

**A modular community ecosystem
for multi-physics particle accelerator modeling and design**

**Jean-Luc Vay, et al.
Berkeley Lab**

**AF7 - Subgroup RF - miniWorkshop on Innovative
Design and Modeling**

January 26, 2021



ACCELERATOR TECHNOLOGY &
APPLIED PHYSICS DIVISION



U.S. DEPARTMENT OF
ENERGY

Office of
Science

LOI written and supported by a team

A modular community ecosystem for multiphysics particle
accelerator modeling and design

Jean-Luc Vay^{*1}, Axel Huebl¹, David Sagan², David Bruhwiler³, Rémi Lehe¹, Cho-Kuen Ng⁴, Ao Liu⁵, Ji Qiang¹, Robert Ryne¹, Eric Stern⁶, Alex Friedman⁷, David Grote⁷, Maxence Thévenet⁸, Henri Vincenti⁹, Spencer Gessner⁴, Benjamin Cowan¹⁰, Andreas Adelman¹¹, Michael Bussmann^{12,13}, Sergei Bastrakov¹², Brigitte Cros¹⁴, Daniel Winklehner¹⁵, and S. R. Yoffe, G. K. Holt¹⁶

¹Lawrence Berkeley National Laboratory, Berkeley, CA, USA

²Cornell University, Ithaca, NY, USA

³RadiaSoft LLC, Boulder, CO, USA

⁴SLAC National Laboratory, Menlo Park, CA, USA

⁵Euclid Techlabs, Bolingbrook, IL, USA

⁶Fermi National Accelerator Laboratory, Batavia, IL, USA

⁷Lawrence Livermore National Laboratory, Livermore, CA, USA

⁸DESY, Hamburg, Germany

⁹LIDYL, CEA-Université Paris-Saclay, CEA Saclay, 91 191 Gif-sur-Yvette, France

¹⁰Tech-X Corporation, Boulder, CO, USA

¹¹Paul Scherrer Institut, Villigen, Switzerland

¹²Helmholtz-Zentrum Dresden-Rossendorf (HZDR), Institute for Radiation Physics,
Dresden, Germany

¹³Center for Advanced Systems Understanding (CASUS), Görlitz, Germany

¹⁴CNRS, Université Paris Saclay, Orsay, France

¹⁵MIT, Cambridge, MA, USA

¹⁶SUPA and University of Strathclyde, Glasgow, United Kingdom

August 31, 2020

LOI on ecosystem is (central) part in pool on “Shared Simulation Tools”*

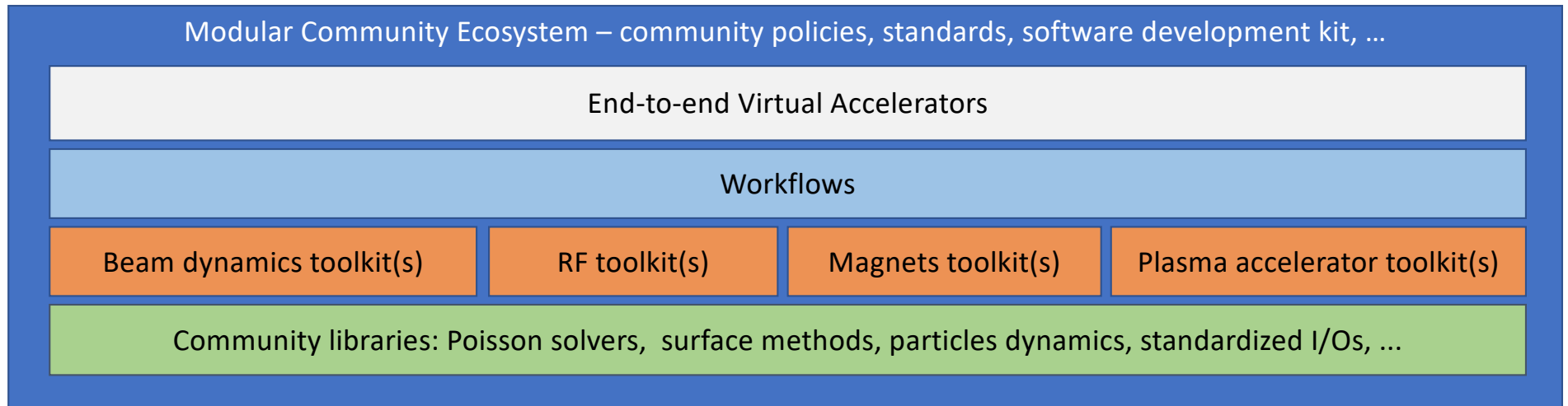
1. **A Parallel Poisson Solver Library for Accelerator Modeling Applications** - Ji Qiang, *et al.*
2. **Surface Methods for Precision Accelerator Design & Virtual Prototyping of Accelerator Systems** - Robert Ryne, *et al.*
3. **Beam Dynamics Toolkit** - David Sagan, *et al.*
4. **A modular community ecosystem for multiphysics particle accelerator modeling and design** - Jean-Luc Vay, *et al.*
5. **EVA (End-to-end Virtual Accelerators)** - Jean-Luc Vay, *et al.*
6. **Accelerator and Beam Physics: Grand Challenges and Research Opportunities** - Sergei Nagaitsev, *et al.*

