COMPUTING ENVIRONMENT FOR THE SUPER-CHARM-TAU FACTORY DETECTOR PROJECT

M. Belozyorova[1,2], D. Maksimov[1,2,a], G. Razuvaev[1,2], A. Sukharev[1,2], V. Vorobyev[1,2], A. Zhadan[1,2] and D. Zhadan[1,2]

[1] *Budker Institute of Nuclear Physics, 11, akademika Lavrentieva prospect, Novosibirsk, 630090, Russia*

[2] *Novosibirsk State University, 1, Pirogova street, Novosibirsk, 630090, Russia*

E-mail: [a] D.A.Maksimov@inp.nsk.su

The project of the Super Charm-Tau (SCT) factory – a high-luminosity electron-positron collider for studying charmed hadrons and tau lepton – is proposed by Budker INP. The project implies single collision point equipped with a universal particle detector. The Aurora software framework has been developed for the SCT detector. It is based on trusted and widely used in high energy physics software packages, such as Gaudi, Geant4, and ROOT. At the same time, new ideas and developments are employed, in particular the Aurora project benefits a lot from the turnkey software for future colliders (Key4HEP) initiative. This paper describes the first release of the Aurora framework, summarizes its core technologies, structure and roadmap for the near future. From the hardware point of view the Budker INP general computing facility (BINP/GCF) providing the required computational and storage resources is described.

Keywords: Super Charm-Tau factory, SCT, Aurora framework

Maria Belozyorova, Dmitry Maksimov, Georgiy Razuvaev, Andrey Sukharev,
Vitaly Vorobyev, Anastasiia Zhadan, Daniil Zhadan

1. Introduction

Proposed by Budker INP team Super Charm-Tau (SCT) factory [1] is a symmetric electron-positron collider with a single collision point going to operate in the energy range between $\sqrt{s}$ = 2 and 6 GeV. This energy range covers the charmonium family, several open charm hadrons thresholds and the τ lepton threshold, thus providing the rich physics program. Projected luminosity of the collider is 10^{35} cm^{-2} s^{-1} . Maximal event rate to deliver to the data storage is expected to be more than 350 kHz for operation at the J/ψ resonance. The collision point is equipped with a universal particle detector covering nearly full solid angle (Figure 1). The detector contains all traditional subsystems.

