Bridging Concepts and Practice in eScience via Simulation-driven Engineering

Rafael Ferreira da Silva*, Henri Casanova†, Ryan Tanaka†, Frédéric Suter‡
*Information Sciences Institute, University of Southern California, Marina Del Rey, CA, USA
†Information and Computer Sciences, University of Hawaii, Honolulu, HI, USA
‡IN2P3 Computing Center, CNRS, Villeurbanne, France
rafsilva@isi.edu, {henric,ryanyt}@hawaii.edu, frederic.suter@cc.in2p3.fr

***Abstract*—The CyberInfrastructure (CI) has been the object of intensive research and development in the last decade, resulting in a rich set of abstractions and interoperable software implementations that are used in production today for supporting ongoing and breakthrough scientific discoveries. A key challenge is the development of tools and application execution frameworks that are robust in current and emerging CI configurations, and that can anticipate the needs of upcoming CI applications. This paper presents WRENCH, a framework that enables simulation-driven engineering for evaluating and developing CI application execution frameworks. WRENCH provides a set of high-level simulation abstractions that serve as building blocks for developing custom simulators. These abstractions rely on the scalable and accurate simulation models that are provided by the SimGrid simulation framework. Consequently, WRENCH makes it possible to build, with minimum software development effort, simulators that that can accurately and scalably simulate a wide spectrum of large and complex CI scenarios. These simulators can then be used to evaluate and/or compare alternate platform, system, and algorithm designs, so as to drive the development of CI solutions for current and emerging applications.**

***Index Terms*—CyberInfrastrucutre Development, Simulation Accuracy, Reproducible Research, Distributed Computing**

I. Introduction

An impediment to the advancement of CyberInfrastratructure (CI), and of distributed computing in general, is the disconnect between theoretical and practical works [1]. There are occasional success stories in which solid theoretical results provide the foundations for improved, or even near-optimal, practice. However, in the vast majority of the cases, theoreticians produce results that are never used by practitioners, and conversely practitioners use approaches that may be vastly sub-optimal because they are not informed by any theory. One of the reasons for this disconnect is that theoretical work must be done using formally defined models of computation. Ideally, these models are complete enough to be relevant to practice, but simple enough that obtaining theoretical results (e.g., optimality results, complexity bounds) is tractable. As target hardware/software infrastructures and application workloads become more complex, the assumptions embedded in theoretical models often break down in practical settings. These assumptions pertain to the nature of the infrastructure and the application workload, but also to the knowledge available thereof. For instance, a theoretical framework could assume a fully-connected network with no contention and complete and accurate knowledge about the size of intermediate data files produced by an application, while in practice the network has a topology that is subject to contention and only coarse statistical characterizations of data file sizes are known. Accounting for network contention and/or stochastic data file sizes could, however, make the development of theoretical results that would be useful in practice much more challenging, if not intractable.

The above leads to a situation in which theoreticians may say "accounting for these details makes our work impossible" and practitioners may say "if your work does not account for these details then we cannot use it." It is not realistic to expect that theoreticians will somehow become able to obtain solid results when working with drastically more complex models in which all assumptions are realistic. Conversely, it is not realistic to expect that practitioners will use theoretical results hoping that these results might prove useful in spite of unrealistic assumptions. The latter is because incorporating theoretical results in software infrastructures often requires large software design/development efforts. Yet, it is not the case that theoretical work is useful only if all the underlying assumptions about the hardware/software infrastructure and application workloads are completely realistic. Unrealistic assumptions will likely cause the effectiveness of theoretical results to degrade when used in practice. However, the magnitude of these degradations may be small enough to still warrant the use of theoretical results. Or, their magnitude may be so large that new theoretical approaches are needed if practice is to be impacted. We claim that in order to bridge the theory/practice disconnect these degradations must be quantified systematically while developing theoretical results.

