ISBN: 978-3-95450-208-0 doi:10.18429/JACoW-IPAC2019-THPMP046

KNOWLEDGE EXCHANGE WITHIN THE PARTICLE ACCELERATOR COMMUNITY VIA CLOUD COMPUTING*

D. L. Bruhwiler†, D. T. Abell, N. M. Cook, C. C. Hall, M. V. Keilman, P. Moeller, R. Nagler, B. Nash, RadiaSoft LLC, Boulder, USA

Abstract

The development, testing and use of particle accelerator modeling codes is a core competency of accelerator research laboratories around the world, and likewise for synchrotron radiation and X-ray optics codes at lightsource facilities. Such codes require time and training to learn a command-line workflow involving multiple input and configuration files, execution on a high-performance server or cluster, post-processing with specialized software and finally visualization. Such workflows are error prone and difficult to reproduce. Cloud computing and UI design are core competencies of RadiaSoft LLC, where the Sirepo framework is being developed to make state of the art codes available in the browser of any desktop, laptop or tablet. We present our initial successes as real world examples of knowledge exchange between industry and the research community. This work is leading to broader knowledge exchange throughout the community by facilitating education of students and enabling instantaneous sharing of simulation details between colleagues. Sirepo design objectives include: seamless integration with legacy codes, low barrier to entry for new users, configuration transfer to command-line mode, catalog of provenance to aid reproducibility, and simplified collaboration through multimodal sharing. The combination of intuitive browser-based GUIs and Sirepo's server-side application container technology enables simplified computational archiving and reproducibility. If embraced by the community, this could become an important asset for the design, commissioning and future upgrade of particle accelerator and X-ray beamline facilities.

SIREPO – A SOFTWARE FRAMEWORK

Sirepo is an open source framework [1] for bringing any 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 [2] and Angular [3]. The D3 library [4, 5] is used for interactive 2D graphics, while VTK [6] is used for 3D. The Sirepo server is built on Flask [7], a lightweight framework for web development with Python.

The scientific codes supported by Sirepo, and all of their dependencies, are containerized via Docker [8], which is an open platform for distributed applications. RadiaSoft has developed open source software [9] and expertise for building, deploying and reliably executing scientific codes in Docker containers [10, 11]. RadiaSoft docker images are publicly available [12] as part of the Sirepo ecosystem.

SIREPO – A SCIENTIFIC GATEWAY

RadiaSoft maintains a free scientific gateway for the particle accelerator community [13], which provides a broad selection of supported codes. The most mature implementations are for SRW (Synchrotron Radiation Workshop) [14-16] and elegant [17, 18]. SRW is an open source [19] physical optics code with powerful features for calculating synchrotron radiation and X-Ray optics, including successful detailed benchmarking against state-of-the-art X-Ray beamlines [20]. Sirepo/SRW has a growing number of users at synchrotron and free electron laser (FEL) user facilities around the world, including: NSLS-II, LCLS, APS and ALS in the USA, ELETTRA in Italy, European XFEL in Germany, ESRF and SOLEIL in France, PSI in Switzerland, Diamond in the UK and LNLS in Brazil.

Just as elegant is one of the most widely used codes in the world for particle accelerator simulation and design, Sirepo/elegant has many more users than the other supported accelerator codes, including regular classroom use at the US Particle Accelerator School (USPAS) [21, 22]. Other well-supported codes provide important capabilities: Synergia [23, 24] offers 2D and 3D space charge models and special features for simulating nonlinear integrable optics in the IOTA ring [25, 26], while Zgoubi [27] provides spin tracking and JSPEC [28, 29] models intrabeam scattering and electron cooling.

There are two Sirepo implementations of the massively-parallel open source particle-in-cell (PIC) code Warp [30-33]. Warp PBA enables use of Warp's quasi-3D electromagnetic PIC modeling of beam- and laser-driven plasma-based accelerators. Warp VND uses Warp's electrostatic PIC capabilities to simulate a wide range of problems, with near-term emphasis on thermionic converters and other types of vacuum nanoelectronic devices.

SIREPO ENTERPRISE

A Sirepo gateway can be installed and maintained by any institution with suitable computer servers and sysadmin expertise, or with RadiaSoft support. This *Sirepo Enterprise* approach enables, for example, internal control over intellectual property protection and user management. To our knowledge, three institutions have independent Sirepo gateways for private access to Sirepo/SRW: the National Synchrotron Light Source (NSLS-II) for the design and commissioning of X-Ray beamlines and experimental support, the Advanced Light Source (ALS) to simulate X-Ray

* This work is supported by the US Department of Energy under Award Nos. DE-SC0011237, DE-SC0011340, DE-SC0018719, DE-SC0015212, DE-SC0017181 and DE-SC0017162.
† bruhwiler@radiasoft.net

beamlines for the proposed ALS-2 upgrade, and La Trobe University for classroom instruction [34].

