 shows the GEANT4 model with the shielding and the luminometer that was used for background simulation studies [1].

The compactness of the MDI design, determined by the space available to host all the necessary components like the first final focus quadrupole and the anti-solenoids being placed inside the detector, poses interesting technical challenges. Part of the challenge is the development of modern flexible software tools for the beam optics model, including the beam-induced background scattering processes with an interface with the experiments.

This essay is organised as follows. We give an overview of the existing accelerator codes related to the interaction region and MDI design in Sect. 2; in Sect. 3, we describe the experiment software and the current strategy to interface the accelerator and experiment codes, and in Sect. 4, we discuss the geometrical description of the relevant elements. Finally, in Sect. 5 we summarise the status and the challenges ahead.

2 Review of the accelerator codes and MDI considerations

From the beginning of the studies for FCC, it was realised that the design of the interaction regions is particularly challenging and it is necessary that the requirements of the machine and experiments should be evaluated and optimised concurrently.

2.1 Lattice codes

The most popular code used for accelerator design at CERN and several other laboratories is the Methodical Accelerator Design program (MAD). The MAD program has been developed, maintained and upgraded for more than 3 decades. The main version used at LEP was MAD8 [5], written in FORTRAN V. Since then, it was largely rewritten as MAD- X using a combination of FORTRAN95 and C, and later also C++ for selected modules. The activity started in the new millennium [6], coinciding with the end of LEP operation and the shift of priorities motivated by developments for the LHC and its injectors, where synchrotron radiation only plays a minor role.

For the FCC-ee design, we also profit from the more recent experience and code developments for lepton colliders, by working with the Strategic Accelerator Design (SAD) [7] program used for SuperKEKB. SAD, like MAD- X, is an accelerator lattice design code, developed independently at KEK, and therefore, it provides a good opportunity for comparison and cross-checking. Both SAD and MAD- X read the machine description from formatted text files, using their own specific commands to specify magnet types, strengths and position. These input files are typically referred to as sequence files. A translator to convert the SAD lattices into MAD- X format exists.

The MAD- X lattice input text format allows the specification of aperture information like beam pipe shapes, sizes, and more recently also information on materials, provided as formatted comments [8]. Even for large machines like FCC with the order of 10000 magnets, the size of the sequence files is always manageable, often less than a megabyte. The basis for the MDI work is the lattice description in MAD- X format, typically starting from SAD translated to MAD- X format, with additional information on aperture and beam pipe material.

The output from MAD- X provides extra information per element like, beam position, beam size, TWISS parameters (beta functions, phase advance, etc.), 6×6 element transfer matrices and optionally $6 \times 6 \times 6$ second-order transfer maps. The MAD- X output format is known as TFS format [5], a human readable tabular format with extra general header information specifying parameters that apply to the whole machine like nominal beam energy and beam particle type. Even if we select all options and write numbers with 17 digits to avoid any loss in precision, the file sizes remain below 100 megabytes.

2.2 MDI considerations

The software used in MDI studies must be able to simulate in detail what happens in the interaction region. The beam pipe aperture should be sufficiently large to allow the beam particles to pass without any major losses for all modes of operation, including injection, and also be compatible with possible failure scenarios. Heating by HOM (higher-order modes losses, induced by the electromagnetic fields of the intense bunched beams) must be minimised. Even under optimal conditions, the intense particle beams stored in the FCC will always result in some level of particle losses and synchrotron radiation, and the latter is considered by the experiments as unwanted beam-induced backgrounds. The modelling of