In this paper, we discuss our experience using simulation-driven engineering (see Figure 1) for bridging theoretical and practical aspects of CyberInfrastratructure (CI) development. To this end, we introduce the WRENCH project [2], [3], a CI simulation framework that provides high-level simulation abstractions for building accurate and scalable full-fledged simulators of CI scenarios with minimal software development efforts. We then briefly describe two use cases in which WRENCH was used for implementing simulators of production application execution frameworks. These uses cases

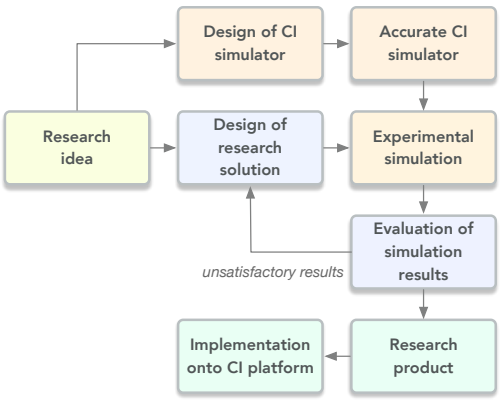

Fig. 1. Simulation-driven engineering life cycle.

demonstrate the ease-of-development, accuracy, and scalability of simulators developed with WRENCH. They also demonstrate how simulation makes it possible to explore emerging CI capabilities. Finally, we also explain how WRENCH supports innovative avenues for CI education.

II. Motivation and Background

Simulation-driven CI engineering is preconditioned on the ability to simulate CI execution scenarios scalably (i.e., running quickly on a standard laptop computer or small server) and accurately (i.e., representative of real-world workload executions on real-world deployments). To this end, we build on the existing SimGrid simulation framework [4], [5]. SimGrid is open source, freely available, has been stable for many years, is actively developed, has a large user community, and has been used successfully in many distributed computing domains (cluster, peer-to-peer, grid, cloud, volunteer computing, etc.). Importantly, SimGrid has also been the object of thorough invalidation and validation studies [6], and its simulation models have been shown to provide large accuracy and/or scalability advantages over competing simulation frameworks [4]. As a result, accurate SimGrid simulations execute in seconds or minutes on a standard laptop computer.

One significant drawback of SimGrid, especially for full-fledged CI simulations, is that its simulation abstractions are too low-level. This drawback was discussed in [7], in which the authors conduct a critical analysis of the state-of-the-art distributed computing simulation frameworks. They observe that many popular simulation frameworks employ inaccurate simulation models, which had already been reported in [4], and acknowledge that SimGrid does provide accurate and scalable simulation capabilities. However, they also observe that using it to implement a simulator of a complex system, such as CI workloads executed on CI deployments, would be extremely labor-intensive.

Given the above, we need a CI simulation framework built on top of SimGrid, so that simulations can be accurate and scalable, but that exposes convenient, reusable, and high-level simulation abstractions, so that implementing simulators of CI scenarios can be done with minimal software engineering efforts. To fill this gap, the WRENCH project [3] emerged as a framework that provides accurate, scalable, and easy-to-use high-level CI abstractions for building simulators of workload executions on production CI deployments. For instance, in order to simulate the example system depicted in Figure 2, one would have to implement all communication and computing behavior of each process instantiated by the components. In SimGrid, simulating such system is labor-intensive and error-prone, requiring the development of thousands of lines of code. By contrast, WRENCH's philosophy is to expose high-level simulation abstractions to provide reusable building blocks for developing custom simulators with minimum effort. Such abstractions represent implementations of simulated CI services that assemble all software pieces (e.g., processes, communication, etc.) necessary for mimicking services behaviors. It is thus straightforward for CI developers to build WRENCH simulators for driving the development of more efficient system designs for CI software, and for CI researchers to build WRENCH simulators for driving the development of efficient algorithms that are at the core of such CI software.

III. The WRENCH Simulation Framework

WRENCH is an open-source C++ library for developing CyberInfrastructure simulators. Its set of core services are composed of two layers as follows:

- *Simulation Core:* All necessary simulation models and base abstractions (computing, communicating, storing), provided by SimGrid.
- *Simulated Core CI Services:* Abstractions for simulated CI components that can be used by an application execution framework to execute computational workloads (compute services, storage services, network proximity services, data location services, etc.).

The Simulated Core CI services layer provides high-level simulation abstractions of commonplace CI services. WRENCH currently provides the following set of simulated CI services (Version 1.4 was released in April 2019):

- *Compute Services:* These services provide mechanisms for executing application tasks, which entail I/O and computation. These include bare-metal servers, cloud platforms, virtualized cluster platforms, and batch-scheduled clusters.
- *Storage Services:* These services provide mechanisms to store application files, which can then be accessed in reading/writing by the compute services when executing tasks that read/write files.
- *File Registry Services:* These services, often known as replica catalogs, are simply databases of key-value pairs of the storage services on which replicas of files are available. They are used during application execution to decide where input files for tasks can be acquired.
- *Network Proximity Services:* These services monitor the network and maintain a database of host-to-host network distances. This database can be queried by the CI system to make informed decisions, e.g., to pick from which

