A Community Roadmap for Scientific Workflows Research and Development

Ferreira da Silva, R.; Casanova, H.; Chard, K.; Altintas, I.; Badia, R.M.; Balis, B.; Coleman, T.; Coppens, F.; Di Natale, F.; Enders, B.; Fahringer, T.; Filgueira, R.; Fursin, G.; Garijo, D.; Goble, C.; Howell, D.; Jha, S.; Katz, D.S.; Laney, D.; Leser, U.; Malawski, M.; Mehta, K.; Pottier, L.; Ozik, J.; Peterson, J.L.; Ramakrishnan, L.; Soiland-Reyes, S.; Thain, D.; Wolf, M.

DOI
10.1109/WORKS54523.2021.00016

Publication date
2021

Document Version
Author accepted manuscript

Published in
2021 IEEE Workshop on Workflows in Support of Large-Scale Science (WORKS 2021)

A Community Roadmap for Scientific Workflows Research and Development

Rafael Ferreira da Silva*†, Henri Casanova‡, Kyle Chard§¶, Ilkay Altintas‖, Rosa M Badia**, Bartosz Balis††, Tainã Coleman†, Frederik Coppens‡‡[x], Frank Di Natale[xi], Bjoern Enders[xxi], Thomas Fahringer[xii], Rosa Filgueira[xiii], Grigori Fursin[xiv], Daniel Garijo[xv], Carole Goble[xvi], Dorran Howell[xvii], Shantenu Jha[xviii], Daniel S. Katz[xix], Daniel Laney[xi], Ulf Leser[xx], Maciej Malawski††, Kshitij Mehta*, Loïc Pottier†, Jonathan Ozik§¶, J. Luc Peterson[xi], Lavanya Ramakrishnan[xxi], Stian Soiland-Reyes[xvi xxii], Douglas Thain[xxiii], Matthew Wolf*

*Oak Ridge National Laboratory, Oak Ridge, TN, USA †University of Southern California, Marina Del Rey, CA, USA
‡University of Hawaii, Honolulu, HI, USA §Argonne National Laboratory, Lemont, IL, USA
¶The University of Chicago, Chicago, IL, USA ‖University of California, San Diego, La Jolla, CA, USA
**Barcelona Supercomputing Center, Spain ††AGH University of Science and Technology, Krakow, Poland
‡‡Ghent University, Ghent, Belgium [x]VIB Center for Plant Systems Biology, Belgium
[xi]Lawrence Livermore National Lab, Livermore, CA, USA [xii]University of Innsbruck, Innsbruck, Austria
[xiii]Heriot-Watt University, Edinburgh, UK [xiv]OctoML, USA [xv]Universidad Politécnica de Madrid, Spain
[xvi]The University of Manchester, Manchester, UK [xvii]Tweag, Zürich, Switzerland
[xviii] Brookhaven National Laboratory, Upton, NY, 11973 [xix]University of Illinois at Urbana-Champaign, USA
[xx]Humboldt-Universität zu Berlin, Berlin, Germany [xxi]Lawrence Berkeley National Lab, Berkeley, CA, USA
[xxii]University of Amsterdam, Amsterdam, The Netherlands [xxiii]University of Notre Dame, Indiana, USA

***Abstract*—The landscape of workflow systems for scientific applications is notoriously convoluted with hundreds of seemingly equivalent workflow systems, many isolated research claims, and a steep learning curve. To address some of these challenges and lay the groundwork for transforming workflows research and development, the WorkflowsRI and ExaWorks projects partnered to bring the international workflows community together. This paper reports on discussions and findings from two virtual "Workflows Community Summits" (January and April, 2021). The overarching goals of these workshops were to develop a view of the state of the art, identify crucial research challenges in the workflows community, articulate a vision for potential community efforts, and discuss technical approaches for realizing this vision. To this end, participants identified six broad themes: FAIR computational workflows; AI workflows; exascale challenges; APIs, interoperability, reuse, and standards; training and education; and building a workflows community. We summarize discussions and recommendations for each of these themes.**

***Index Terms*—Scientific workflows, community roadmap, data management, AI workflows, exascale computing, interoperability**

I. Introduction

Scientific workflow systems are used almost universally across scientific domains for solving complex and large-scale computing and data analysis problems, and have underpinned some of the most significant discoveries of the past decades [1]. Many of these workflows have significant computational, storage, and communication demands, and thus must execute on a wide range of large-scale platforms, from local clusters over science or public clouds to upcoming exascale HPC platforms [2]. Managing these executions is often a significant undertaking, requiring a sophisticated and versatile software infrastructure.

Historically, many of these infrastructures for workflow execution consisted of complex, integrated systems, developed in-house by workflow practitioners with strong dependencies on a range of legacy technologies—even including sets of ad-hoc scripts. Due to the increasing need to support workflows, dedicated workflow systems were developed to provide abstractions for creating, executing, and adapting workflows conveniently and efficiently while ensuring portability. While these efforts are all worthwhile individually, there are now hundreds of independent workflow systems [3]. These are created and used by thousands of researchers and developers, leading to a rapidly growing corpus of workflows research publications. The resulting workflow system technology landscape is fragmented, which may present significant barriers for future workflow users due to the tens of seemingly comparable, yet usually mutually incompatible, systems that exist.

In the current workflow research, there are conflicting theoretical bases and abstractions for what constitutes a workflow system. It may be possible to translate between systems that use the same underlying abstractions; however, the contrary is not feasible. Specifically, typical systems have a layered model that abstractly underlies it: (i) if the models are the same for two systems, they are compatible to some extent, and if they implement the same layers, they can be interchanged (modulo some translation effort); (ii) if the models are the same for two systems, but they are implemented by components at different layers, they can be complementary, and may have common elements that could be shared; (iii) if the models are distinct, workflows or system components are likely not exchangeable or interoperable. As a result, many teams still elect to build their own custom solutions rather than adopt,

TABLE I
SUMMARY OF CURRENT WORKFLOWS RESEARCH AND DEVELOPMENT CHALLENGES AND PROPOSED COMMUNITY ACTIVITIES.