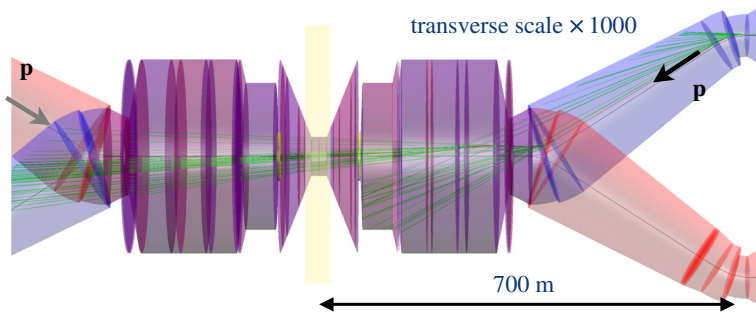


Fig. 2 Illustration of the FCC-hh interaction region using ROOT, and interfacing the MAD-X generated machine layout with MDISIM to GEANT4 to track the protons through the interaction region and generate synchrotron radiation photons [14, 15]

the beam-induced background effects has often been performed with custom Monte Carlo codes and relying on input and geometry data in specific formats which have to be written or modified by hand.

These days, computers are sufficiently powerful and have enough memory and storage capacity that it could be possible to build a supercode that combines all that is needed in a single program. Another possibility would be to make all codes available from a unified code library. This was already attempted in the nineties [9] for accelerator codes, but with rather limited adoption and lack of support. The choices made can also be influenced by sociological perspectives and personal preferences. Working with large programs which have grown historically can be intimidating and may appear less rewarding than creating well defined smaller programs and code pieces associated with the names of a few authors.

For FCC (ee and hh), we have made an effort in the development of MDISIM [10] to use and combine existing codes as much as possible using a light, flexible interface, minimising the need for hand coded geometries, and privileging open-source and well-supported codes and exchange formats. We use MDISIM to read the TFS files generated by MAD-X and automatically translate them into an exchange format, directly readable by GEANT4 [11] and ROOT [12]. The format used is GDML [13] for the geometry information, complemented by magnet strengths, initial beam positions and directions provided by automatically generated human readable text files. GEANT4 is used to track the particles through the interaction region with generation of secondary particles and simulation of interactions in materials. The ROOT Event Visualization Environment is used for the display. An example is shown in Fig. 2 [14].

Further information on the geometry and material outside the beam pipe like vacuum equipment, shielding, magnet material or the experiment detectors can then be added on the GDML level. Many commercial and open software codes are being developed that can work with GDML and interface with other flexible geometry descriptions including CAD formats [16–24].

2.3 Beam-induced backgrounds

To simulate and minimise the impact of all relevant processes that can result in the loss of beam particles or secondary particles generated by the beams in the experiment detectors is a complex task. Beam-gas, Touschek and thermal photon scattering as well as synchrotron radiation will always be present, even if beams are not colliding, and they require the whole ring to be studied. For FCC-ee, the minimisation of synchrotron radiation effects is of primary

importance and has strongly influenced the basic design and layout choices. The collisions of the beams in the interaction regions will generate additional losses and produce synchrotron radiation by deflection in the electromagnetic fields of the opposing beam (referred to as beamstrahlung).

The MDISIM code is capable of efficiently generating the accelerator and beam pipe geometry for shower simulations for the whole ring. The GEANT4 toolkit [11] has code for all major particle scattering processes including the generation of synchrotron radiation [25], as well as tracking in magnetic fields, and is well suited for detector simulations. GEANT4 can be considered as a candidate for a supercode that simulates both machine and experiment and has been used as the basis for the combined codes G4BEAMLIN and BDSIM [26,27].

For benchmarking purposes, we use GEANT4 directly to track a few particles over several turns in small machines and have been contributing to developments to improve the tracking precision in GEANT4. For large machines like FCC, this is not realistic at present: we would need to consider some 10^{11} particles per bunch, circulating many times in a 100 km ring, to be able to determine the effects of radiation and the loss of a tiny fraction of the circulating particles in the interaction regions. At present, we restrict the GEANT4-based simulations to roughly a kilometre around the interaction region and work with other, more dedicated codes to complete these studies when needed. To gain speed, it is easier to guarantee a high level of numerical precision in accelerator codes that track deviations from the design path rather than tracking absolute positions.