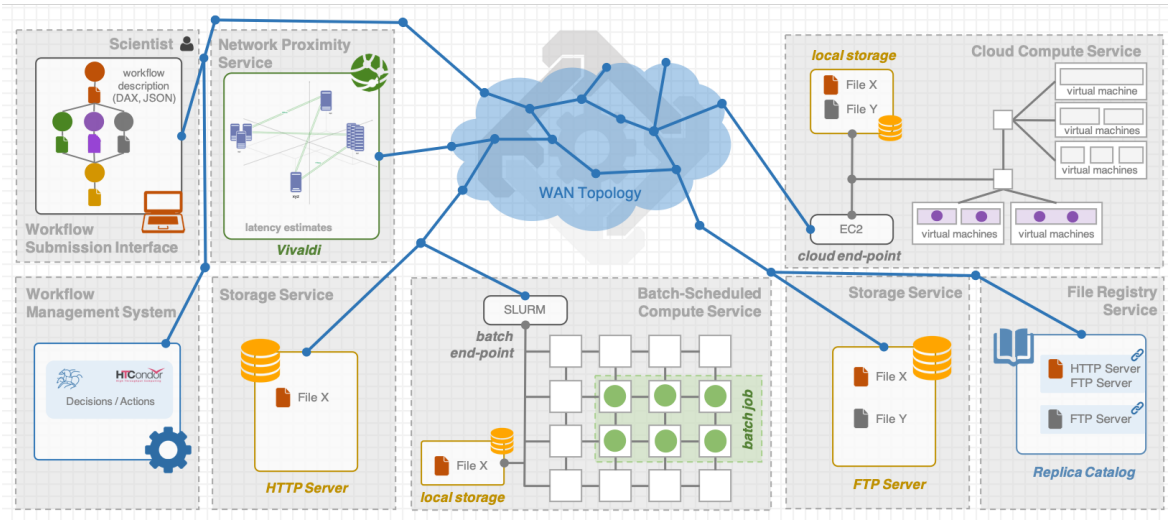

Fig. 2. Example of a CI system that can be simulated using WRENCH. High-level simulation abstractions provided as WRENCH simulated core CI services can be customized and combined to build complex, heterogeneous CI systems.

storage service a file should be retrieved so as to reduce communication time. Typically, network distances are estimated based on round-trip-times between hosts.

- *Workflow Management Systems (WMSs):* A workflow management system provides the mechanisms for executing workflow applications, including decision-making for optimizing various objectives (the most common one is to minimize workflow execution time). WRENCH does not provide a WMS implementation as part of its core components. Instead it provides abstractions for building custom WMS components.

WRENCH service implementations are all parameterizable (e.g., underlying algorithms, various overheads, message payloads, etc.), which allows developers to calibrate the instantiation of the simulated services so that they behave similarly to actual CI services of interest. The ability to define such parameterizable services is key for developing accurate CI simulators, from which research products evaluated via experimental simulation could be seamlessly integrated into actual CI platforms (see Figure 1). All aforementioned services and capabilities are made available through the WRENCH *Developer* API. This API is designed to ease the development of complex (simulated) systems comprised of distributed components that interact both synchronously and asynchronously, such as current and emerging CI systems.

In addition to the Developer's API, WRENCH also provides a *User* API, which makes it possible to build a full-fledged simulator with only a few lines of code. This API provides the mechanisms to control simulation execution (platform configuration, services instantiations, simulation launching, etc.) and analyze simulation outcomes. For an exhaustive description of the WRENCH architecture, its functionalities, and APIs, we refer the reader to the latest WRENCH research article [2] and the project's online documentation [3].

IV. WRENCH's Impact on CI Research, Development, and Education

Although WRENCH is a young project and its first stable release was made available just over one year ago, the framework has already impacted CI research, development, and education. In the last year, we have delivered four WRENCH's stable releases, which were downloaded over 300 times, either directly from our GitHub repository[1] or via our Docker Hub repository[2].