Theme	Challenges	Community Activities
FAIR Computational Workflows	• Define FAIR principles for computational workflows that consider the complex lifecycle from specification to execution and data products • Define metrics to measure the FAIRness of a workflow. • Engage the community to define principles, policies, and best practices	• Review prior and current efforts for FAIR data and software with respect to workflows, and outline rules for FAIR workflows • Define recommendations for FAIR workflow developers and systems • Automate FAIRness in workflows by recording necessary provenance data
AI Workflows	• Lack of support for heterogeneity of compute resources and fine-grained data management features, versioning, and data provenance capabilities • Lack of capabilities for enabling workflow steering and dynamic workflows • Integration of ML frameworks into the current HPC landscape	• Develop comprehensive use cases for sample problems with representative workflow structures and data types • Define a process for characterizing the challenges for enabling AI workflows • Develop AI workflows as a way to benchmark HPC systems
Exascale Challenges and Beyond	• Resource allocation policies and schedulers are not designed for workflow-aware abstractions, thus users tend to use an ill-fitted job abstraction • Unfavorable design of resource descriptions and mechanisms for workflow users/systems, and lack of fault-tolerance and fault-recovery solutions	• Develop documentation in the form of workflow templates/recipes/miniapps for execution on high-end HPC systems • Specify benchmark workflows for exascale execution • Include workflow requirements as part of the machine procurement process
APIs, Reuse, Interoperability, and Standards	• Workflow systems differ by design, thus interoperability at some layers is likely to be more impactful than others • Workflow standards are typically developed by a subset of the community • Quantifying the value of common representations of workflows is not trivial	• Identify differences and commonalities between different systems • Identify and characterize domain-specific efforts, identify workflow patterns, and develop case-studies of business process workflows and serverless workflow systems
Training and Education	• Many workflow systems have high barrier to entry and lack training material • Homegrown workflow solutions and constraints can prevent users from reproducing their functionality on workflow tools developed by others • Unawareness of the workflow technological and conceptual landscape	• Identify basic sample workflow patterns, develop a community workflow knowledge-base, and look at current research on technology adoption • Include workflow terminology and concepts in university curricula and software carpentry efforts
Building a Workflows Community	• Define what is meant by a "workflows community" • Remedy the inability to link developers and users to bridge translational gaps • Pathways for participation in a network of researchers, developers, and users	• Establish a common knowledge-base for workflow technology • Establish a *Workflow Guild*: an organization focused on interaction and good relationships and self-support between workflow developers and their systems

adapt, or build upon, existing workflow systems. This current state of the workflow systems landscape negatively impacts workflow users, developers, and researchers [4].

The WorkflowsRI [5] and ExaWorks [6] projects have partnered to bring the workflows community (researchers, developers, science and engineering users, and cyberinfrastructure experts) together to collaboratively elucidate the R&D efforts necessary to remedy the above situation. They conducted a series of virtual events entitled "Workflows Community Summits", in which the overarching goal was to (i) develop a view of the state of the art, (ii) identify key research challenges, (iii) articulate a vision for potential activities, and (iv) explore technical approaches for realizing (part of) this vision. The summits gathered over 70 participants from a group of international lead researchers and developers, from distinct workflow systems and user communities. The outcomes of the summits have been compiled and published in two technical reports [7], [8]. In this paper, we summarize the discussions and findings by presenting a consolidated view of the state of the art, challenges, and potential efforts, which we eventually synthesize into a community roadmap. Table I presents, in the form of top-level themes, a summary of those challenges and targeted community activities. Table II summarizes a proposed community roadmap with technical approaches.

The remainder of this paper is organized as follows. Sections II-VII provide a brief state of the art and challenges for each theme and proposed community activities. Section VIII discusses technical approaches for a community roadmap. Section IX concludes with a summary of discussions.

II. FAIR COMPUTATIONAL WORKFLOWS

The FAIR principles [9] have laid a foundation for sharing and publishing digital assets and, in particular, data. The FAIR principles emphasize machine accessibility and that all digital assets should be Findable, Accessible, Interoperable, and Reusable. Workflows encode the methods by which the scientific process is conducted and via which data are created. It is thus important that workflows both support the creation of FAIR data and themselves adhere to the FAIR principles.

A. Brief State-of-the-art and Challenges

Workflows are hybrid processual digital assets that can be considered as data or software, or some combination of both. As such, there is a range of considerations to take into account with respect to the FAIR principles [10]. Some perspectives are already well explored in data/software FAIRness, such as descriptive metadata, software metrics, and versioning; however, workflows create unique challenges such as representing a ***complex lifecycle*** from specification to execution via a workflow system, through to the data created at the completion of the workflow execution.

As a specialized kind of software, workflows have two properties that FAIRness fundamentally must address: ***abstraction and composition***. As far as possible a workflow specification, as a graph or some declarative expression, is abstracted from its execution undertaken by a dedicated software platform. Workflows are composed of modular building blocks and expected to be remixed. FAIR applies "all the way down" at the specification and execution level, and for the whole workflow and each of its components. One of the most challenging aspects of making workflows FAIR is ensuring that they can be ***reused***. These challenges include being able to capture and then move workflow components, dependencies, and application environments in such a way as not to affect the resulting execution of the workflow. Further work is required to understand use cases for reuse, before exploring methods for capturing necessary context and enabling reuse in the same or different environments.

TABLE II
SUMMARY OF TECHNICAL ROADMAP MILESTONES PER RESEARCH AND DEVELOPMENT THRUST.

Thrust	Roadmap Milestones
Definition of common workflow patterns and benchmarks	• Define small sets of workflow pattern and benchmark deliverables, and implement them using a selected set of workflow systems • Investigate automatic generation of patterns and configurable benchmarks (to enable weak and strong scaling experiments) • Establish or leverage a centralized repository to host and curate patterns and benchmarks
Identifying paths toward interoperability of workflow systems	• Define interoperability for different roles, develop a horizontal interoperability (i.e., making interoperable components), and establish a requirements document per abstraction layer • Develop real-world workflow benchmarks, use cases for interoperability, and common APIs that represent workflow library components • Establish a workflow systems developer community
Improving workflow systems' interface with legacy and emerging HPC software and hardware stacks	• Document a machine-readable description of key properties of widely used sites, and remote authentication needs from the workflow perspective • Identify new workflow patterns (e.g. motivated from AI workflows), attain portability across heterogeneous hardware, and develop a registry of execution environment information • Organize a community event involving workflow system developers, end users, authentication technology providers, and facility operators

Once use cases are defined, there are many ***metrics and features*** that could be considered to determine whether a workflow is FAIR. These features may differ depending on the type of workflow and its application domain. Prior work in data and software FAIRness [9], [11] provides a starting point, however, these metrics need to be revised for workflows. In terms of labeling, there has been widespread adoption of reproducibility badges for publications and of FAIR labels for data in repositories [12]. Similar approaches could be applied to computational workflows. Finally, developing methods for FAIR workflows requires ***community engagement*** (i) to define principles, policies, and best practices to share workflows; (ii) to standardize metadata representation and collection processes; (iii) to create developer-friendly guidelines and workflow-friendly tools; and (iv) to develop shared infrastructure for enabling development, execution, and sharing of FAIR workflows.