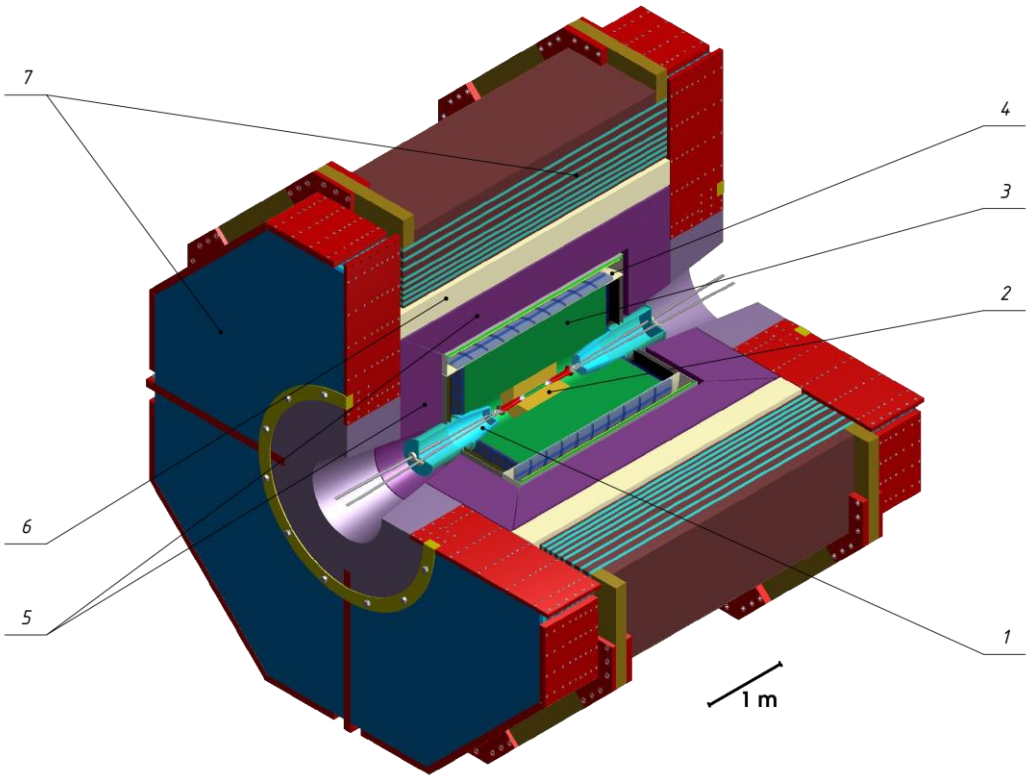

Figure 1. The SCT detector engineering sketch. 1 — Beam pipe and final focus magnets, 2 — Inner tracker, 3 — Drift chamber, 4 — PID system, 5 — Calorimeter, 6 — Magnet coil, 7 — Muon system and yoke.

2. Computing infrastructure

Computing infrastructure for the SCT project is provided by BINP General Computing Facility (BINP/GCF), which has about two thousands of CPU cores, several hundreds TB of HDD/SSD storage, and direct broadband connections to local area computing centers as well as to the LHCone network. These resources are available to BINP groups, either local or participating in remote experiments, and, in particular, to the SCT project. General scheme of the BINP/GCF and its external connections is presented at Figure 2.

3. Software

From the software point of view, the SCT experiment is a typical heavy flavor factory experiment. It requires a complete stack of relevant software, including: event generators, parametric and full detector simulation, event reconstruction algorithms, online event interpretation for trigger decisions, event data model (EDM), I/O interface to conditions data base, I/O interface to data storage, offline data analysis algorithms, build system and release management software. These components and their interconnections are usually referred to as the *software framework*.

