A modular community ecosystem for multiphysics particle accelerator modeling and design

Jean-Luc Vay*[1], Axel Huebl[1], David Sagan[2], David Bruhwiler[3], Rémi Lehe[1], Cho-Kuen Ng[4], Ao Liu[5], Ji Qiang[1], Robert Ryne[1], Eric Stern[6], Alex Friedman[7], David Grote[7], Maxence Thévenet[8], Henri Vincenti[9], Spencer Gessner[4], Benjamin Cowan[10], Andreas Adelmann[11], Michael Bussmann[12,13], Sergei Bastrakov[12], Brigitte Cros[14], Daniel Winklehner[15], and S. R. Yoffe, G. K. Holt[16]

[1]Lawrence Berkeley National Laboratory, Berkeley, CA, USA
[2]Cornell University, Ithaca, NY, USA
[3]RadiaSoft LLC, Boulder, CO, USA
[4]SLAC National Laboratory, Menlo Park, CA, USA
[5]Euclid Techlabs, Bolingbrook, IL, USA
[6]Fermi National Accelerator Laboratory, Batavia, IL, USA
[7]Lawrence Livermore National Laboratory, Livermore, CA, USA
[8]DESY, Hamburg, Germany
[9]LIDYL, CEA-Université Paris-Saclay, CEA Saclay, 91 191 Gif-sur-Yvette, France
[10]Tech-X Corporation, Boulder, CO, USA
[11]Paul Scherrer Institut, Villigen, Switzerland
[12]Helmholtz-Zentrum Dresden-Rossendorf (HZDR), Institute for Radiation Physics, Dresden, Germany
[13]Center for Advanced Systems Understanding (CASUS), Görlitz, Germany
[14]CNRS, Université Paris Saclay, Orsay, France
[15]MIT, Cambridge, MA, USA
[16]SUPA and University of Strathclyde, Glasgow, United Kingdom

August 31, 2020

*jlvay@lbl.gov

1 Existing landscape of accelerator codes

Simulations are critical to the design, development and operation of all accelerator facilities, some costing in the billions of U.S. Dollars. The simulation of all the components and physical processes that are involved is a very wide-ranging and complex endeavor. While a lot of efforts have been aimed at the development and maintenance of very useful and successful simulation tools, they were largely uncoordinated. This resulted in a rich collection of (often relatively small, with some large and sophisticated) codes that are not interoperable, use different I/O formats and quite often duplicate some physics functionalities using the exact same underlying algorithms (e.g., many codes have standard Particle-in-Cell cores that are distinct but algorithmically identical), impoverishing the effective diversity of the whole.

Accelerator and beam physics software, like many other types of software, are subject to a 'natural cycle', driven by, e.g., developers retiring or moving to other projects, evolution of programming languages or computer hardware, leading to regular retirement or loss of codes that may need to be rewritten from scratch. The rewriting can be particularly costly when the previous code is poorly documented (if at all) and the original developer(s) is/are not available.

Another challenge accelerator physicists face is related to multiphysics integration in that most accelerator codes only implement modeling and simulation of specific processes, such as RF field calculations, particle tracking, space charge effects, and so forth. Taking RF cavity modeling as an example, part of the emitted particles in the cavity structure are accelerated to relatively high energy (MeV or higher) level and are lost on the wall. Surface impact creates secondary electrons (and cascaded activation of free electrons) and associated photon radiation. Researchers need to specify the radiation level to radiation safety personnel before the cavity can run at high power. Thus, the integration of RF modeling, particle tracking and particle-matter interaction simulation is highly desired by the community.

However, the lack of standardization of I/O between the many codes that exist means that it is difficult to (a) build workflows that link several codes together to solve a larger problem or solve a problem with more physical processes, (b) benchmark and validate the various codes against each other and known solutions, and (c) establish reproducibility for new scientific results. While standards are being developed and adopted that can alleviate such issues [1], the next level of integration, beyond the development of I/O standards, would require the development of an ecosystem of software that are interoperable and whose components can be assembled into topical packages, toolkits and start-to-end workflows.

This transition to a more coordinated and collaborative approach is also driven by the need to transition a large body of software from CPUs to GPUs, a transition that is more disruptive than most past transitions, including the transition from serial to parallel codes (and it is anticipated to be one of many transitions needed to adopt to the rapidly evolving computer hardware).

2 Ecosystem of modeling tools

Rather than starting from scratch, we propose to adopt an existing ecosystem framework and tools, and build our beam and accelerator modeling ecosystem on it. Leveraging on the Interoperable Design of Extreme-scale Application Software (IDEAS) project [2] and its Extreme-scale Scientific Software Development Kit (xSDK) [3], an ecosystem of software for the modeling of beam and particle accelerator physics can be established.

The main goal of the IDEAS project is to "help move scientific software development toward an approach of building new applications as a composition of reusable, robust, and scalable software components and libraries, using the best available practices and tools."[2], while the xSDK project is being developed to "provide a coordinated infrastructure for independent mathematical libraries to support the productive and efficient development of high-quality applications" [3]:

> Rapid, efficient production of high-quality, sustainable extreme-scale scientific applications is best accomplished using a rich ecosystem of state-of-the art reusable libraries, tools, lightweight frameworks, and defined software methodologies, developed by a community of scientists who are striving to identify, adapt, and adopt best practices in software engineering. The vision of the xSDK is to provide infrastructure for and interoperability of a collection of related and complementary software elements—developed by diverse, independent teams throughout the high-performance computing (HPC) community—that provide the building blocks, tools, models, processes, and related artifacts for rapid and efficient development of high-quality applications.