B. A Vision for Potential Community Activities

Given current efforts for developing FAIR data and software, it is important to first understand what efforts could be adapted to workflow problems. An immediate activity include participating in working groups focused on applying FAIR principles to data and software. For instance, FAIR4RS [13] coordinates community-led discussions around FAIR principles for research software. Workflows could then be initially tackled from the point of view of workflows as software, which could originate a novel ***task force***. Proposed working groups such as FAIR for Virtual Research Environments [14] represent adequate progress towards this goal.

A fundamental tenet of FAIR is the universal availability of machine processable metadata. The European EOSC-Life Workflow Collaboratory, for example, has developed a metadata framework for FAIR workflows based on schema.org [15], RO-Crate [16], and CWL [17]. This could be a community starting point for standardization of metadata about workflows.

An integral aspect of a FAIR computational workflows task force would be to collect a set of real-world use cases and workflows in several domains to examine from the perspective of the FAIR data principles. This exercise will likely highlight areas in which the FAIR data principles adequately represent challenges in workflows. Based on these experiences, a set of ***simple rules*** could be defined for creating FAIR workflows, similar to the ones in [18]. From these rules, prominent workflow repositories (e.g., WorkflowHub.eu [19] and Dockstore [20]), communities, and workflow systems can define ***recommendations*** to support the development and sharing of FAIR workflows. These efforts relate not only to the workflows themselves, but the workflow components, execution environments, and the different types of data.

Ensuring ***provenance*** can capture the necessary information is key for enabling FAIRness in workflows. Many provenance models [21] can be implemented or extended to capture the information needed for FAIR workflows. Additionally, FAIR principles are more likely to be followed if the process for capturing these metrics is automated and embedded in workflow systems. In this case, a workflow execution will become FAIR by default, or perhaps with minimal user curation.

III. AI WORKFLOWS

Artificial intelligence (AI) and machine learning (ML) techniques are becoming more and more popular within the scientific community. Workflows increasingly integrate ML models to guide analysis, couple simulation and data analysis codes, and exploit specialized computing hardware (e.g., GPUs, neuromorphic chips) [22]. These workflows inherently couple various types of tasks such as short ML inference, multi-node simulations, long-running ML model training, etc. They are also often iterative and dynamic, with learning systems deciding in real time how to modify the workflow, e.g. by adding new simulations or changing the workflow all together. AI-enabled workflow systems therefore must be capable of optimally placing and managing single- and multi-node tasks, exploit heterogeneous architectures (CPUs, GPUs, and accelerators), and seamlessly coordinate the dynamic coupling of disparate simulation tools.

A. Brief State-of-the-art and Challenges

Workflows empowered with ML techniques largely differ from traditional workflows running on HPC machines. While workflows (i.e., one large-scale application simulating a scientific object or process) traditionally take little input data and produce large outputs, ML approaches target model training, which usually requires the input of a large quantity of data (either via files or from a collection of databases) and produces a small number of trained models. After training, these models

are used to infer new quantities (during "model inference") and behave like very lightweight applications that produce small quantities of output data. These models can be standalone applications, or even embedded within larger traditional simulations. There exists an inherent tension between traditional HPC, which evolved around executing large capacity-style codes and AI-HPC, which requires the coordinated execution of many smaller capability-scale applications (e.g., large ensembles of data generation co-mingled with inference and coinciding with periodic retraining of models).

With its reliance of data, effective AI-workflows should provide ***fine-grained data management*** and versioning features, as well as adequate data provenance capabilities. This data management will have to be flexible: some applications and workflows might need to move data via a file-system, while others could be better served from a traditional database, data store, or a streaming dataflow model. During inference, it may be best to couple the (lightweight) model as close to the data it is processing as possible. In any case, effective data management is a key feature of successful AI workflows.

Another key feature of AI workflows is the inherent incorporation of non-traditional hardware, such as GPUs and tensor processing units (TPUs), which can significantly accelerate both training and inference steps. Workflow systems thus need to provide mechanisms for managing ***heterogeneous resources*** – offloading heavy computations to GPUs, and managing data between GPU and CPU memory hierarchies. Furthermore, since ML training and inference may be best executed on different hardware from the main simulation, AI workflows might need to be executed on **multi-machine** federated systems (with the main code executed on a traditional HPC system, but the ML model training or inference on a separate system). Additionally, it is also necessary to provide tight ***integration to widely-used ML frameworks***, the development of which is not driven by the HPC community. ML frameworks use Python and R-based libraries and do not follow the classic HPC model: C/C++/Fortran, MPI, and OpenMP, and submission to an HPC batch scheduler. Yet, some efforts in the HPC community seem promising, e.g. LBANN [23], EMEWS [24], [25], and eFlows4HPC [26]. Other approaches like Merlin [27] blend HPC and cloud technologies to enable federated workflows. However, there is a clear disconnect between HPC motivations, needs, and requirements, and AI/ML current practices.

Finally, one of the major differences between traditional and AI workflows is the inherent ***iterative nature of ML processes*** – AI workflows often feature feedback loops over a data set. Data are created, the model is retrained, and its accuracy evaluated. ML training tasks might leverage hyperparameter optimization frameworks [28]–[30] to adjust their execution settings in real time. The final trained model is often used to select new data to acquire (in an "active learning" environment, the model is used to decide which new simulations to run to better train the model on the next iteration). By design, ML-empowered workflows are dynamic, in contrast to traditional workflows with more structured and deterministic computations. At runtime, the workflow execution graph can potentially evolve based on internal metrics (accuracy), which may reshape the graph or trigger task preemption. Workflow systems should thus support ***dynamic branching*** (e.g., conditionals, criteria) and partial workflow re-execution on-demand.

B. A Vision for Potential Community Activities

To address the disconnect between HPC systems and practices and AI workflows, the community needs to develop sets of example ***use cases for sample problems*** with representative workflow structures and data types. In addition to expanding upon the above challenges, the community could "codify" these challenges in example use cases. However, the set of challenges for enabling AI workflows is extensive. The community thus needs to define a ***systematic process*** for identifying and categorizing these challenges. A short-term recommendation would be to write a "community white paper" about AI Workflow challenges/needs.

Building from the use cases above for the needs and requirements of AI workflows, the community could define ***AI-Workflow mini-apps***, which could be used to pair with vendors/future HPC developers so that the systems can be benchmarked against these workflows, and therefore support the co-design of emerging or future systems (e.g., MLCommons [31] and the Collective Knowledge framework [32]).

IV. Exascale Challenges and Beyond

Given the computational demands of many workflows, it is crucial that their execution be not only feasible but also effortless and efficient on large-scale HPC systems, and in particular upcoming exascale systems [2]. Exascale systems are likely to contain millions of independent computing elements that can be concurrently scheduled across more than 10,000 nodes, millions of cores, and tens of thousands of accelerators.