WRENCH has already enabled the development of two simulators of two production application execution frameworks [8], [9] for supporting the study of energy-efficiency for I/O-intensive workflow applications [10], and the study of simulation accuracy and scalability [2]. WRENCH has also gained overseas attention, and received contributions from users in France and Ivory Coast, who are conducting research on workflow scheduling. In the context of CI development, a WRENCH-enabled Pegasus simulator is being used to evaluate novel algorithms for energy-efficiency, workflow data footprint constraints, and cost estimation for running large-scale workflows on heterogeneous computing platforms. In the context of education, we have implemented a small set of simulation-driven pedagogic modules. The modules are available on-line[3] and provide students with simulation-driven interactive learning opportunities. They target essential parallel and distributed computing learning objectives in the specific context of scientific workflow applications running on bare-metal hardware resources.

A. Simulation Accuracy and Scalability

Two major concerns regarding simulation are *accuracy* (the ability to capture the behavior of a real-world system with as

[1] https://github.com/wrench-project/wrench
[2] https://hub.docker.com/r/wrenchproject/wrench
[3] http://wrench-project.org/wrench-pedagogic-modules

little bias as possible) and *scalability* (the ability to simulate large systems with as few CPU cycles and bytes of RAM as possible). By leveraging SimGrid's accurate models [6] and their scalable implementations, WRENCH simulators can yield nearly identical behaviors when compared to actual CI systems, provided that one instantiates correctly the configurable parameters of the simulation models, a process typically referred to as "simulation calibration." The current practice of simulation calibration in our field, based on what is reported in the literature, is ad-hoc, labor-intensive, and thus often poorly realized and documented (e.g., networking research [6], [11]). Automating such process is challenging and still an open question. As part of the WRENCH project, we are studying mechanisms to develop a solid automated calibration approach.

We have recently implemented realistic simulators of two state-of-the-art application execution frameworks, Pegasus [12] and WorkQueue [13]. We have calibrated simulation parameters manually by analyzing and comparing simulated and real-world execution event traces. In addition to regular computing tasks, the WRENCH-enabled Pegasus simulator also accounts for "auxiliary" tasks added by the workflow system, i.e. data stage in/out, cleanup tasks, and data registration.

Figure 3 shows real-world and simulated ECDFs for sample runs of two real-world scientific applications with task completion date ECDFs. We observe that the simulated ECDFs ("wrench") track the real-world ECDFs ("pegasus" and "workqueue") closely. Therefore, we argue that research products prototyped in the simulators (e.g., envisioned enhancements to a target CI systems) that lead to good results in simulation will also, when implemented back into actual software (e.g., the actual target CI systems), lead to good results in practice. Although our simulation results yield good accuracy, we underline that such accuracy may decrease when simulating a different platforms, unless a recalibration of the simulator is performed. However, we would expect that many components of the simulation would not need recalibration (e.g., those simulation parameters that correspond to the behavior/implementation of the CI software being simulated, such as message sizes, timeout threshold, various delay values).

Detailed evaluation of WRENCH-enabled simulator's accuracy can be found in [2]. In that study, we have also evaluated simulation scalability. For this purpose, we have conducted simulation runs with workloads comprising 10K+ jobs running in a simulated cloud platform. Experimental results show that such large-scale, complex simulations ran under 13 minutes on a standard laptop computer. As a result, large numbers of large-scale simulations can be executed so as to identify performance, scalability, and correctness issues in the research products before production-level efforts and resources are committed to the implementation of the research product in actual production software.

All these simulators were implemented using often less 700 lines of code (including code for parsing simulator-specific configuration files), whereas if using SimGrid directly

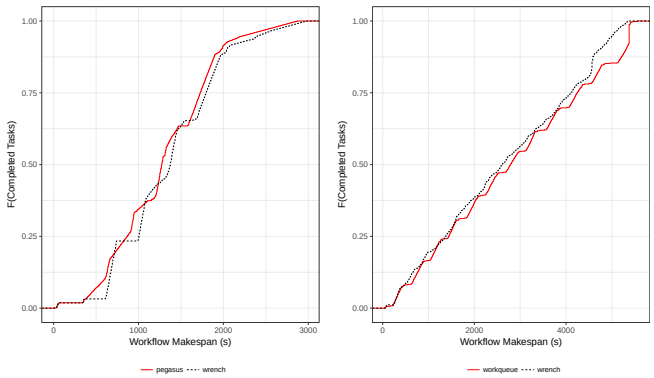

Fig. 3. Empirical cumulative distribution function of task completion times for sample real-world ("pegasus" and "workqueue") and simulated ("wrench") executions.