The synchrotron radiation is emitted in excellent approximation in beam direction and not deflected by magnet fields. With scattering and reflection processes included, we find that only the synchrotron radiation generated in a limited range (some hundred meters) around the interaction is relevant as a source of detector backgrounds, and that more general effects like non-Gaussian tail generation in collisions can be taken into account by a proper choice of the beam distribution. Details of the GEANT4-based synchrotron radiation background simulations are described in Ref. [28]. We also make comparisons with the more dedicated SYNC_BKG [29] and SYNRAD+ [30] codes. Non-Gaussian tails were observed at LEP [31]. They can be generated by scattering processes and be enhanced by nonlinearities in the beam–beam interaction or strong sextupoles in combination with machine imperfections. A popular code much used at LHC to study the effect of nonlinearities and imperfections in multiturn tracking is SIXTRACK [32]. It would have to be adapted for e^+e^- multiturn tracking, for example, to account for the synchrotron radiation damping. A new tracking tool named XTRACK, part of the Xsuite project [33] is being considered for collimation studies at FCC-ee.

The GUINEAPIG [34] code is used as generator of beamstrahlung and radiative Bhabha scattering in the interaction regions. We also use the BBBREM [35] code as generator for the simulation of radiative Bhabha scattering. This process is characterised by an energy loss of one of the colliding particles and very small scattering angles, such that the scattered particles remain within the beam pipe. The process has been integrated into SAD and was in particular used for simulations of FCC-ee at the Z energy, where the beam intensity and luminosity are highest.

Particle losses by beam-gas and thermal photon scattering in multiple turns around the ring are taken into account using SAD, MAD-X, or PTC for the transport element by element, performing aperture checks at element boundaries. The results [36] were compared with the more detailed GEANT4 simulations performed around the interaction regions [37], and they were found to be in good agreement. The vacuum pressure profile in the MDI area, as well as in the whole ring, can be used as input for the beam-gas scattering simulations rather than assuming a constant pressure profile inside the beam pipe. The local pressure profile

can be evaluated by means of the Monte Carlo code MOLFLOW+ [38]. MOLFLOW+ provides detailed 3D calculations of vacuum properties in the molecular flow regime, such as pressure profiles, effective pumping speeds and adsorption distributions which are of interest mainly for vacuum engineers. It also allows the simulation of gas propagation in CAD imported geometries and simulates pumpdown processes.

Touschek scattering is an intra-beam scattering particularly relevant for low energy storage rings and is the major beam lifetime limitation for lepton colliders like DAΦNE, SuperKEKB and all the modern low and ultra-low emittance light sources. For the high energy FCC-ee, a simpler, more dedicated Monte Carlo embedded in particle tracking as developed for DAΦNE [39] and used for SuperKEKB should be fully sufficient.

Beam polarisation will be important for the FCC-ee and in particular for the precise determination of the beam energy. It requires studies of the whole ring including realistic modelling of imperfections and optics corrections [40]. A good candidate for detailed simulations with polarisation is the SITROS code [41]. Another good candidate is BMAD [42] which is a flexible software toolkit for the simulation of charged particles and X-rays including the spin tune and polarisation.

3 Experiment software

The software used to study the physics potential of FCC goes generically under the name of FCCSW and comprises a set of tools covering the needs of an experiment: signal generation, detector description and simulation, event reconstruction and final analysis [43]. FCCSW is a result of a process started just after the FCC project kick-off in 2014. The design goal has been to support physics and detector studies with parameterised, fast and full simulation, also allowing a mixture of the three. It has to be modular enough to allow for evolution, allowing component parts to be improved separately. Finally, it has 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 solutions developed for LHC, such as the Gaudi framework [44] and to look at ongoing common projects; among the latter, it is worth mentioning those developed under the AIDA EU R&D effort [45]: PODIO [46], used to define the event data model, and DD4HEP [47], used for the geometrical description of all the elements relevant for the physics measurements, i.e. the sensitive and passive elements of the sub-detectors, supports, magnet and elements of the interaction region affecting the detector performance, such as the beam pipe and other elements which can scatter or produce particle debris in the detector.

The Gaudi framework implements an architecture in which data flow through a transient data store (in memory), where they can be modified by algorithms representing the various steps of the data processing chain, e.g. generation, simulation or reconstruction. Readers and converters are special algorithms that can inject data into the transient store for processing.