A. Brief State-of-the-art and Challenges

HPC resource allocation policies and schedulers designs typically do not consider workflow applications: they provide a mere "job" abstraction instead of workflow-aware abstractions. Workflow users/systems are forced to make their workflows run on top of this ***ill-fitted abstraction*** – e.g., it is difficult to control low-level behavior critical to workflows (i.e., precise mapping of tasks to specific compute resources on a compute node). Furthermore, there is a clear lack of support for elasticity (i.e., scaling up/down the number of nodes). Overall, it is currently difficult to run workflows efficiently and conveniently on HPC systems without extending (or even overhauling) resource management/scheduling approaches, which ideally would allow programmable, fine-grain application-level resource allocation and scheduling.

Related to the above challenge, it is currently not possible to support both workflow and non-workflow users harmoniously and/or efficiently on the same system. Some features needed by workflows are often unavailable. For instance, batch schedulers can support elastic jobs (e.g., Slurm); however, experience shows that system admins may not be keen on

enabling this capability, as they deem long static allocations preferable. A ***cultural change*** is perhaps needed as it seems that workflows are not yet considered as high-priority applications by high-end compute facilities.

Hybrid architectures are key to high performance and many workflows can or are specifically designed to exploit them. However, on HPC systems, the necessary ***resource descriptions and mechanisms*** are not necessarily available to workflow users/systems (even though some workflow systems have successfully interfaced to such mechanisms on particular systems) [33]. Although these resource descriptions and mechanisms are typically available as part of the "job" abstraction, it is often not clear how a workflow system can discover and use them effectively.

Finally, ***fault-tolerance and fault-recovery*** have been extensively studied on exascale systems, with several works and working solutions for traditional parallel jobs [34]. In the context of scientific workflows, specific techniques have been the subject of several studies [35], however workflow-specific solutions are typically not readily available or deployed. Moreover, workflows are built on smaller platforms, thus operating and testing at exascale would entail expressing new requirements/capabilities and dealing with new constraints (e.g., what is a "local" exascale workflow?).

B. A Vision for Potential Community Activities

An immediate activity consists in developing documentation in the form of ***workflow templates/recipes/miniapps*** for execution on high-end HPC systems to be hosted on a community web site. Some efforts underway provide partial solutions [36]. For instance, collections of workflows exist but typically do not provide large scale execution capabilities (e.g., community testbeds). Some compute facilities provide workflow tool documentation and help with their users [37]. These solutions should be cataloged as a starting point, and HPC facilities could promote yearly "workflow days", in which they give workflow users and developers training and early access to machines to try out their workflows, thus gathering feedback from users and developers.

To drive the design of workflow-aware abstractions, the community could specify ***community benchmark workflows*** for exascale execution, exerting all relevant hardware as well as functionality capabilities. Then it becomes possible for different workflow systems to execute these benchmarks – initial efforts could build on previous benchmark solutions [38], [39]. These benchmarks could then be included in ***exascale machines acceptability tests***. Note that there will be a need to pick particular workflow systems to run these benchmarks, which will foster training and education of HPC personnel.

Last, including workflow requirements very early on in ***machine procurement process*** for machines at computing facilities will significantly lower the barriers for enabling workflow execution and therefore porting workflow applications. This effort is therefore preconditioned on the availability of miniapps and/or benchmark specifications, as well as API/scheduler specifications, as outlined above.

V. APIs, Interoperability, Reuse, and Standards

There has been an explosion of workflow technologies in the last decade [3]. Individual workflow systems often serve a particular user community, a specific underlying compute infrastructure, a dedicated software engineering vision, or follow a specific historical trait. As a result, there are substantial technical and conceptual overlaps. Reasons for divergence include (i) use cases require different workflow structures, (ii) organizations have very different optimization goals, (iii) predefined execution systems provide fundamentally different capabilities, or (iv) availability and scarcity of different types of resources. Another reason is that it is relatively easy to start building a workflow system for a specific narrow focus (i.e., these systems have a gentle software development curve [40]), leading to large numbers of packages that provide some basic functionality, and developers who are subject to the sunk cost fallacy and then continue to invest in their custom packages, rather than joining forces and building community packages. This divergence leads to missed opportunities for interoperability. It is often difficult for workflows to be ported across systems, for system components to be interchanged, for provenance to be captured and exploited in similar ways, and for developers to leverage different execution engines, schedulers, or monitoring services.

A. Brief State-of-the-art and Challenges

Workflow systems often grow organically: developers start by solving a concrete data analysis problem and they end up with a new workflow tool. In some cases, workflow systems may ***differ by design***, rather than by accident. For example, they offer fundamentally different abstractions or models for a workflow: DAG-structured *vs.* recursive, imperative *vs.* declarative, data flow *vs.* control (and data) flow. These fundamental differences, catering for different use cases, make it such that full interoperability may simply not be possible. Alternately, workflow systems have many different layers and components that may be interchangeable, e.g., workflow specifications, task descriptions, data passing methods, file handling, task execution engines, etc. Interoperability at some layers is likely to be more impactful than others; for instance, being able to run the same workflow specification (with appropriately encapsulated task implementations) on different workflow infrastructures would be a major relief for users trying to reuse implemented workflows in other organizations. Further, interoperability does not need to imply agreement and for workflow systems to implement a standard interface; instead, it may occur via shim layers or intermediate representations, in a similar manner to compiling to a high level language. With the reuse goal, projects as eFlows4HPC proposes the HPC Workflows as a Service (HPCWaaS) methodology, where workflows will be defined by expert developers and provided as a service to community users [26].

Most efforts to unify workflow systems and/or their components have led to the definition of a "standard" developed by a subset of the community [41], [42]. However, the specialization of some of these standards may require that other

systems conform to that specification, thus resulting in low adoption. Attempts to standardize also may lead to overly generic interfaces that in the end inhibit usability and lead to hidden incompatibilities.

A particularly pressing problem at the interface of workflow technology and HPC systems is the need for a ***common submission model*** that is compatible to heterogeneous platforms. The differences between the ways workflow engines, schedulers, and execution engines interact is a universal challenge faced by workflow developers when trying to target multiple infrastructures underlying long-lasting design decisions. Further challenges relate to authentication and authorization models deployed on many systems (e.g., two-factor authentication). Some efforts in this area are currently undergoing [43].