it would require thousands of lines of code. For example, in [14] a specific simulator of a simple fork-join workflow was implemented directly on top of SimGrid for studying the impact of file transfers on performance. This simulator consists of more than a thousand lines of code and only offers a minuscule fraction of the simulation capabilities provided by WRENCH. Most of the simulator's code pertains to the implementation of control message exchanges between the different processes involved in the simulated system and to the different interaction loops of these processes. Implementing these message exchanges and interactions is a tedious and error-prone process when using the low-level SimGrid API (and very few users would commit to the required effort). By contrast, WRENCH completely hides the implementation of these necessary aspects of a distributed system simulator, which are implemented inside the WRENCH core. Instead, it exposes to users a higher level and user-friendly interface.

B. Energy-aware Computing

In a recent work [10], we have leveraged the WRENCH-enabled Pegasus simulator to investigate the impact of resource utilization and I/O operations on the energy usage, as well as the impact of executing multiple tasks concurrently on multi-socket, multi-core compute nodes. Our simulator allowed us to draw direct comparisons between real-world and modeled power and energy consumption. Figure 4 shows the simulated power and energy consumption measurements as well as with the traditional model (from the literature). We find that our model has high accuracy when compared to real-world executions. Furthermore, our model improves accuracy by about two orders of magnitude when compared to the traditional models used in the energy-efficient scheduling literature.

C. Pedagogic Modules

To support simulation-driven computing education, we have implemented a few simulation-driven pedagogic modules supported by WRENCH-based simulators. All these simulators were implemented using often less than one hundred lines of code. Modules are packaged as Docker containers that expose

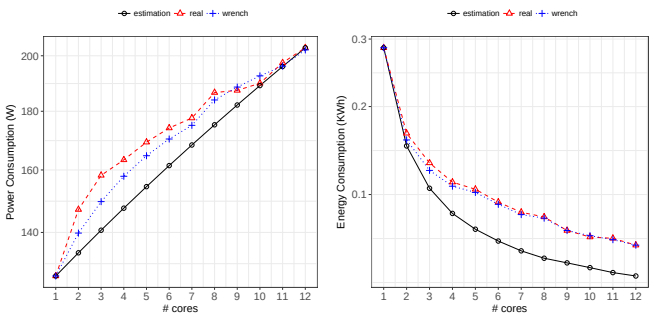

Fig. 4. Comparison of power (left) and energy (right) consumption measurements for a real-world application ("real") using a well-known model from the literature ("estimation") and our WRENCH model ("wrench").

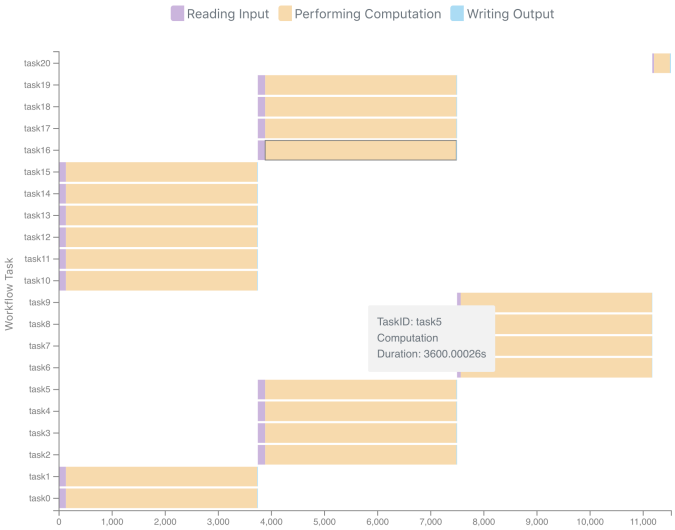

Fig. 5. Gantt chart of simulated workload task executions for a scientific workflow running on a distributed platform.

an interactive Web application through which students can perform various hands-on activities. Each activity entails running, through the Web application, a simulator with different input parameters. Students run these simulations to explore different execution options for different application and/or platform configuration scenarios. Students are presented with simulation output that includes visualizations of the simulated execution, such as resource utilization metrics, event timelines, and interactive Gantt charts (see Figure 5). Students must analyze this output and draw a range of conclusions that lead them to achieve a well-defined set of learning objectives. These modules have already been used in the classroom, in an undergraduate course at the University of Hawai'i in Spring 2019 [15].