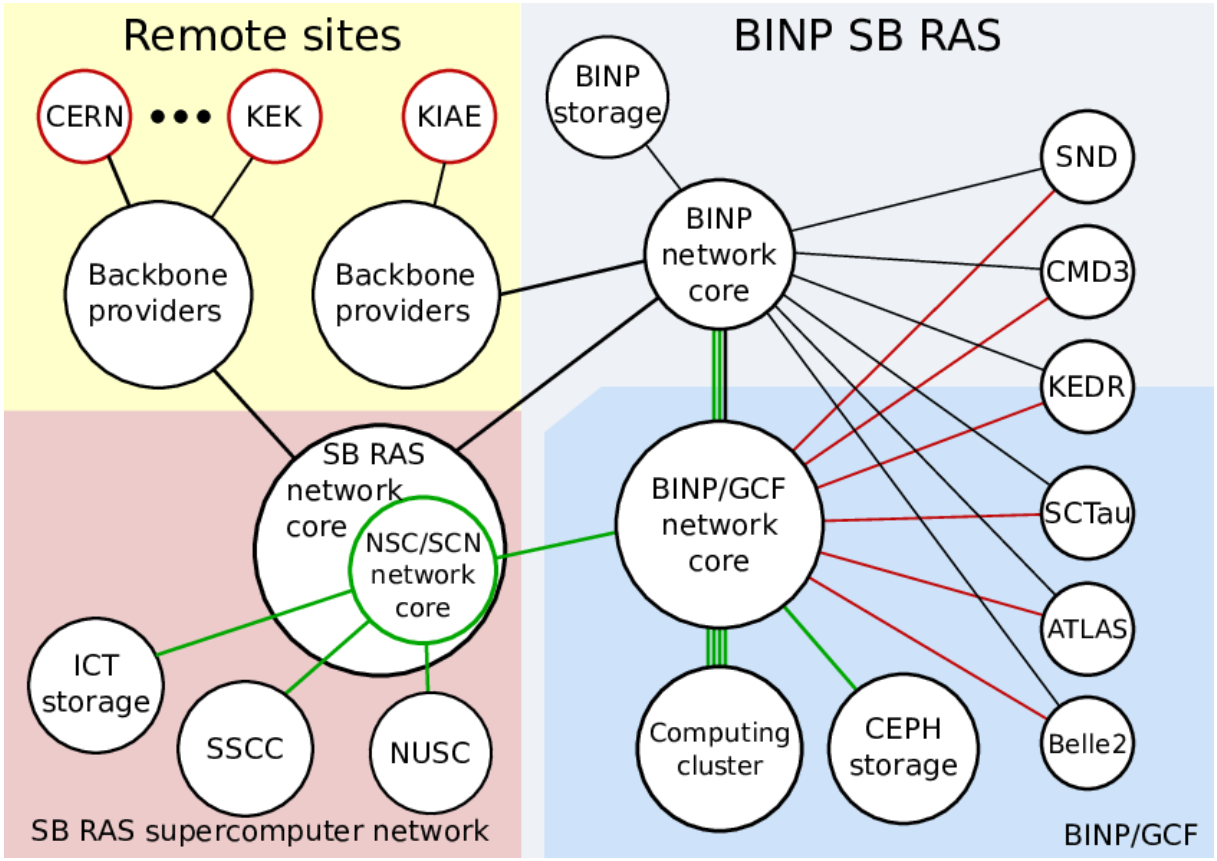

Figure 2. The BINP/GCF and external connections

Main components of the full framework and its information flows are illustrated in Figure 3. Although not all the components presented there are really mandatory at the detector design stage, we must keep them in mind to make further framework development easier.

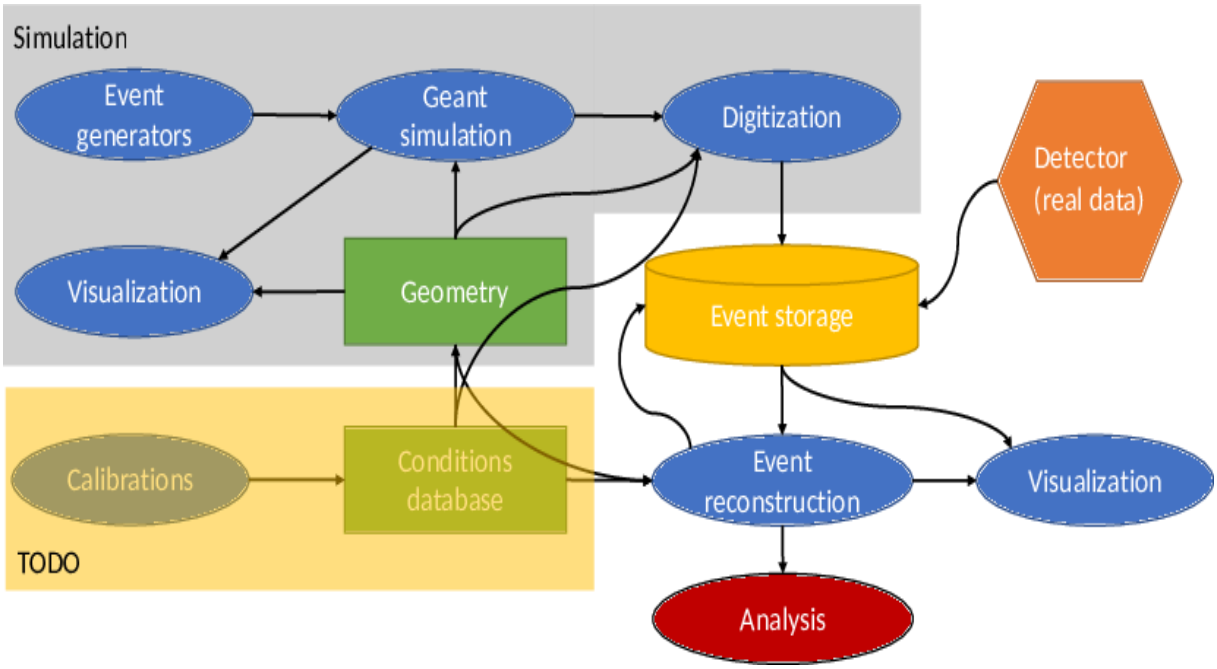

Figure 3. The SCT detector software and data flows scheme

We adopt the following guiding principles to work on the detector software:

1. Rely on trusted existing solutions where it is possible.

2. Adopt the best practices of collaborative software development, including the code style guidance and code reviewing procedure.
3. Be in touch with and contribute to the global community of the detector software developers.