Because of its capability of coping effectively with the data processing needs of High Energy Physics (HEP) experiments and the challenges of HL-LHC, Gaudi has also been chosen as the main framework of the Key4hep common software project, around which FCCSW is going to evolve in the future.

3.1 MDI-induced backgrounds

The processes described in Sect. 2.3, in addition to influencing the beam lifetime and stability, can be sources of backgrounds—and therefore of systematic effects—for the physics

measurements and need to be controlled as precisely as possible. Before FCC-ee, the use of the codes described in Sect. 2.2 was restricted to a few experts who were producing estimates of what turned out to be small effects, which were possibly mentioned as upper limits on systematic errors in final analysis. The only beam-related effects included by physicists in the simulation of the detector response were spreads in the beam energy and the position of the effective interaction point, which are important and non-negligible but do not cover the full picture. The unprecedented design luminosities of FCC-ee require a better evaluation of all the effects, including those of Sect. 2.3, which can only be obtained by simulating these backgrounds in the experimental apparatus to properly estimate detector occupancies and the level of spurious objects, such as additional tracks. This requires inter-operability of the relevant codes with FCCSW.

3.2 Interplay between accelerator and experiment codes

There are several levels of software programs that can inter-operate. The one that we will pursue in this case is one which is at the lowest level and which goes through common data formats and which can also work for programs running on different hardware or operating systems. We have seen in Sect. 2 that the codes for MDI-induced backgrounds typically produce outputs in the form of formatted text files. There is no common output format for all the programs, but there is enough information to understand the outputs and use them in other contexts.

The underlying idea is to develop a set of Gaudi readers and/or converters to inject the events produced by the MDI-background codes in the data processing chain.¹

FCCSW will then simulate the interaction of these particles in the detector to evaluate occupancies and levels of spurious objects. Eventually FCCSW will provide the possibility to overlay these events on signal events for a more detailed background simulation, possibly with a weighted mixture of MDI processes.

3.3 Towards an MDI “supercode”

By “supercode”, we do not mean a new big program doing everything but a common interface to all relevant codes to effectively simulate a single big program behaviour. The ultimate goal is that physicists wanting to study these backgrounds are able to do so from any of the supported computing infrastructures.

A software ecosystem based on Gaudi allows an approach of this type. The relevant components which need to be provided and/or identified are the following:

1. A shared file system available on the computing infrastructures supported for the project;
2. A software stack providing all the relevant applications built in a coherent way;
3. A set of good default configuration files available in the shared file system accessible by the relevant applications;
4. A wrapper to run external applications in Gaudi;
5. A set of Gaudi readers and converters, as mentioned in the previous section;
6. A set of application command line controls covering all the identified needs.

For the shared file system, the obvious choice is CernVM-FS [49, 50] which is now ubiquitous in HEP communities and beyond. The Key4hep stack [51] is the choice for 2, with all

¹ While in the default running mode inter-operability is through persistent files, the availability of dedicated readers and/or converters opens the way for alternative inter-operation options, for example, through FIFO channels [48].

the stack software available under `/cvmfs/sw.hsf.org/`. Some of the relevant codes, e.g. GUINEAPIG, are already available in Key4hep. Part of the work would be to make sure that all the relevant software is in a form suitable for being added and maintained in the common stack. Good default configuration files should be the result of the detailed evaluation of each of the codes mentioned earlier; they could be stored on the shared CernVM-FS repository, though the possibility to use different settings should be maintained. The integration with Gaudi, points 4 and 5, is part of the work, and no insurmountable issues are anticipated. The identification of a common set of switches to cope with all the codes will need some iterations, though, again, it should not pose insolvable problems.

The whole integration process just described is being proof-tested with GUINEAPIG with promising results already. Detailed documentation of this prototype work and of course of all the components described in this subsection is part of the work. It should contain examples at different levels, together with a detailed reference guide.