*Full list of LOIs on Accelerator & Beam Physics Modeling at <https://snowmass-compf2-accbeammodel.github.io/>

Challenges being addressed and scope

- Many simulation tools support the accelerator community but often:
 - lack documentation, hard to modify, duplication (E.g., many beam dynamics codes, Poisson solvers) ;
 - different I/O → hard to compare & combine in multi-physics workflows;
 - little coordination.
- Huge cost to maintain and rewrite as codes die when author moves on.
- Developing efficient tools that run on new hardware (e.g., GPUs) is complex.

Proposed solutions & benefits to the community



Benefits

- Reusable components, portable across platforms (CPUs, GPUs, ...) with single source.
 - ➔ Reduced time for development with better, well benchmarked (less bugs), well documented, well supported and sustainable software stack.
- Easier sharing of data across programs and with ML engines.
- Benefits will extend across entire accelerator community with savings of millions \$.

Our community could leverage Exascale ecosystem

Interoperable Design of Extreme-scale Application Software (IDEAS) project <https://ideas-productivity.org/> & its Extreme-scale Scientific Software Development kit (xSDK) <http://xsdk.info/>:



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.

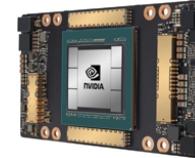
The vision of the xSDK is to provide infrastructure for and interoperability of a collection of related and complementary software elements - developed by diverse, independent teams throughout the high-performance computing (HPC) community- that provide the building blocks, tools, models, processes, and related artifacts for rapid and efficient development of high-quality applications.

Advantages of adopting policies & tools such as 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* 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 can be combined and used as foundation of accelerator and beam physics toolkits and codes
 - including numerical solvers (e.g., HyPre), multi-parameter optimizer (e.g., LibEnsemble), and mesh-refinement frameworks (e.g., AMReX).
- A wide community of developers that can help improve software capability and sustainability across projects.
- Time of development and maintenance of domain specific software is greatly reduced
 - ➔ the domain scientists can concentrate on the domain-specific functionalities while lower-level, cross-cutting, numerical packages are maintained by dedicated specialists.

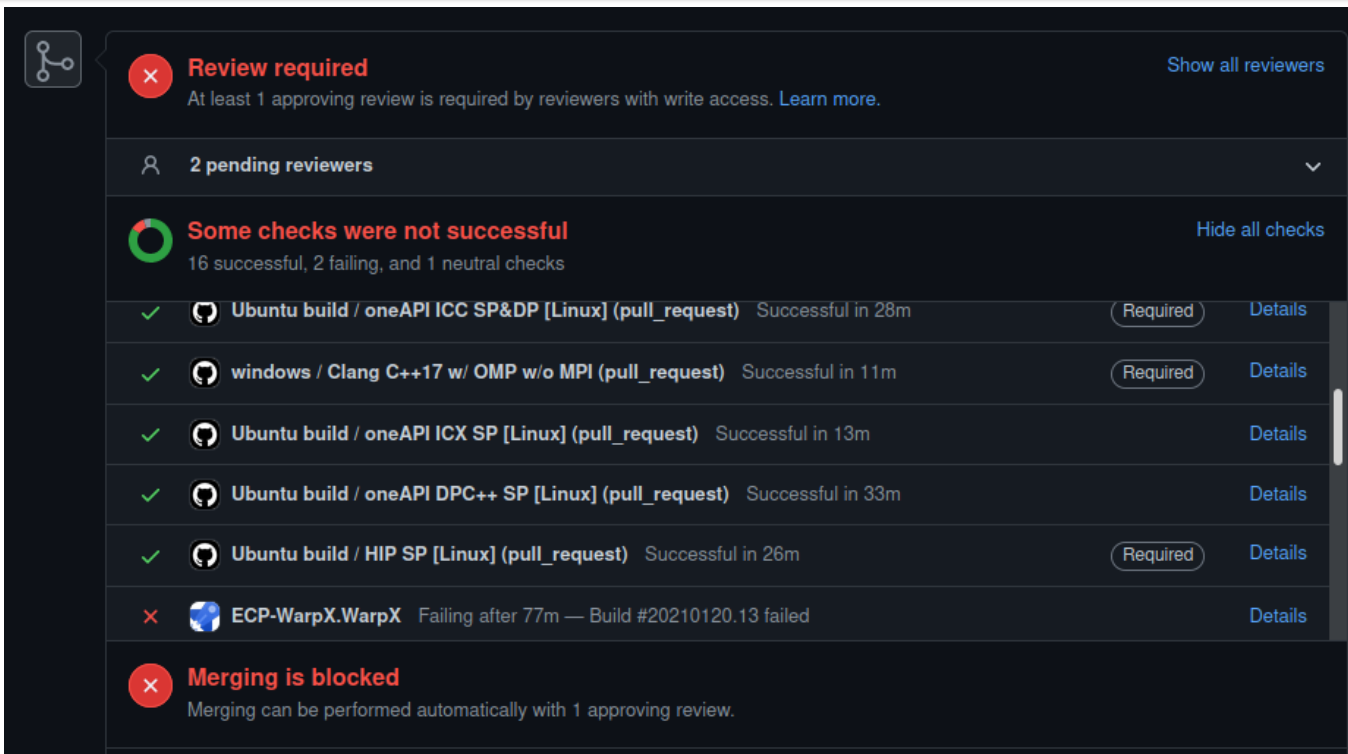
Advantages of adopting policies & tools such as IDEAS/xSDK - 2