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 [35] software for experimental control and data management.

Sirepo can bring any scientific, engineering or educational software to the cloud, with a rich browser-based interface that works in any modern browser on any computing device, including tablets. For example, Sirepo is used with the magnetohydrodynamics code FLASH [36] to facilitate simulations of capillary discharges. It has also been used with an orders-of-magnitude faster algorithm for designing high-current electron linacs [37, 38]. Hence, the enterprise solution makes it possible for an institution to multiply the value of proprietary internal codes by wrapping them with a Sirepo GUI for use by all relevant staff.

TRANSCENDING GUI LIMITATIONS

Sirepo has been designed to transcend the limitations that discourage many scientists from working with GUI-driven applications. Foremost is the easy back and forth between existing command-line workflows and use of the Sirepo GUI. 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 for the user a zip file with everything needed to run the identical simulation from the command line. This enables an expert to immediately begin working with GUI-generated simulation results from another user. Some expert code users presently use the Sirepo GUI for particular value added features, and continue to use their command-line workflow as well.

A command-line workflow for accelerator design is typically on a server, cluster or supercomputer that is remotely accessible. This gives the designer freedom to work from their chosen computer, while the code runs remotely in parallel. In contrast, a GUI-driven scientific code typically runs on the user's desktop computer. Sirepo's cloud-based paradigm keeps the GUI on the user's desktop, laptop or tablet, while the code executes on a powerful server in the cloud. RadiaSoft is generalizing Sirepo for submitting jobs to supercomputers.

SIREPO FOR KNOWLEDGE EXCHANGE

Here we describe three different types of knowledge exchange that are facilitated by the development and use of Sirepo-based GUIs for community physics codes.

...between Industry and Research Institutions

The development, testing and use of particle accelerator modeling codes is a core competency of accelerator research laboratories around the world, and likewise for synchrotron radiation and X-ray optics codes at lightsource facilities. Such codes require significant time and training to learn a command-line workflow. As a result, some qualified scientists do not use these codes – for example, busy senior scientists or experimentalists.

Sirepo has been developed in part to leverage the scientific value of community codes by greatly expanding the user base. The associated costs are low compared to the financial and personal investments that have already been and continue to be made in these codes by the development teams and the funding agencies.

In many cases, the code development teams are small and the resources required for user support, bug fixes and maintenance, as well as new feature development, must compete with scientific and other institutional priorities. Hence, any GUI development cannot place any burden on their time or funding. This is consistent with the Sirepo philosophy, which treats every code as a unique system, making effective use of the same inputs and outputs that expert users work with on the command line.

This user interface design expertise yields intuitive browser-based GUIs for state-of-the-art physics codes at a reasonable cost. Together with application container development and other server-side expertise, this represents the knowledge that RadiaSoft is transferring to the academic research community. In return, RadiaSoft (and all Sirepo users) are receiving tremendous knowledge that has been encapsulated within these codes by world class scientists.

...between Scientific Collaborators

Cloud computing creates the opportunity for novel and powerful features. For example, Sirepo users can share their full simulation with colleagues instantly, just by messaging the URL from their browser. Neither person has to spend time installing the code or making sure that both parties have installed the same version with the same compiler flags. Time is not wasted with transferring input files, nor with struggling to interpret plots that are being made with different visualization tools.

We call this instantaneous collaboration. The rapid, error-free exchange of ideas and information that it makes possible is difficult to fully appreciate until you experience it directly. Both parties may be experts in use of the code, and they are primarily using the GUI to enable iterative exploration of ideas, as well as the occasional exchange of URLs so that each can see first hand the relative merits of the other person's design.

A more powerful use case is when one person is an expert in working with the code, while the other person is not, but does have relevant expertise and perhaps important insight into the problem. In this scenario, without Sirepo, the collaboration involves one person describing their simulation results to the other. With Sirepo, a fundamental change has occurred, in that both parties are interacting directly with the simulation results and independently pursuing ideas to better understand the physical system in question.

...between Students and Teachers

The first two examples of knowledge exchange are relatively symmetric. When educating students, the exchange is more one-sided. The instantaneous collaboration made possible by Sirepo is a feature that has great practical use for teaching.

For example, the instructor sets up an initial simulation in Sirepo and then posts the URL from their browser to a forum the students can access. Each student clicks on the link and is invited to create a copy of the example. They then follow provided instructions, which could represent a short classroom exercise, a homework problem or even a test. When each student completes the exercise, in addition to any required external documentation of what they did, they can email or otherwise message the link for their Sirepo simulation to the instructor. As part of the grading process, the instructor can make a quick copy of each student's simulation and then check whether certain expected changes were made correctly. The first author has personal experience with this workflow.