4 Aspects related to geometry description

The description of the geometry and materials of the relevant accelerator and detector elements is a crucial ingredient of most of the codes discussed in this essay. Having a coherent description based on the same source of information is highly desirable for several reasons, not least to reduce the risk of errors due to several implementations of the same item.

The design and implementation of these elements follows different paths depending on the nature of the element. *Detector components* are designed starting from a detector concept, and then, it is mostly space constraints that are applied. The tool chosen to describe the relevant concepts is DD4hep [47], an open source toolset introducing the compact detector description concept, provided through minimalistic XML formats, to allow composition of basic sub-detector elements to form complex detector structures. DD4hep was developed for conceptual design studies and initially applied to linear collider cases. However, its flexibility and generality quickly appealed the LHC community; as of today, DD4hep has been adopted by CMS for use starting from Run3 and is being seriously considered by LHCb and ATLAS. *Accelerator elements* usually arise from optimisation studies carried on with CAD engineering tools which allow working in 3D with the solid features which are essential for the task. Currently, the tool mostly used is AUTODESK INVENTOR [52], though CATIA [53] is currently being evaluated; both these tools are commercial, because the open-source CAD tools currently available do not provide the required quality.

While there are good reasons for both approaches, the desirable requirement to have the same source of information becomes a challenge, because a satisfactory conversion procedure of CAD supported formats to DD4hep is currently missing. DD4hep has some capabilities to read CAD formats through an interface to the external open-source library ASSIMP [54]. As well as the fact that only read support is currently implemented—which would result in one-way only conversion, namely CAD to DD4hep—the major limitations come from the CAD file formats currently supported by ASSIMP and the overlap with those supported by AUTODESK INVENTOR. Early investigations have shown that currently the only testable solution is to use the STL (Standard Tessellation Language) format [55]. However, this format has some inherent limitations, since it focuses on surfaces and does not seem to provide a natural way to describe material information, which is a must-have for any simulation activity. Other conversion options are under investigation such as the ones discussed in Ref. [24] or the suggestions available for GEANT4 users [16–24]. Finding a satisfactory solution to the

mutual conversion between CAD formats and DD4hep is clearly one of the challenges of the MDI detector studies.

5 Conclusion

In this essay, we reviewed the main aspects affecting accelerator codes and their interplay with experimental software. The existing accelerator codes are the result of many years of development and have been validated with respect to various accelerator facilities. Often two or more alternative codes for each of the different operational and experimental aspects exist. The challenge in these cases is to get full control of the codes, often not available in version repositories. We need to facilitate access and configuration and, when relevant, provide clear and, if required, solid ways of combining the results (see, for example, the case of synchrotron radiation).

For the integration with experimental software, the main challenge is to provide the relevant Gaudi components to enable the interplay between accelerators codes and the data processing chain. Finally, to achieve the objective of a single geometry source for all the components, a solid conversion solution to and from CAD will be required.

The interaction region design for FCC is particularly challenging and requires a combined optimisation of accelerator, engineering and experiment aspects. A comprehensive and flexible software environment as outlined in this essay will be very helpful for the MDI design of a future collider.

Funding Open access funding provided by CERN (European Organization for Nuclear Research). This project is co-funded from the European Union's Horizon 2020 research and innovation programme under grant agreement No 95175.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. FCC Collaboration, FCC-ee: the lepton collider. *Eur. Phys. J. C* **79**, 474 (2019). <https://doi.org/10.1140/epjst/e2019-900045-4>. The numbers have been taken from Table 2.1
2. M. Boscolo, H. Burkhardt, K. Oide, M.K. Sullivan, IR challenges and the machine detector interface at FCC-ee. EPJ+ Special Issue Part II. <https://doi.org/10.1140/epjp/s13360-021-02031-5>
3. K. Oide et al., *Phys. Rev. Accel. Beams* **19**, 111005 (2016)
4. G. von Holtey et al., Study of beam induced particle backgrounds at the LEP detectors. *Nucl. Instrum. Meth. A* **403**, 205 (1998)
5. H. Grote, C. Iselin, The MAD program (methodical accelerator design) version 8.4: user's reference manual. CERN-SL-90-13-AP-REV.2 (1991), <http://mad.web.cern.ch/mad/>
6. H. Grote, F. Schmidt, in MAD-X: an upgrade from MAD8. Proc. PAC 2003 and CERN-AB-2003-024-ABP
7. SAD, <http://acc-physics.kek.jp/SAD/index.html>
8. L. Deniau et al., Upgrade of MAD-X for HL-LHC project and FCC studies. <https://doi.org/10.18429/JACoW-ICAP2018-TUPAF01>
9. C. Iselin, The CLASSIC project. CERN-SL-96-61-AP. <https://doi.org/10.1063/1.52382>

