Snowmass2021 Letter of Interest:

“Collaboration between industry and the HEP community”

D.L. Bruhwiler,[1,†] R. Nagler,[1] P. Moeller,[1] R. O’Bara,[8] D.T. Abell,[1] S. Baturin,[5] E. Carlin,[1] S. Coleman,[1] N.M. Cook,[1] J. Edelen,[1] C. Federer,[1] A.F. Habib,[2] C.C. Hall,[1] T. Heinemann,[2] B. Hidding,[2] A. Huebl,[4] M.V. Keilman,[1] R. Lehe,[4] E. Carlin,[1] P. Messamer,[1] B. Nash,[1] C.-K. Ng,[9] C.S. Park,[7] P. Piot,[5,6] I. Pogorelov,[1] E. Poore,[1] A. Sauers,[3] P. Scherkl,[2] J. Tourtellott,[8] J.-L. Vay[4] and S.D. Webb[1]

[1]RadiaSoft LLC, Boulder, Colorado 80301, USA
[2]University of Strathclyde, Glasgow G11XQ, UK
[3]Fermi National Accelerator Laboratory, Batavia, Illinois 60510, USA
[4]Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA
[5]Northern Illinois University, DeKalb, Illinois 60115, USA
[6]Argonne National Laboratory, Lemont, Illinois 60439, USA
[7]Korea University, Sejong Campus, Sejong, Republic of Korea
[8]Kitware, Inc., Clifton Park, New York 12065, USA
[9]SLAC National Accelerator Laboratory, Menlo Park, California 94025, USA

August 31, 2020

† bruhwiler@radiasoft.net

Abstract Effective long-term collaboration between national laboratories, academia and industry will lead to important benefits for the entire HEP community. Labs and universities will have access to better software with lower lifecycle costs. Companies will be strengthened by knowledge transfer from labs and universities. Computational scientists will be able to concentrate on core competencies, without spending time on UI design, ease of use, cloud computing, etc. Society will reap the benefits of better science, more innovation, and stronger businesses. State-of-the-art simulation codes will become readily available to students. Training time and associated costs will be reduced, as new team members will become productive more quickly. This will contribute to equity, diversity and inclusion (EDI), as barriers to entry are removed for scientists in developing countries and for those at US institutions with less federal funding and no direct access to code developers.

Statement of the problem The development and implementation of algorithms is a core competency of universities and research laboratories. When instantiated, the resulting codes often make use of a command line workflows which are error prone and difficult to reproduce. These workflows require excessive time and training to learn, involve multiple input and configuration files, execute on a high-performance server or cluster, necessitate post-processing with specialized software and additional visualization steps.

Professional software developers can make important contributions; however, they are expensive to hire and difficult to retain. Software sustainability and ease-of-use are very difficult to do well, but not especially interesting from the point of view of a computational physicist or computer scientist who needs to publish their work. Some excellent software developers and data scientists will be more easily hired, incentivized and retained in a small business environment, so close collaboration with industry can help to address career pipeline challenges with which the community is struggling.

Effective partnerships between industry and national laboratories, as well as between industry and universities, are necessary to maximize the productivity of available software development resources.

One example – particle accelerator codes There are many world class particle accelerator design codes that are freely available to the community – a small subset includes MAD-X [1, 2], elegant [3, 4], Synergia [5, 6], Zgoubi [27], OPAL [8], Warp [9-12] and JSPEC [13, 14]. These codes offer wide ranging capabilities, with significant overlap, but each with uniquely important features. Each code varies in the difficulty required to compile and install, as well as the quality and quantity of user documentation.

Typically, users must learn a command-line workflow, which may involve script development and/or the understanding and editing of multiple input and configuration files. Typically, the codes are run in parallel on a Linux cluster or supercomputing center. The codes generate a variety of output and simulation results, sometimes in multiple files, using plain text and binary formats. Visualization and post-processing generally requires specialized software. The resulting workflows can be idiosyncratic, opaque and brittle. Some code development teams provide user support, but generally the important details of these complicated workflows are not available to scientists who don't have a good connection to expert users at major institutions.

Ease-of-use is important -- a fact that is widely recognized by the development teams and by the community. However, it is not practical to develop a GUI for each separate code. The importance of code benchmarking and inter-comparison is also widely recognized, but there is not much incentive or reward for such efforts, so it is necessary to better facilitate the use of many codes together. A closely related problem is the difficulty of using multiple codes in sequence for beginning-to-end simulation of complex facilities.

Reproducibility and long-term sustainability are two important and related difficulties, which are not always adequately addressed. Simulations play an essential role in high-energy particle accelerator facilities over a period of decades, from pre-conceptual design, to final design, to commissioning and onward to a sequence of upgrades. It is essential that project scientists are able to reproduce past simulation results and to understand whether differences arise from improved modeling capabilities, changes in the design, or other factors. These concerns apply to many application codes throughout high energy physics.