As noted previously, Sirepo/SRW has been used to help teach classes at La Trobe University [21]. About a dozen students have participated with good results. For USPAS, it is Sirepo/elegant that has primarily been used, including by the first author of this paper [24]. A recent class also used Sirepo/elegant [39] and reportedly with positive results [40]. In 2018, the Korean particle accelerator school used Sirepo/Synergia to good effect [41].

COMPUTATIONAL REPRODUCIBILITY

Every scientist agrees in principle that scientific research should be reproducible; however, this is an active field of study in itself, and there is consensus that much work needs to be done. See e.g. Ref.'s [42-44] and references therein. This topic is far too broad to discuss here in any detail. We present below how Sirepo technology makes positive contributions to the related issues of replicability and usability.

Computational Replicability

Here, we briefly consider the problem of *computational replicability*, by which we loosely mean that two scientists should be able to get identical results from the same code. It is possible for close collaborators to achieve computational replicability; however, success may remain elusive, even in this special case, due to technical issues: different hardware, different versions of the source code, different compilation flags, confusion regarding input and configuration files, confusion regarding one or more post-processing steps, etc. There will be additional obstacles for scientists from other institutions, including: software that is difficult to obtain, build, install or use; post-processing steps that are unspecified, unclear, or complex; simulations that require many processor-hours or create large data sets.

Sirepo-based codes satisfy the conditions of replicability in a restricted sense. For the code version in a particular Sirepo server instantiation, for those code features that are exposed to users by the Sirepo GUI, any scientist with access to this Sirepo server can replicate an original simulation in complete detail and to machine precision, as long as the original scientist shares the corresponding URL. This is the *instantaneous collaboration* feature of Sirepo.

The two scientists who require replicability may in some case be a single individual at two different points in time. For example, when a scientist returns to a simulation after weeks or months or perhaps even years, it can be difficult or even impossible for them to replicate the earlier results, due to changes in the system on which they were running.

Use of a Sirepo-supported code solves this problem for time periods short enough that the system has not been updated with a new code release or with other significant changes in the Docker image being deployed on the server. Sirepo also enables computational replicability for past instances of a supported code, although some work would be required to make this feature available to users.

Usability

Many physics codes are difficult to use, in part because they rely on complicated command-line workflows. They typically require computational experts to use correctly and even so, the assistance of someone experienced with that specific code may be required. If a code is very difficult to use, then the value of computational reproducibility can be greatly reduced.

Suppose for example that a single script is provided to execute all of the preprocessing, code execution, post-processing and visualization steps. This sort of black box experience may not enable understanding of what is being replicated. Reproducibility should enable understanding, comparison and further work, and the various steps required should each be clear and individually replicable. It should be possible to modify each step and see the results.

The intuitive browser-based GUIs provided by Sirepo successfully address several of these concerns.

CONCLUSION

We have described three different types of knowledge exchange that are facilitated by the development and use of Sirepo-based GUIs for community physics codes. Sirepo is an emerging framework for particle accelerator design and simulation and is making contributions toward the goal of computational reproducibility. Three Sirepo codes – SRW, elegant and Synergia – are used in the classroom.

ACKNOWLEDGMENTS

Many scientists have contributed to Sirepo through collaborations with RadiaSoft, their independent development efforts, using and testing various codes and helping us understand the problem, helping us understand a code, suggesting new features, etc. In particular, we sincerely thank the following individuals: O. Chubar, M. Rakitin, J.-L. Vay, G. Andonian and P. Tzeferacos.

This work is supported by the US Department of Energy Office of Science Award Nos. DE-SC0011237 (BES); DE-SC0011340, DE-SC0013855, DE-SC0015897 and DE-SC0018719 (HEP); DE-SC0015212, DE-SC0017181 (NP) and DE-SC0017162 (ASCR).

REFERENCES

[1] Sirepo on GitHub, `https://github.com/radiasoft/sirepo`
[2] The Bootstrap homepage, `http://getbootstrap.com`

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

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

[5] M. Bostock, V. Ogievetsky and J. Heer. “D3 Data-Driven Documents”, in *IEEE Transactions on Visualization and Computer Graphics,* vol. 17, pp. 2301–2309, 2011.

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

[7] The Flask homepage, http://flask.pocoo.org

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

[9] High-performance computing (HPC) container repository, https://github.com/radiasoft/containers

[10] R. Nagler, D. L. Bruhwiler, P. Moeller and S. D. Webb, “Sustainability and Reproducibility via Containerized Computing”, presented at Comp. Science & Eng. Software Sustainability and Productivity Challenges Workshop, 2015, arXiv:1509.08789.