In particular, we follow the Key4HEP [2] project and we look forward to benefit from it.

2.1 The framework

The SCT detector software framework is based on Gaudi [3] and has name Aurora. It covers all the aspects of software components interaction at runtime, including configuration, data exchange and job running. The framework architecture follows the traditional scheme: it separates code and data. Physics data are processed by Algorithms on event by event basis. An algorithm takes input data, manipulates it and produces new output data. Some parts of this work may be delegated to Tools. The framework provides data stores for data exchange between algorithms.

Tools and algorithms are implemented with C++. Python is used for configuration and steering scripts. The standard computing environment for the framework is Scientific Linux 7 at x86_64 architecture, with GCC 9 as the main compiler. Python 2 is still used, however transition to Python 3 is anticipated in the nearest future.

The data flow in the processing chain is conventional. Events are generated by the chosen generator, then injected into the simulation, yielding the event picture in the detector. The picture is processed with digitization modules, their output data format being exactly the same as it would be for the real detector hardware. Thus the reconstruction could operate with either simulated or real (in future) data. Reconstructed events are analyzed using corresponding high-level tools. On each stage of this chain the intermediate data could be routed directly from one module to another, or stored and read back later.

2.2 Event Generators

The required set of event generators should provide a reasonable description of e^+e^- interactions in the energy range between 2 and 6 GeV. Generators can be divided into three categories

1. Exclusive decays of hadrons and tau lepton provided by the EvtGen [4] and Tauola [5]. Final-state-radiation is simulated by the PHOTOS [6] generator.
2. Inclusive generators for $e^+e^- \rightarrow$ hadrons. This is a challenging task and only a preliminary solution based on Pythia [7] generator is implemented at the moment.
3. Generators for luminosity measurements and calibrations (MCGPJ [8], BabaYaga [9], BBBREM [10], KKMC [11] and other).

Primary generators produce output in the HepMC3 [12] format which is immediately converted to SCT EDM and can be serialized or used as input to the SCT detector simulation algorithm.

2.3 Detector geometry description

A uniform detector geometry description should be provided for all software modules thus making sure the same geometry is used at all stages of data processing. Each detector subsystem has a corresponding geometry software package. If there are several options for a subsystem, each option has a separate package. The DD4Hep [13] toolkit is our choice for this task. The geometry description consists of one or more XML files containing subsystem elements shapes, sizes, materials, visualization attributes, readout parameters etc. The corresponding C++ class interprets the XML description and creates all necessary objects.

2.4 Detector response simulation

This is actually a two-stage process. First, the detector simulation module uses Geant4 [14] for particle propagation and hit generation. DD4Hep provides a special tool DDG4 to supply the uniformed geometry description to Geant4.

Digitization, i. e. conversion of hits generated by Geant4 to a form of data read from real detector hardware (RawHit), is performed by ad-hoc algorithms, sometimes quite trivial, sometimes rather complicated, developed by subsystems experts. The RawHit consists of an electronics channel identifier and data "read" from the channel.

2.5 Event reconstruction

This is also two-level procedure. Basic reconstruction reads RawHit sets for the subsystem and uses them to restore positional/energetic/temporal information forming corresponding reconstructed hits. Each subsystem group should provide a basic reconstruction package for its subsystem.

Then, the reconstructed hits are combined in a subsystem or in several subsystems to form tracks, clusters or some another high-level objects. Among the options for the track reconstruction package GenFit [15] and ACTS [16] are considered at the moment.

Calorimeter reconstruction is based on a generic topo-cluster technique.

2.6 Parametric detector model

A parametric simulation is a tool to receive a detector response without detailed description of interaction of particles with matter. The detector response is obtained using random numbers whose distribution density is described with a small set of parameters.

The parametric simulation yields the detector response in the SCT EDM format thus allowing to analyze its result in the same manner as the result of the full simulation.