Some requirements for success In order to facilitate the necessary collaborations and to provide the entire community with confidence that the software will be widely available and adequately supported over decades, an open source license is required for the industry software and can be very helpful for the entire software ecosystem [28]. This imposes an open source business model on the corresponding businesses, at least with regard to this specific activity. The software design objectives must include seamless integration with legacy codes, low barrier to entry for new users, easily moving between GUI and command-line modes, cataloging of provenance to aid reproducibility, and simplified collaboration through multimodal sharing.

First example: Sirepo Sirepo is an open source framework [15-19] for bringing scientific, engineering or educational software to the cloud, with a GUI that works in any modern browser on any computing device with sufficient screen size, including tablets. The Sirepo client is built on HTML5 technologies, including the JavaScript libraries Bootstrap [20] and Angular [21]. The D3 library [22, 23] is used for 2D graphics, while VTK [24] is used for 3D. The supported codes and dependencies are containerized via Docker [25], an open platform for distributed applications. RadiaSoft has developed open source software and expertise for building, deploying and executing scientific codes in Docker containers, and the corresponding images are publicly available [26]. These containers are compatible with the Shifter containerization technology at the NERSC supercomputing center, which enables a Sirepo server to automatically launch jobs at NERSC.

A free Sirepo scientific gateway is available to the particle accelerator community [17], providing a broad selection of supported codes. The accelerator tracking codes include MAD-X, elegant, Synergia, OPAL and Zgoubi. Presently under development, the MAD-X sequence file format will be used as a common format to enable rapid code benchmarking and sequential use of multiple codes. The loosely coupled cloud-based architecture of Sirepo enables coupling with other sophisticated software and systems. This is another reason for the enterprise approach. At NSLS-II for example, the DAMA group is integrating Sirepo/SRW with their BlueSky [19, 27] software for experimental control and data management.

Sirepo has been designed to transcend the limitations that discourage many scientists from working with GUI-driven applications. Sirepo can import the necessary input, data or configuration files for the codes that it supports, so experts can quickly transfer their simulation results to a GUI user. Likewise, the GUI can export a zip file with everything needed to run the identical simulation from the command line.

Second example: Computational Model Builder CMB is an open source platform with integrated software tools designed to integrate all processes involved in the life cycle of numerical simulation [29]. CMB is developed with a modular, flexible architecture that has been customized for a number of different scientific fields including hydrology, computational fluid dynamics, and multiphysics casting simulation. In high energy physics, CMB provides a graphical user interface for the ACE3P accelerator modeling codes. In every CMB application, the primary goal is to simplify end-user effort and reduce the manual overhead often taken up by workflow and data management activities associated with simulation based design and analysis.

To prepare simulation inputs, CMB provides form-style fields for entering data combined with selection and highlighting of modeling geometry in 3D views. The input fields are automatically generated from XML template files that describe and organize the keywords making up the simulation code input specification. The UI includes syntax checking to reduce the likelihood of entering invalid data. At the backend, Python scripts are used to write the simulation input files based on the user-entered data. CMB allows new applications to be developed with less effort than custom UI software.

For simulation job execution, CMB relies on the Girder data management platform [30] as a middle-tier server connected between the desktop user and remote HPC or cloud-based systems. When users submit jobs from the CMB desktop, execution status is tracked continuously by Girder and reported back to the desktop. For ACE3P simulation, a Girder server has been deployed on the NERSC Spin platform for submitting and tracking simulation jobs. This system is in the process of being updated with additional resource-location services so that simulation results can be more easily traced back to their source data.

Because the CMB platform is built on ParaView [31], it provides the full set of ParaView postprocessing and visualization features. This includes remote visualization of simulation results and in situ visualization of interim results during execution. CMB and ACE3P are currently being updated to support these features so they can be seamlessly accessed from the CMB user interface, again reducing the effort required by scientific researchers. As with all aspects of the CMB design, the overall goal is to adapt to the needs of the simulation user, instead of requiring the user to adapt to the available computing environment and software tools. For applications that integrate geometry modeling, CMB also includes a geometry module for operations such as model creation, model modification, and discretization using external meshing technologies.

Conclusion Effective long-term collaboration between national laboratories, academia and industry will lead to important benefits for the entire HEP community.

References

[1] L. Deniau, H. Grote, G. Roy, F. Schmidt, “The MAD-X Program (Methodical Accelerator Design) Version 5.06.00 User’s Reference Manual,” http://mad.web.cern.ch/mad/releases/last-rel/madxuguide.pdf

[2] MAD-X homepage, http://madx.web.cern.ch/madx/

[3] M. Borland, “elegant: A flexible SDDS-compliant code for accelerator simulation”, in *Tech. report APS LS-287*, 2000.