[11] D. L. Bruhwiler *et al.*, “Cross-platform and Cloud-based Access to Multiple Particle Accelerator Codes via Application Containers”, in *Proc. 6th Int. Particle Accelerator Conf. (IPAC'15)*, Richmond, VA, USA, May 2015, pp. 720-723. doi:10.18429/JACoW-IPAC2015-MOPMN009

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

[13] The Sirepo Scientific Gateway, https://sirepo.com

[14] O. Chubar and P. Elleaume, “Accurate and Efficient Computation of Synchrotron Radiation in the Near Field Region”, in *Proc. 6th European Particle Accelerator Conf. (EPAC'98)*, Stockholm, Sweden, Jun. 1998, paper THP01G, pp. 1177.

[15] M. S. Rakitin *et al.*, “Sirepo: an open-source cloud-based software interface for X-ray source and optics simulations,” *Journal of Synchrotron Radiation*, vol. 25, p. 1877, 2018.

[16] B. Nash *et al.*, “Detailed X-ray Brightness Calculations in the Sirepo GUI for SRW”, in *AIP Conf. Proc.*, 2019, vol. 2054, p. 060080.

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

[18] 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.

[19] SRW on GitHub, https://github.com/chubar/srw

[20] L. Wiegart, M. Rakitin, A. Fluerasu, and O. Chubar, “X-ray optical simulations supporting advanced commissioning of the coherent hard x-ray beamline at NSLS-II”, in *Proc. Advances Comp. Meth. for X-Ray Optics IV*, 2017, 103880N.

[21] The US Particle Accelerator School (USPAS) homepage, http://uspas.fnal.gov

[22] USPAS course, “Simulation of Beam and Plasma Systems”, http://uspas.fnal.gov/programs/2018/odu/courses/beam-plasma-systems.shtml

[23] 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.

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

[25] S. Antipov, D. Broemmelsiek, D. L. Bruhwiler *et al.*, “IOTA (Integrable Optics Test Accelerator): facility and experimental beam physics program”, *Journal of Instrumentation*, vol. 12, 2017, T03002.

[26] C. C. Hall, D. L. Bruhwiler, J. P. Edelen, A. L. Romanov, A. Valishev, and N. Kuklev, “First Measurements of Nonlinear Decoherence in the IOTA Ring”, presented at the 10th Int. Particle Accelerator Conf. (IPAC'19), Melbourne, Australia, May 2019, paper WEPTS070, this conference.

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

[28] 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

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

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

[31] 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.

[32] 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.

[33] Warp on BitBucket, https://bitbucket.org/berkeleylab/warp

[34] Personal communication from G. van Riessen.

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

[36] B. Fryxell *et al.*, “FLASH: An Adaptive Mesh Hydrodynamics Code for Modeling Astrophysical Thermonuclear Flashes”, in *The Astrophysical Journal Supplement Series,* vol. 131, p. 273, 2000.

[37] S. V. Kutsaev, Y. Eidelman and D. L. Bruhwiler, “Generalized 3D beam dynamics model for industrial traveling wave linacs design and simulations”, *Nuclear Instr. and Meth. in Phys. Research A*, vol. 906, p. 127, 2018.

[38] S. V. Kutsaev, Y. Eidelman, D. L. Bruhwiler, P. Moeller, J. F. Barbe Welzel and R. Nagler, “Cloud-based design of high average power traveling wave linacs”, *J. Phys.: Conf. Series*, vol. 941, p. 012106, 2018.

[39] USPAS course, “Fundamentals of Accelerator Physics and Technology with Simulations and Measurements Lab,” http://uspas.fnal.gov/programs/2019/knoxville/courses/fundamentals.shtml

[40] Personal communication from K. Ruisard.

[41] Personal communication from C. S. Park.

[42] S. Feger, S. Dallmeier-Tiessen, A. Schmidt and P. Woźniak, “Designing for Reproducibility: A Qualitative Study of Challenges and Opportunities in High Energy Physics”, 2019, arXiv:1903.05875

[43] D. B. Allison, R. M. Shiffrin and V. Stodden, “Reproducibility of research: Issues and proposed remedies”, *PNAS*, vol. 115, 2018, p. 2561, https://doi.org/10.1073/pnas.1802324115

[44] A. Brinckman, K. Chard, N. Gaffney *et al.*, “Computing environments for reproducibility: Capturing the ‘Whole Tale’”, *Future Generation Computer Systems*, vol. 94, p. 854, 2019.