B. A Vision for Potential Community Activities

An immediate and continuous action would be to host several ***"bake-offs"*** to compare workflow systems, including task and workflow definitions, a benchmark set of workflows with defined input data and outputs, as well as job execution interfaces. This would entail engaging participants to write and execute these workflows and identifying commonalities between systems. A successful example is the GA4GH-DREAM challenge [44]. An open question is whether such attempts should be domain-specific or domain-overarching; there is likely a greater opportunity for standardization within domains (and indeed some domains have already made significant progress), but domain-specific standards would only partly solve the interoperability problem. The workflow community should then review these areas, determine and then publicize what has worked, and build on successful prior efforts.

With the emergence of FaaS (Function-as-a-Service) systems (e.g., AWS Lambda and Step Functions, Azure Durable Functions, Google Cloud Functions, IBM Composer), or CaaS (Container-as-a-Service) services (e.g. AWS Fargate, Google Cloud Run), the community should identify a set of ***suggested use cases*** and compare them against an implementation with popular or recently developed FaaS-enabled workflow systems [45]–[47]. Such a comparison may turn out complementary features that can be of benefit for both industry and the workflows community. In addition to features, a set of common workflow patterns could also be identified. However, there is still some uncertainty regarding the scope of previously developed patterns (e.g., for representing patterns in dynamic workflows). Thus, it is necessary to ***survey published patterns*** [48], [49] and identify gaps seen by the community.

Although the above proposed activities have the potential to advance interoperability, the current funding and research recognition models often implicitly work against standardization by constantly requiring innovative ideas even in areas where outreach, uptake, and maintenance rather than innovation seems to be the most pressing problem. Developing ***sustained funding models*** for building and evolving workflow standards, encouraging their adoption, supporting interoperability, testing, and providing user and developer training would help address these challenges.

VI. Training and Education for Workflow Users

There is a strong need for more, better, and new training and education opportunities for workflow users. Many users "re-invent the wheel" without reusing software infrastructures and workflow tools that would make their workflow execution more convenient, more efficient, easier to evolve, and more portable. This is partly due to the lack of comprehensive and intuitive training materials that would guide users through the process of designing a workflow (besides the typical "toy" examples provided in tutorials).

A. Brief State-of-the-art and Challenges

Using workflow tools can require large amounts of effort and time, due to a ***steep learning curve***. A contributing factor is that users may not know the required terminology and concepts. As a result, some have noted that what would be needed in the current technology landscape is to "ship a developer along with the workflow tool".

One of the reasons for the above challenge is that there are ***few "recipes" or "cookbooks"*** for workflow systems. Furthermore, given that workflows and their execution platforms are complex and diverse, in addition to mere training material, there is a need for a training infrastructure that consists of workflows and accompanying data (small enough to be used for training purposes but large enough to be meaningful) as well as execution testbeds for running these workflows.

Given the multitude of workflow systems [3], and the lack of standards (Section V), users cannot easily pick the appropriate systems for their needs. More importantly, there is an understandable fear of being locked into a tool that at some point in the near future will no longer be supported. Although documentation can be a problem, ***guidance*** is the more crucial issue. Many users have the basic skills to create and execute workflows on some system, but as requirements gradually increase many users evolve their simple approaches in ad-hoc ways, thus developing/maintaining a working but ***imperfect homegrown system***. There is thus a high risk of hitting technological or labor-intensiveness roadblocks, which could be remedied by using a workflow system. But, when "graduating" to such a system, there will likely be constraints that prevent users from reproducing the functionality of their homegrown system. The benefits of using the workflow system should thus largely outweigh the drawback of these constraints.

Given all the above challenges, it is not easy to ***reach out to users at the appropriate time***. Reach out too early and users will not view using a particular workflow system as compelling. Reach out too late, and users are already locked into their homegrown system, even though in the long run this system will severely harm their productivity.

B. A Vision for Potential Community Activities

Lowering the entry barrier is key for enabling the next-generation of researchers to benefit from workflow systems. An initial approach would be to provide a basic set of simple, yet conceptually rich, ***sample workflow patterns*** (e.g.,

"hello world" one-task workflows, chain workflows, fork-join workflows, simple dynamic workflows), all with a few ways of handling data and I/O, and all with a few target execution platforms. Then workflow system teams can provide (interactive) documentation (or could be hosted on a community Web site) on how to run these patterns with their system [37]. Additionally, mechanisms should be identified at the institutional level to commit workflow systems ***training efforts in person***: (i) this should be based on existing facilities and universities efforts; (ii) the scope of the training should be narrowed down so it is manageable; and (iii) the issue of "who trains the trainers?" needs to be addressed.

In light of workforce training, workflow concepts should be taught at early stages of the researchers/users education path. Precisely, these concepts should be included in university curricula, including domain science curricula. Recent efforts have produced pedagogic modules that target workflow education [50], [51]. Pedagogic content could also be distributed as workflow modules to existing software carpentry efforts [52].

There is an established community of workflow researchers, developers, and users that has extensive expertise knowledge regarding specific tools, systems, applications, etc. It is crucial to capture such knowledge and bootstrap a ***community workflow knowledge-base*** (following standards for documentation, interoperability, etc.) for training and education. The workflows community would also benefit from collaborations with social scientists and sociologists so as to help define an overall strategy for approaching some of the above challenges.

VII. Building a Workflows Community

Given the current large size and fragmentation of the workflow technology landscape, there is a clear need to establish a cohesive community of workflow developers and users. This community would be crucial for avoiding unnecessary duplication of effort and would allow for sharing, and thus growing, of knowledge. To this end, there are four main components that need to be addressed for building a community: (i) identity building, (ii) trust, (iii) participation, and (iv) rewards.

A. Brief State-of-the-art and Challenges

The most natural idea is to think of two ***distinct communities***: (i) a Workflow Research and Development Community, and (ii) a Workflow User Community. The former gathers people who share interest in workflow R&D, and corresponding sub-disciplines. Subgroups of this community are based on common methodologies, technical domains (e.g., computing, provenance, design), scientific disciplines, as well as geographical and funding areas. The latter gathers anyone using workflows for optimization of their work processes. However, most domain science users think of themselves in their specific disciplines first, as they just happen to use workflows to get their work done.

The two aforementioned communities are not necessarily disjoint, but currently have little overlap. And yet, it is crucial that they interact. Such interaction seems to happen only on a case-by-case basis, rather than via organized community efforts. One could, instead, envision a single community (e.g., team of users, or "team-flow") that gathers both workflow system developers and workflow-focused users, with the common goal of spreading knowledge and adoption of workflows, thus working towards increased ***sharing and convergence/interoperation*** of technologies and approaches.

Establishing trust and processes is key for bringing both communities together. There is no one-size-fits-all workflow system or solution for all domains, instead each domain presents their own specific needs and have different preferred ways to address problems. There is a pressing need for ***maintaining documentation and dissemination*** that fits different usage options and needs.