Although xSDK targets "extreme-scale scientific applications" of relevance to exascale supercomputing, its derived community policies apply well to all scales of computing, from laptops to clusters. Hence, the paradigm and tools are readily applicable to the full set of modeling needs of the particle accelerator community.

Individual packages in such an ecosystem are composable libraries, frameworks, and domain components developed by individual groups in the community. Each package publishes technical design documents, documentation (reference, tutorials, how-to guides), API definitions, and a concrete implementation [4]. In order to solve a specific physics case, a typical application uses functionalities from several packages, often in an innovative way.

There are many advantages to adopting policies and tools such as the IDEAS/xSDK:

- Community policies have been established over years by teams of specialists in scientific software development and computing, and they include best practices in software development.
- For reusable components, the policies require the use of *permissive open source licenses* ("Non-critical optional dependencies can use any OSI-approved license.") that allow reuse of source and binary code, including for commercial and proprietary applications, fostering collaborations across laboratories, academia and industrial partners.
- A wide-ranging set of open source, interoperable tools from a variety of backgrounds (including numerical solvers, e.g., Hypre, multi-parameter optimizer, e.g. LibEnsemble, and mesh-refinement frameworks, e.g. AMReX) can be combined and used as foundation of accelerator and beam physics toolkits and codes.
- A wide community of developers that can help improve software capability and sustainability across projects.
- Partial overlap in functionality is not problematic and in fact improves the diversity of the whole.
- Time of development and maintenance of domain specific software is greatly reduced, as the domain scientists can concentrate on the domain-specific functionalities while lower-level, cross-cutting, numerical packages are maintained by dedicated specialists.
- Portability across platforms (CPUs, GPUs, etc.) of low-level libraries ensures portability of the software that is built upon them. Building new tools on these low-level libraries greatly reduces the overall burden on the community for porting codes to new platforms.
- The domain-specific packages that are built, following established policies are portable across platforms and interoperable, and can be further combined to form larger toolkits and codes.
- Since packages are reused across software and duplication is minimized, bugs and inefficiencies are spotted earlier.
- Many codes, libraries and packages exist in the community that can be reused as building blocks, and progressively modernized and blended in a module that is portable and runs efficiently on modern CPUs and GPUs. Hence, not all functionalities have to be rewritten from scratch at once, offering a progressive path from the current status to an ecosystem of accelerator modeling tools.

It is to be emphasized that the goal is not to propose that only one code be developed to study, e.g., RF accelerators or plasma-based accelerators. Neither is it to channel support to a particular team or collaboration. On the contrary, the goal is to provide a coordinated infrastructure that enables inclusive and collaborative development of modeling tools for the community, and for these tools, to follow modern practices for scientific software: be portable, efficient, robust and leading to consistently accurate and reproducible results. This will also level the playing field for new researchers (U.S. or international) who can then easily contribute their new ideas and have them available for use, testing and further improvements by the entire community.

3 Outlook

The development of a modular ecosystem of accelerator and beam modeling tools would enable the modeling capabilities of the community to reach another level of breadth and depth. The libraries (e.g., Poisson solvers library [5]), toolkits (e.g., beam dynamics toolkit [6]) and codes (e.g., for electron cooling [7] or plasma accelerators [8]) that will be developed as part of the ecosystem can be further combined with standardized workflows [1], augmented with machine learning and surrogate models [9], to enable End-to-end Virtual Accelerators (EVA) [10]. Center(s) dedicated to accelerator and beam modeling [11] (including accelerator and beam physicist collaborating with applied mathematicians, computer scientists and software engineers), coordinated in consortia (such as, e.g., CAMPA [12]), would be especially well fitted for the development and maintenance of such an ecosystem of accelerator and beam modeling tools.

References

[1] Axel Huebl et al. “Develop/integrate data standards & start-to-end workflows for Accelerator Physics”. In: *Snowmass21 LOI* (2020).

[2] *IDEAS: Interoperable Design of Extreme-scale Application Software.* URL: `https://ideas-productivity.org`.

[3] *xSDK: Extreme-scale Scientific Software Development Kit.* URL: `http://xsdk.info`.

[4] Rémi Lehe et al. “Embracing modern software tools and user-friendly practices, when distributing scientific codes”. In: *Snowmass21 LOI* (2020).

[5] Ji Qiang and He Zhang. “A Parallel Poisson Solver Library for Accelerator Modeling Applications”. In: *Snowmass21 LOI* (2020).

[6] David Sagan et al. “Beam dynamics toolkit”. In: *Snowmass21 LOI* (2020).

[7] He Zhang. “Electron Cooling Simulation Based on First Principles”. In: *Snowmass21 LOI* (2020).

[8] C. B. Schroeder et al. “Plasma acceleration theory and simulation: status and needs”. In: *Snowmass21 LOI* (2020).

[9] Rémi Lehe et al. “Machine learning and surrogates models for simulation-based optimization of accelerator design”. In: *Snowmass21 LOI* (2020).

[10] Jean-Luc Vay et al. “EVA (End-to-end Virtual Accelerators)”. In: *Snowmass21 LOI* (2020).

[11] Jean-Luc Vay et al. “Center(s) for Accelerator and Beam Physics Modeling”. In: *Snowmass21 LOI* (2020).

[12] *CAMPA: Consortium for Advanced Modeling of Particle Accelerators.* URL: `http://campa.lbl.gov`.