2.7 Offline data analysis

The Analysis module implements high-level tools for the offline analysis of reconstructed events. The analysis procedure is candidate-based meaning that candidates of the process of interest (e. g. $D0 \rightarrow K^{-}\pi^{+}$ decay) are searched in each event, and necessary information is saved for each candidate. The guiding principles behind software for the offline analysis are: 1) the analysis tools should implement all required data manipulations, 2) the analysis tools should be easy to use, 3) an analysis should be easily reproducible, and 4) the analysis result should be in a standardized format. These principles imply that a user would be forced to use only provided high-level tools and not process low-level data him or herself and impose strict requirements to the analysis tool: to be versatile, extendable, and user-friendly.

The following features are implemented in the Analysis module at the moment: accessing reconstructed final-state-particles (input for the analysis), formation of candidate list for arbitrary particle decay, imposing selection criteria, applying kinematic constraints to reconstructed particle decays, saving the resulting information to a flat ROOT TTree for further analysis (analysis output).

The Analysis module is going to be permanently improved and extended.

3. Conclusion

The SCT software development goes on promptly, for both the core framework and subsystem-specific parts, with more experimental groups joining the activity. Some of the framework components are absent yet (for instance, the conditions database), being irrelevant at the present stage of the project.

The Aurora framework now contains all tools minimally required at the present stage of the SCT detector project development. All described software is available as Aurora 1.0.1 release since June 2021.

4. Acknowledgments

We are grateful to the Belle II collaboration for access to the Belle II detector software [17]. We also thank the FCCSW project [18] and ATLAS experiment [19].

This work was supported by RSF grant 19-72-20114.

References

[1] V. V. Anashin et al. (SCT Colaboration), Super Charm–Tau Factory, Conceptual design report (draft), part one (physics program, detector), https://ctd.inp.nsk.su/wiki/images/4/47/CDR2_ScTau_en_vol1.pdf (2018), accessed 2021-02-25

[2] Key4HEP: Turnkey Software for Future Colliders, https://github.com/key4hep, accessed 2021-02-26

[3] The Gaudi project at CERN, http://gaudi.web.cern.ch/gaudi/, accessed 2021-02-25

[4] EvtGen Monte Carlo event generator, https://evtgen.hepforge.org/, accessed 2021-02-25

[5] Tauola, http://tauolapp.web.cern.ch/tauolapp/, accessed 2021-02-25

[6] Photos, http://photospp.web.cern.ch/photospp/, accessed 2021-02-25

[7] T. Sjöstrand, S. Mrenna, P. Skands, Comput. Phys. Commun. 178 852 (2008), arXiv:1410.3012, http://home.thep.lu.se/~torbjorn/Pythia.html

[8] A. B. Arbuzov, G. V. Fedotovich, F. V. Ignatov, E. A. Kuraev, A. L. Sibidanov, Eur. Phys. J. C 46, 689 (2006), arXiv:hep-ph/0504233

[9] G. Balossini, C. M. Carloni Calame, G. Montagna, O. Nicrosini, F. Piccinini, Nucl. Phys. B758 227-253 (2006), arXiv:hep-ph/0607181

[10] R. Kleissa, H. Burkhardt, Comput. Phys. Commun. 81 372 (1994), arXiv:hep-ph/9401333

[11] S. Jadach et al., Comput. Phys. Commun. 130 260 (2000), arXiv:hep-ph/9912214

[12] A. Buckley et al., Comput. Phys. Commun. 260 107310 (2021), arXiv:1912.08005, http://hepsoftware.org/e/hepmc3

[13] Detector Description Toolkit for High Energy Physics, https://dd4hep.web.cern.ch/dd4hep/, accessed 2021-02-25

[14] Geant4 simulation toolkit, https://geant4.web.cern.ch/, accessed 2021-02-25

[15] GenFit Package, https://github.com/GenFit/GenFit, accessed 2021-02-25

[16] Acts Common Tracking Software, https://github.com/acts-project/acts, accessed 2021-02-25

[17] T. Kuhr, C. Pulvermacher, M. Ritter, T. Hauth, N. Braun, Computing and Software for Big Science 3, Article number: 1 (2018),

[18] Software for the Future Circular Collider, http://hep-fcc.github.io/FCCSW/, accessed 2021-02-26

[19] The ATLAS Experiment, https://atlas.cern/, accessed 2021-02-25