B. A Vision for Potential Community Activities

Given the above, there are a number of existing community efforts that could serve as inspiration, e.g., the WorkflowHub Club [19] and Galaxy [53]. One approach is to gather experience from computing facilities where teams have successfully adopted and are successfully running workflow systems [37]. Another possibility is to use proposal/project reviews as mechanisms for spreading workflow technology knowledge. Specifically, finding ways to make proposal authors (typically domain scientists) aware of available technology would prevent their proposed work to not entail re-inventing the wheel. Finally, it is clear that solving the "community challenge" has large overlap with solving the "education challenge" (Section VI).

A short-term activity would entail ***establishing a common knowledge-base for workflow technology*** so that workflow users would be able to navigate the current technology landscape. User criteria (for navigation) need to be defined. Workflow system developers can add to this knowledge base via self-reporting and could include test statuses for a set of standard workflow configurations, especially if workflow systems are deployed across sites. There is large overlap with similar proposed community efforts identified in Sections IV and VI.

An ambitious vision would be to ***establish a "Workflow Guild"***, i.e., an organization focused on interaction and good relationships and self-support between subscribing workflow developers and their systems, as well as dissemination of best-practices and tools that are used in the development and use of these systems. However, there are still barriers to be conquered: (i) such a community could be too self-reflecting, and yet still remain fragmented; (ii) a cultural/social problem is that creating a new system is typically more exciting for computer scientists as opposed to re-using someone's system; and (iii) building trust and reducing internal competition will be difficult, though building community identity will help the Guild work together against external competitors.

VIII. A Roadmap for Workflows Research and Development

In the previous sections, we have listed broad challenges for the workflows community and proposed a vision for community activities to address these challenges. Here, we explore

technical approaches for realizing (part of) that vision. Based on the outcomes of the first summit [7], we identified three technical thrusts for discussion in the second summit [8]. Some of these thrusts align with a single theme of the first summit and some are cross-cutting. In the following subsections, we present the summary of discussions at the second summit and propose roadmap milestones that emerged from these discussions. Additional details can be found in [8], and a summary of the roadmap milestones is shown in Table II.

A. Defining Common Workflow Patterns and Benchmarks

The above sections point to strong needs for establishing repositories of common workflow patterns and benchmarks (Sections III, IV, V, and VI). One objective is to develop workflow patterns in which each pattern should be easy for users to leverage as starting point for their own specific workflow applications – they should provide links to one or more implementations, where each implementation is for a particular workflow system and can be downloaded and easily modified by the user. However, the *level of abstraction* of these patterns (i.e., the level of connection to real application use-cases) should still be defined. At one extreme, workflow patterns could be completely abstract with no connection to any real-world application. At the other extreme, workflow patterns could be completely use-case-driven and correspond to actual scientific applications, with realistic task computations and data sets. The end goal is thus to identify useful patterns that span the spectrum of possible levels of abstraction.

Another aspect is the level of detail with which a pattern specifies the platform on which it is to be executed and the logistics of the execution on that platform. The platform description could be left completely abstract, or it could be fully specified. Under-specifying execution platforms and logistics may render the pattern not useful, but over-specifying them could render the pattern too niche.

Benchmark specifications should make it easy for workflow system developers to develop them or to determine that their system cannot implement these specifications. Each benchmark should provide links to implementations and data sets, where each implementation is for a particular workflow system. These implementations would be provided, maintained, and evolved by workflow system developers. They should be able to be packaged so that they are executed out of the box on the classes of platforms they support. Moreover, the input data of these workflows should be configurable in size to enable both weak and strong scaling experiments. For all configurations, also the output of the workflow must be provided to allow functional testing.

Given the above, the following milestones are proposed:

M1. Define small sets (between 5 and 10) of workflow patterns and of workflow benchmark deliverables. These should be defined by eliciting feedback from users and workflow system developers, as well as based on existing sources that provide or define real-world or synthetic workflow patterns.

M2. Work with a selected set of workflow systems to implement the above patterns and benchmarks.

M3. Investigate options for automatic generation of patterns and/or benchmarks using existing approaches [39], [54].

M4. Identify or create a centralized repository to host and curate the above patterns and benchmarks [19], [39].

B. Paths Toward Interoperability of Workflow Systems

Workflow systems differ at varying degrees such as expressivity, execution models, and ecosystems. These differences are mainly due to individual implementations of language, control mechanisms (e.g., fault tolerance, loops), data management mechanisms, execution backends, reproducibility aspects for sharing workflows, and provenance and FAIR metadata capturing. The need for interoperability is paramount and can happen at multiple technical levels (e.g., task, tools, workflows, data, metadata, provenance, and packaging) as well as non-technical level including semantics, organizational, and legal issues (e.g., licenses compatibility, data sharing policies).

The need for interoperability of workflow applications and systems is commonly modeled as a problem of porting applications across systems, which may require days up to weeks of development effort [55], [56]. Most of the previous approaches for tackling the interoperability problem attempted to develop complete vertical solutions. However, there is no attempt to develop an approach from a perspective of making interoperable components. Interoperable components require standardized APIs, which are still an open challenge [57], [58].

There is a tendency to tie the workflow with its execution model and data structures (e.g, the intertwine between the abstract workflow and its execution). Understanding which component in the workflow system architecture accounts for which functionality, is then paramount. Thus, separation of concerns is key for interoperability at many levels, e.g. separation of orchestration of the workflow graph from its execution.

Given the above, the following milestones are proposed:

M5. Define concrete notions of interoperability for different stakeholders, in particular workflow designers, workflow system designer, and workflow execution organizations.

M6. Establish a "requirements" document per a small set of *abstraction layers* that will (i) capture the commonalities between components of workflow systems; and (ii) perform a separation of concerns to identify interoperability gaps.

M7. Develop real-world workflow *benchmarks* featuring different configurations and complexities (see previous section). Such benchmarks would be key to evaluate the functionality of workflow systems and computing platforms systematically.

M8. Develop *use cases* for interoperability based on real-life scenarios, e.g., porting workflows across platforms that would provide different file system and/or different resource manager.

M9. Develop *common APIs* that represent a set of workflow library components, so as interoperability could be achieved at the component level [17], [59], [60], including APIs for defining inputs, storing intermediate results, and output data.

M10. Establish a workflow systems *developer community*. An immediate activity would be to develop a centralized repos-

itory of workflow-related research papers, and a workflow system registry aimed at DevOps and/or users.