10. M. Boscolo, H. Burkhardt, Tools for flexible optimisation of IR designs with application to FCC, pp. 2072–2074. <https://doi.org/10.18429/JACoW-IPAC2015-TUPTY031>
11. S. Agostinelli et al., Geant4—a simulation toolkit. Nucl. Instrum. Meth. A **506**, 250–303 (2003). J. Allison et al., Recent developments in Geant4. Nucl. Instrum. Meth. A **835** 186–225 (2016)
12. I. Antcheva et al., ROOT: a C++ framework for petabyte data storage, statistical analysis and visualization. Comput. Phys. Commun. **180**, 2499–2512 (2009). <https://doi.org/10.1016/j.cpc.2009.08.005>
13. R. Chytráček, J. McCormick, W. Pokorski, G. Santin, Geometry description markup language for physics simulation and analysis applications. IEEE Trans. Nucl. Sci., **53**(5), 2892–2896, <https://GDMML.web.cern.ch/GDMML/>
14. F. Collamati, M. Boscolo, H. Burkhardt, R. Kersevan, Synchrotron radiation backgrounds for the FCC-hh experiments, <https://doi.org/10.1088/1742-6596/874/1/012004>
15. F. Collamati, M. Boscolo, H. Burkhardt, R. Kersevan, “Synchrotron radiation backgrounds for the FCC-hh experiments”. J. Phys. Conf. Ser. **874**(1), 012004 (2017). <https://doi.org/10.1088/1742-6596/874/1/012004>
16. InStep, <https://www.solveering.com/InStep/instep.aspx>
17. SALOME, <http://www.salome-platform.org>
18. CADMesh, <http://code.google.com/p/cadmesh/>
19. Blender, <http://projects.blender.org>
20. STEP Solutions, <http://www.steptools.com/products/stdev/>
21. VTCAD, <http://www.cogenda.com/article/products#VTCAD>
22. SW2GDML, <https://github.com/cvuosalo/SW2GDMLconverter>
23. CadMC, <http://polar.psi.ch/cadmc/>
24. S. Boogert et al., PYG4OMETRY: a Python library for the creation of Monte Carlo radiation transport physical geometries, [arXiv:2010.01109](https://arxiv.org/abs/2010.01109) [physics.comp-ph]
25. H. Burkhardt, Monte Carlo generation of the energy spectrum of synchrotron radiation. CERN-OPEN-2007-018
26. G4beamline, <http://www.muonsinternal.com/muons3/G4beamline>
27. L.J. Nevay et al., BDSIM: an accelerator tracking code with particle-matter interactions. Comput. Phys. Commun. **252**, 107200 (2020)
28. M. Lüeckhof, Background processes affecting the machine-detector interface at FCC-ee with focus on synchrotron radiation at 182.5 GeV beam energy. PhD thesis, Hamburg University, January, urn:nbn:de:gbv:18-ediss-92722 (2021)
29. M.K. Sullivan, Unpublished, Originally made by Al Clark of LBNL
30. R. Kersevan, in “SYNRAD: a Monte Carlo synchrotron radiation ray-tracing program”. Conf. Proc. C930517, Washington, vol. 5, pp. 3848–3850 (1993)
31. H. Burkhardt, I. Reichel, G. Roy, Transverse beam tails due to inelastic scattering. PRSTAB **3**, 091001 (2000). <https://doi.org/10.1103/PhysRevSTAB.3.091001>
32. R. De Maria et al., “SixTrack version 5: status and new developments”. <https://doi.org/10.18429/JACoW-IPAC2019-WEPTS043>, <http://sixtrack.web.cern.ch/SixTrack/>
33. <https://xsuite.readthedocs.io/en/latest/>
34. D. Schulte, in 5th Intern. Computational Accel. Physic. Conf., Monterey, CA, USA, Spet. (1998). CLIC-NOTE 387
35. R. Kleiss, H. Burkhardt, BBBREM: Monte Carlo simulation of radiative Bhabha scattering in the very forward direction. Comput. Phys. Commun. **81**, 372 (1994). [https://doi.org/10.1016/0010-4655\(94\)90085-X](https://doi.org/10.1016/0010-4655(94)90085-X)
36. A. Ciarma, Talk presented at the FCC WEEK 2020, <https://indico.cern.ch/event/923801/contributions/4044075/>
37. M. Boscolo, O. Blanco-García, H. Burkhardt, F. Collamati, R. Kersevan, M. Lueckhof, “Beam-gas background characterization in the FCC-ee IR”. J. Phys. Conf. Ser. **1067**(2), 022012 (2018). <https://doi.org/10.18429/JACoW-IPAC2018-MOPMF085>
38. R. Kersevan, J.L. Pons, Introduction to MOLFLOW+: new graphical processing unit-based Monte Carlo code for simulating molecular flows and for calculating angular coefficients in the compute unified device architecture environment. J. Vacuum Sci. Technol. A **27**, 1017–1023 (2009). <https://doi.org/10.1116/1.3153280>. (Preprint)
39. M. Boscolo, P. Raimondi, Phys. Rev. ST Accel. Beams **15**, 104201 (2012). <https://doi.org/10.1103/PhysRevSTAB.15.104201>
40. E. Gianfelice-Wendt, Phys. Rev. Accel. Beams **19**(10), 101005 (2016). <https://doi.org/10.1103/PhysRevAccelBeams.19.101005>
41. J. Kewisch, “Simulation of electron spin depolarization with the computer code SITROS”. DESY-83-032, <https://inspirehep.net/literature/190447>