V. Conclusion

In this paper, we have introduced WRENCH, a framework for enabling simulation-driven engineering for evaluating and developing CI application execution frameworks. We have reported on how WRENCH has aided to bridge concepts and practice in eScience via simulation. We have described the overall framework capabilities and available services, and briefly exposed WRENCH's impact on (*i*) research and development, through the implementation of two state-of-the-art application execution frameworks and a case study on energy-efficiency; and (*ii*) education, via the development of online self-contained pedagogic modules.

Although relying on simulation is key for advancing state-of-the-art CI software, attempting to do so from scratch, i.e., implementing a full-fledged simulation based on base models of the infrastructure and application workloads is not worthwhile. It is too labor-intensive and in practice results in the implementer opting for simplifications that translate in loss of simulation accuracy. Instead, relying on simulation frameworks, such as SimGrid and WRENCH, allows users to focus on their own research, development, and/or teaching interests while keeping the gap between concepts and practice as narrow as possible.

An interesting future direction is that of automated simulation calibration. The "simulation calibration" question is crucial to establishing a solid simulation-driven experimental science approach for CI research, development, and education. Yet this question is poorly understood and current calibration practices are both inadequate and labor-intensive. This issue is not confined to WRENCH, but is faced by all distributed system simulators. An additional challenge is that of the data necessary for performing calibration is often not available to the community—e.g., execution and performance traces are scarce and fragmented.

Acknowledgments

This work is funded by NSF contracts #1642369 and #1642335; by CNRS under grant #PICS07239; and partly funded by NSF contracts #1923539 and #1923621: "Cyber-Training: Implementation: Small: Integrating core CI literacy and skills into university curricula via simulation-driven activities". We thank Martin Quinson, Arnaud Legrand, and Pierre-François Dutot for their valuable help.

References

[1] M. J. Brooker, "The space between theory and practice in distributed systems," https://brooker.co.za/blog/2014/08/10/the-space-between.html, 2014.

[2] H. Casanova, S. Pandey, J. Oeth, R. Tanaka, F. Suter, and R. Ferreira da Silva, "WRENCH: A Framework for Simulating Workflow Management Systems," in *13th Workshop on Workflows in Support of Large-Scale Science (WORKS'18)*, 2018, pp. 74–85.

[3] "WRENCH Project," https://wrench-project.org, 2019.

[4] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, 2014.

[5] "The SimGrid Project," https://simgrid.org, 2019.

[6] P. Velho, L. M. Schnorr, H. Casanova, and A. Legrand, "On the validity of flow-level TCP network models for grid and cloud simulations," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 23, no. 4, p. 23, 2013.

[7] G. Kecskemeti, S. Ostermann, and R. Prodan, "Fostering energy-awareness in simulations behind scientific workflow management systems," in *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*. IEEE, 2014, pp. 29–38.

[8] "The WRENCH Pegasus Simulator," https://github.com/wrench-project/pegasus, 2019.

[9] "The WRENCH WorkQueue Simulator," https : / / github . com / wrench-project/workqueue, 2019.

[10] R. Ferreira da Silva, A.-C. Orgerie, H. Casanova, R. Tanaka, E. Deelman, and F. Suter, "Accurately Simulating Energy Consumption of I/O-intensive Scientific Workflows," in *2019 International Conference on Computational Science (ICCS)*, 2019.

[11] T. R. Andel and A. Yasinac, "On the credibility of manet simulations," *Computer*, no. 7, pp. 48–54, 2006.

[12] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira da Silva, M. Livny, and K. Wenger, "Pegasus, a workflow management system for science automation," *Future Generation Computer Systems*, vol. 46, pp. 17–35, 2015.

[13] L. Yu, C. Moretti, A. Thrasher, S. Emrich, K. Judd, and D. Thain, "Harnessing parallelism in multicore clusters with the all-pairs, wavefront, and makeflow abstractions," *Cluster Computing*, vol. 13, no. 3, pp. 243–256, 2010.

[14] A. Chai, M.-M. Bazm, S. Camarasu-Pop, T. Glatard, H. Benoit-Cattin, and F. Sutern, "Modeling distributed platforms from application traces for realistic file transfer simulation," in *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE Press, 2017, pp. 54–63.

[15] R. Tanaka, R. Ferreira da Silva, and H. Casanova, "Teaching Parallel and Distributed Computing Concepts in Simulation with WRENCH," https://arxiv.org/submit/2766761/view, 2019, submitted for publication.