- Portability across platforms (CPUs, GPUs, etc.) of low-level libraries ensures portability of the software that is built upon them.

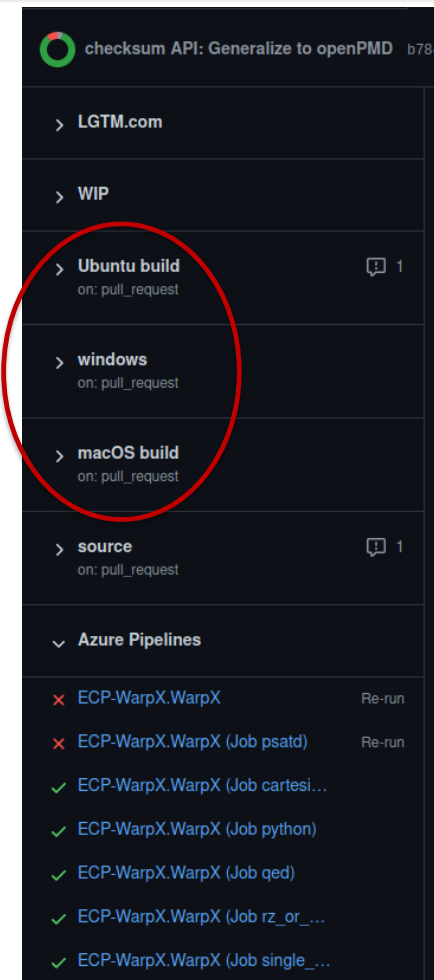


- Building new tools on these low-level libraries greatly reduces the overall burden on the community for porting codes to new platforms.
- The domain-specific packages that are built, following established policies are portable across platforms and interoperable
 - ➔ can be further combined to form 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 (across CPUs, GPUs, etc).
 - ➔ 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.

Continuous Integration is a great benefit for rapid development



This screenshot shows the GitHub Actions interface for a pull request. At the top, a red 'Review required' message states that at least one approving review is needed. Below this, it indicates '2 pending reviewers'. A green progress indicator shows that 'Some checks were not successful', with 16 successful, 2 failing, and 1 neutral checks. A list of workflow jobs follows, including 'Ubuntu build / oneAPI ICC SP&DP [Linux] (pull_request)' (Successful in 28m, Required), 'windows / Clang C++17 w/ OMP w/o MPI (pull_request)' (Successful in 11m, Required), 'Ubuntu build / oneAPI ICX SP [Linux] (pull_request)' (Successful in 13m), 'Ubuntu build / oneAPI DPC++ SP [Linux] (pull_request)' (Successful in 33m), 'Ubuntu build / HIP SP [Linux] (pull_request)' (Successful in 26m, Required), and 'ECP-WarpX.WarpX' (Failing after 77m). At the bottom, a red 'Merging is blocked' message indicates that merging can only be performed with one approving review.



This screenshot shows the details of a workflow run for 'checksum API: Generalize to openPMD'. The workflow is divided into several stages: 'LGTM.com', 'WIP', 'Ubuntu build' (on: pull_request), 'windows' (on: pull_request), 'macOS build' (on: pull_request), and 'source' (on: pull_request). The 'Ubuntu build' and 'windows' stages are circled in red. Below these, the 'Azure Pipelines' section is expanded, showing a list of jobs: 'ECP-WarpX.WarpX' (Re-run), 'ECP-WarpX.WarpX (Job psatd)' (Re-run), 'ECP-WarpX.WarpX (Job cartesi...)', 'ECP-WarpX.WarpX (Job python)', 'ECP-WarpX.WarpX (Job qed)', 'ECP-WarpX.WarpX (Job rz_or_...)', and 'ECP-WarpX.WarpX (Job single_...)'.

As developers commit changes, the code is automatically compiled on targeted platforms (Linux, macOS, Windows, ...) and physics benchmarks are run.

Final points

- **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
 - ➔ 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.

Thank you for your attention