42. <http://www.classe.cornell.edu/bmad/>
43. See for example J. Cervantes et al., “A software framework for FCC studies: status and plans”, CHEP 2019. EPJ Web of Conferences vol. 245, p. 05018 (2020). <https://doi.org/10.1051/epjconf/202024505018>
44. See for example M. Clemencic et al., “Gaudi evolution for future challenges”, CHEP 2016. <https://doi.org/10.1088/1742-6596/898/4/042044>
45. The AIDA-2020 Collaboration, “AIDA-2020: 2nd periodic report”, CERN, Jun 2018. AIDA-2020-NOTE-2018-002, <https://cds.cern.ch/record/2628353>
46. See for example F. Gaede et al., “PODIO: recent developments in the Plain Old Data EDM toolkit”, CHEP 2019. <https://doi.org/10.1051/epjconf/202024505024>
47. M. Frank, F. Gaede, M. Petric, A. Sailer, “AIDAsoft/DD4hep”. <https://doi.org/10.5281/zenodo.592244>, <http://dd4hep.cern.ch/>
48. See, for example. <https://opensource.com/article/19/4/interprocess-communication-linux-channels>
49. J. Blomer et al., The CernVM file system. <https://doi.org/10.5281/zenodo.4114078>
50. J. Blomer et al., Distributing LHC application software and conditions databases using the CernVM file system. J. Phys.: Conf. Ser. **331**, 042003 (2011)
51. See for example G. Ganis, C. Helsens and V. Völkl, Key4hep, a framework for future HEP experiments and its use in FCC. Published in this issue
52. Autodesk Inventor. https://en.wikipedia.org/wiki/Autodesk_Inventor
53. CATIA. <https://en.wikipedia.org/wiki/CATIA>
54. OpenAssImp, The open asset import library, <https://github.com/assimp/assimp>
55. STL - Standard Tessellation Language, [https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format))