C. Improving Workflow Systems' Interface with Legacy and Emerging HPC Software and Hardware Stacks

Improving the interface between workflow systems and existing as well as emerging HPC and cloud stacks is particularly important as workflows are designed to be used for long periods of time and may be moved between computing providers. This challenge is exacerbated with the specialization of hardware and software systems (e.g., with accelerators, virtualization, containers, and cloud or serverless infrastructures). Thus, it is important to address the challenges faced by workflow systems with respect to discovering and interacting with a diverse set of cyberinfrastructure resources and also the difficulties authenticating remote connections while adhering to facility policies.

Workflow systems require a standard method for querying a site on how to use that site, for example, information about the batch system, file system configuration, data transfer methods, and machine capabilities. It is crucial then to first understand what information is needed by workflow systems, what information could be made available programmatically and what would need to be manually curated (similar ongoing efforts [61] may provide the foundations for this effort).

A key capability provided by workflow systems is remote job execution, which is necessary in cases where workflows span facilities. However, authentication has always been challenging. Many workflow systems rely on fragile SSH connections and in the past the use of GSISSH for delegated authentication. Recently, sites have moved towards two factor authentication and even OAuth-based solutions. There are though ongoing efforts to provide programmatic identity and access management in scientific domains [62], [63]. While the topic of remote authentication is much more broad than the workflows community, there are important considerations that should be included in this discussion related to programmatic access, community credentials, and long-term access.

Given the above, the following milestones are proposed:

M11. Document a machine-readable description of the essential properties of popular sites, e.g., define a JSON schema and share it on GitHub.

M12. Document remote authentication requirements from the workflow perspective and organize an event involving workflow system developers, end users, authentication technology providers, and facility operators.

IX. Conclusion

In this paper, we have documented and summarized the wealth of information acquired as a result of a series of virtual events entitled the "Workflows Community Summit". The goal of these summits was to identify the common and current challenges faced by the workflows community, and outline a vision for short- and long-term community activities. From this vision, we have defined a community roadmap consisting of 12 milestones, which proposes solutions and technical approaches for achieving that vision. This initial series of successful events bespoke the need for continued engagement among workflow researchers, developers, and users, as well as enlarging the scope of the community to also embrace key stakeholders (e.g., computing facility operators, funding agency representatives, etc.) for enabling the proposed vision and roadmap.

References

[1] R. M. Badia Sala *et al.*, "Workflows for science: A challenge when facing the convergence of hpc and big data," *Supercomputing frontiers and innovations*, vol. 4, no. 1, 2017.

[2] R. Ferreira da Silva, R. Filgueira, I. Pietri, M. Jiang *et al.*, "A characterization of workflow management systems for extreme-scale applications," *Future Generation Computer Systems*, vol. 75, 2017.

[3] "Existing workflow systems," https://s.apache.org/existing-workflow-systems, 2021.

[4] E. Deelman *et al.*, "The future of scientific workflows," *International Journal of High Performance Computing Applications*, vol. 32, 2018.

[5] "workflowsRI," https://workflowsri.org, 2021.

[6] A. Al-Saadi, D. H. Ahn, Y. Babuji, K. Chard, J. Corbett, M. Hategan, S. Herbein, S. Jha, D. Laney, A. Merzky *et al.*, "ExaWorks: Workflows for Exascale," *arXiv preprint arXiv:2108.13521*, 2021.

[7] R. Ferreira da Silva, H. Casanova, K. Chard, D. Laney, D. Ahn, S. Jha, C. Goble, L. Ramakrishnan *et al.*, "Workflows Community Summit: Bringing the Scientific Workflows Community Together," Mar. 2021.

[8] R. Ferreira da Silva, H. Casanova, K. Chard *et al.*, "Workflows Community Summit: Advancing the State-of-the-art of Scientific Workflows Management Systems Research and Development," Jun. 2021.

[9] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak *et al.*, "The FAIR guiding principles for scientific data management and stewardship," *Scientific Data*, vol. 3, no. 1, 2016.

[10] C. Goble, S. Cohen-Boulakia, S. Soiland-Reyes, D. Garijo, Y. Gil, M. R. Crusoe, K. Peters, and D. Schober, "FAIR Computational Workflows," *Data Intelligence*, vol. 2, no. 1-2, 2020.

[11] D. S. Katz, M. Gruenpeter, and T. Honeyman, "Taking a fresh look at FAIR for research software," *Patterns*, vol. 2, no. 3, 2021.

[12] "Software and Data Artifacts in the ACM Digital Library," https://www.acm.org/publications/artifacts, 2021.

[13] "FAIR for Research Software (FAIR4RS) WG," https://www.rd-alliance.org/groups/fair-research-software-fair4rs-wg, 2021.

[14] "FAIR for Virtual Research Environments WG," https://www.rd-alliance.org/groups/fair-virtual-research-environments-wg, 2021.

[15] A. Williams *et al.*, "Computationalworkflow: Bioschemas specification for describing a computational workflow," 2021. [Online]. Available: https://bioschemas.org/profiles/ComputationalWorkflow/1.0-RELEASE

[16] S. Soiland-Reyes, P. Sefton, M. Crosas, L. J. Castro, F. Coppens *et al.*, "Packaging research artefacts with ro-crate," *Data Science*, 2021.

[17] M. R. Crusoe, S. Abeln, A. Iosup *et al.*, "Methods included: Standardizing computational reuse and portability with the common workflow language," *Communications of the ACM*, 2021.

Acknowledgments. This work was funded by NSF awards #2016610, #2016682, and #2016619, and by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the US DOE and the NNSA. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. DOE under Contract #DE-AC05-00OR22725. CG and SSR acknowledge funding from European Commission's contracts BioExcel-2 (H2020-INFRAEDI-2018-1 823830), EOSC-Life (H2020-INFRAEOSC-2018-2 824087), IBISBA 1.0 (H2020-INFRAIA-2017-1-two-stage 730976), PREP-IBISBA (H2020-INFRADEV-2019-2 871118), SyntheSys+ (H2020-INFRAIA-2018-1 823827). UL acknowledges funding from the German Research Council for CRC 1404 FONDA. FC is supported by the Research Foundation-Flanders (FWO, I002819N) and by the European Union's Horizon 2020 research and innovation programme under grant #824087 (EOSC-Life). BSC authors acknowledge EuroHPC JU under contract 955558 (eFlows4HPCproject). We thank all participants of the Workflows Community Summits, held in January 2021 and April 2021.