[4] Y. Wang and M. Borland. “Pelegant: A Parallel Accelerator Simulation Code for Electron Generation and Tracking”. in *AIP Conf. Proc.*, vol. 877, 2006, p. 241.

[5] J. Amundson, P. Spentzouris, J. Qiang and R. Ryne, “Synergia: A 3D Accelerator Modelling Tool with 3D Space Charge”, *J. Comput. Phys.*, vol. 211, 2005, pp. 229-248.

[6] The Synergia homepage, https://web.fnal.gov/sites/synergia

[7] F. Méot, "Zgoubi users guide”, FERMILAB-TM-2010, 1997.

[8] The Object Oriented Parallel Accelerator Library (OPAL), https://gitlab.psi.ch/OPAL/src/-/wikis/home

[9] D. P. Grote *et al*., “The WARP Code: Modeling High Intensity Ion Beams”, in *AIP Conf. Proc*. **749**, 55 (2005).

[10] J.-L. Vay *et al*., “Novel Methods in the Particle-In-Cell Accelerator Code Framework Warp”, in *Comput. Sci. Disc.*, vol. 5, p. 014019, 2012.

[11] A. Friedman *et al*., “Computational Methods in the Warp Code Framework for Kinetic Simulations of Particle Beams and Plasmas”, in *IEEE Transactions on Plasma Science*, vol. 42, pp. 1321–1334, May 2014.

[12] The open source Warp development repository, https://bitbucket.org/berkeleylab/warp

[13] H. Zhang, J. Chen, R. Li, Y. Zhang, H. Huang, and L. Luo, “Development of the Electron Cooling Simulation Program for JLEIC”, in *Proc. 7th Int. Particle Accelerator Conf. (IPAC'16)*, Busan, Korea, May 2016, pp. 2451-2453. doi:10.18429/JACoW-IPAC2016-WEPMW014

[14] JLab Simulation Package for Electron Cooling (JSPEC) on GitHub, https://github.com/zhanghe9704/electroncooling

[15] D.L. Bruhwiler, D.T Abell, N.M. Cook, C.C. Hall, M.V. Keilman, P. Moeller, R. Nagler and B. Nash, “Knowledge Exchange Within the Particle Accelerator Community via Cloud Computing,” in Proc. Int. Part. Accel. Conf., THPMP046 (2019); https://accelconf.web.cern.ch/ipac2019/papers/thpmp046.pdf

[16] R. Nagler, P. Moeller, M.V. Keilman and E. Carlin; the Sirepo development repository on GitHub, https://github.com/radiasoft/sirepo

[17] Sirepo scientific gateway, https://www.sirepo.com/

[18] M.S. Rakitin, P. Moeller, R. Nagler, B. Nash, D.L. Bruhwiler, D. Smalyuk, M. Zhernenkov and O. Chubar, “Sirepo: an open-source cloud-based software interface for X-ray source and optics simulations,” Journal of Synchrotron Radiation **25**, 1877 (2018); https://doi.org/10.1107/S1600577518010986

[19] M. Rakitin, A. Giles, K. Swartz, J. Lynch, P. Moeller, R. Nagler et al., “Introduction of the Sirepo-Bluesky interface and its application to the optimization problems,” Advances in Computational Methods for X-Ray Optics, ed. O. Chubar and K. Sawhney, Proc. of SPIE **11493**, 1149311 (2020); https://www.spiedigitallibrary.org/conference-proceedings-of-spie/11493/1149311/Introduction-of-the-Sirepo-Bluesky-interface-and-its-application-to/10.1117/12.2569000.short?SSO=1

[20] The Bootstrap homepage, http://getbootstrap.com

[21] The AngularJS homepage, https://angularjs.org

[22] The D3 homepage, http://d3js.org

[23] M. Bostock, V. Ogievetsky and J. Heer. “D3 Data-Driven Documents”, in *IEEE Transactions on Visualization and Computer Graphics* **17**, p. 2301 (2011).

[24] The vtk.js homepage, https://kitware.github.io/vtk-js

[25] The Docker homepage, https://www.docker.com

[26] RadiaSoft scientific containers are publicly available on DockerHub, https://hub.docker.com/u/radiasoft

[27] GitHub repository for Bluesky/Sirepo integration, https://github.com/NSLS-II/sirepo-bluesky

[28] A. Huebl, et al., “Aspiration for Open Science in Accelerator & Beam Physics Modeling”, Letter of Interest to Snowmass21.

[29] Computational Model Builder, https://www.computationalmodelbuilder.org/

[30] Girder, a data management platform, https://girder.readthedocs.io/

[31] ParaView: Large Data Visualization Made Easier, https://www.paraview.org/overview