[18] A. Monteil *et al.*, "Nine best practices for research software registries and repositories: A concise guide," *arXiv:2012.13117*, 2020.
[19] "WorkflowHub.eu," https://workflowhub.eu, 2021.
[20] D. Yuen *et al.*, "The dockstore: enhancing a community platform for sharing reproducible and accessible computational protocols," *Nucleic Acids Research*, 2021.
[21] W. Oliveira *et al.*, "Provenance analytics for workflow-based computational experiments: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, 2018.
[22] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, 2017.
[23] B. Van Essen, H. Kim, R. Pearce, K. Boakye, and B. Chen, "Lbann: Livermore big artificial neural network hpc toolkit," in *Workshop on Machine Learning in High-Performance Computing Environments*, 2015.
[24] J. Ozik *et al.*, "From desktop to Large-Scale Model Exploration with Swift/T," in *2016 Winter Simulation Conference (WSC)*, 2016.
[25] J. Ozik, J. M. Wozniak *et al.*, "A population data-driven workflow for COVID-19 modeling and learning," *The International Journal of High Performance Computing Applications*, vol. 35, no. 5, 2021.
[26] J. Ejarque, R. Badia *et al.*, "Enabling dynamic and intelligent workflows for hpc, big data, and ai convergence," *Future Generation Computing Systems*, vol. under review, 2021.
[27] J. L. Peterson, B. Bay, J. Koning, P. Robinson *et al.*, "Enabling machine learning-ready HPC ensembles with merlin," 2021.
[28] T. Akiba, S. Sano, T. Yanase *et al.*, "Optuna: A next-generation hyperparameter optimization framework," in *25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019.
[29] J. Bergstra *et al.*, "Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms," in *12th Python in science conference*, vol. 13, 2013.
[30] J. M. Wozniak, R. Jain, P. Balaprakash, J. Ozik *et al.*, "CANDLE/Supervisor: a workflow framework for machine learning applied to cancer research," *BMC Bioinformatics*, vol. 19, no. 18, 2018.
[31] "ML Commons," https://mlcommons.org/en/, 2021.
[32] G. Fursin, "Collective knowledge: organizing research projects as a database of reusable components and portable workflows with common apis," *arXiv:2011.01149*, 2020.
[33] D. H. Ahn *et al.*, "Flux: Overcoming scheduling challenges for exascale workflows," *Future Generation Computer Systems*, vol. 110, 2020.
[34] S. Heldens *et al.*, "The landscape of exascale research: A data-driven literature analysis," *ACM Computing Surveys*, vol. 53, no. 2, 2020.
[35] S. Prathiba and S. Sowvarnica, "Survey of failures and fault tolerance in cloud," in *2nd International Conference on Computing and Communications Technologies (ICCCT)*, 2017.
[36] P. A. Ewels *et al.*, "The nf-core framework for community-curated bioinformatics pipelines," *Nature biotechnology*, vol. 38, no. 3, 2020.
[37] "NERSC Workflow Management Tools," https://docs.nersc.gov/jobs/workflow-tools/, 2021.
[38] "OpenEBench," https://openebench.bsc.es, 2021.
[39] T. Coleman, H. Casanova, L. Pottier, M. Kaushik, E. Deelman, and R. F. da Silva, "Wfcommons: A framework for enabling scientific workflow research and development," *Future Generation Computer Systems*, 2021.
[40] D. S. Katz, "Introducing the software development curve," https://danielskatzblog.wordpress.com/2021/05/14/software-development-curve/, May 2021.
[41] G. Terstyanszky *et al.*, "Enabling scientific workflow sharing through coarse-grained interoperability," *Future Generation Computer Systems*, vol. 37, 2014.
[42] "Common Workflow Language: Metadata and Annotations," https://www.commonwl.org/user_guide/17-metadata/index.html, 2021.
[43] D. S. Katz, R. M. Badia, C. Kyle, and J. Ejarque, "JLESC research project: A common workflow registry of compute endpoints and applications," https://jlesc.github.io/projects/workflow-endpoint-registry/, 2020.
[44] "GA4GH-DREAM Workflow Execution Challenge," http://dx.doi.org/10.7303/syn8507133, 2021.
[45] R. Chard, Y. Babuji, Z. Li *et al.*, "Funcx: A federated function serving fabric for science," in *29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020.
[46] F. Smirnov *et al.*, "Apollo: Modular and distributed runtime system for serverless function compositions on cloud, edge, and iot resources," in *1st Workshop on High Performance Serverless Computing*, 2020.
[47] M. Malawski, A. Gajek *et al.*, "Serverless execution of scientific workflows: Experiments with hyperflow, aws lambda and google cloud functions," *Future Generation Computer Systems*, vol. 110, 2020.
[48] "Workflow Patterns," http://workflowpatterns.com, 2021.
[49] D. Garijo, P. Alper, K. Belhajjame, O. Corcho, Y. Gil, and C. Goble, "Common motifs in scientific workflows: An empirical analysis," *Future Generation Computer Systems*, vol. 36, 2014.
[50] H. Casanova, R. Tanaka, W. Koch, and R. F. da Silva, "Teaching Parallel and Distributed Computing Concepts in Simulation with WRENCH," *Journal of Parallel and Distributed Computing*, vol. 156, 2021.
[51] "The EduWRENCH Pedagogic Modules," https://eduwrench.org, 2021.
[52] "Software Carpentry," https://software-carpentry.org, 2021.
[53] "Galaxy Community Hub," https://galaxyproject.org, 2021.
[54] D. S. Katz, A. Merzky *et al.*, "Application skeletons: Construction and use in escience," *Future Generation Computer Systems*, vol. 59, 2016.
[55] C. Schiefer *et al.*, "Portability of scientific workflows in ngs data analysis: a case study," *arXiv preprint arXiv:2006.03104*, 2020.
[56] F. Lehmann, D. Frantz *et al.*, "Force on nextflow: Scalable analysis of earth observation data on commodity clusters," in *Int. Workshop on Complex Data Challenges in Earth Observation*, 2021.
[57] M. Turilli *et al.*, "Middleware building blocks for workflow systems," *Computing in Science & Engineering*, vol. 21, no. 4, 2019.
[58] J. J. Billings and S. Jha, "Toward common components for open workflow systems," *arXiv preprint arXiv:1710.06774*, 2017.
[59] J. Arshad, G. Terstyanszky, T. Kiss, and N. Weingarten, "A definition and analysis of the role of meta-workflows in workflow interoperability," in *7th International Workshop on Science Gateways*, 2015.
[60] G. Fursin, "Collective knowledge: organizing research projects as a database of reusable components and portable workflows with common interfaces," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2197, 2021.
[61] "SGCI Resource Inventory," https://github.com/SGCI/sgci-resource-inventory, 2021.
[62] J. Alt *et al.*, "Oauth ssh with globus auth," in *Practice and Experience in Advanced Research Computing*, 2020.
[63] A. Withers, B. Bockelman, D. Weitzel *et al.*, "Scitokens: Capability-based secure access to remote scientific data," in *Practice and Experience on Advanced Research Computing*, 2018.