

Advancing Digital Transformation in Material Science: The Role of Workflows Within the MaterialDigital Initiative

Simon Bekemeier, Celso Ricardo Caldeira Rêgo, Han Lin Mai, Ujjal Saikia, Osamu Waseda, Markus Apel, Felix Arendt, Alexander Aschemann, Bernd Bayerlein, Robert Courant, Gordian Dziwis, Florian Fuchs, Ulrich Giese, Kurt Junghanns, Mohamed Kamal, Lukas Koschmieder, Sebastian Leineweber, Marc Luger, Marco Lukas, Jürgen Maas, Jana Mertens, Björn Mieller, Ludger Overmeyer, Norbert Pirch, Jan Reimann, Sebastian Schröck, Philipp Schulze, Jörg Schuster, Alexander Seidel, Oleg Shchyglo, Marek Sierka, Frank Silze, Simon Stier, Marvin Tegeler, Jörg F. Unger, Matthias Weber, Tilmann Hickel, and Jörg Schaarschmidt*

The MaterialDigital initiative represents a major driver toward the digitalization of material science. Next to providing a prototypical infrastructure required for building a shared data space and working on semantic interoperability of data, a core focus area of the Platform MaterialDigital (PMD) is the utilization of workflows to encapsulate data processing and simulation steps in accordance with findable, accessible, interoperable, and reusable principles. In collaboration with the funded projects of the initiative, the workflow working group strives to establish shared standards, enhancing the interoperability and reusability of scientific data processing steps. Central to this effort is the Workflow Store, a pivotal tool for disseminating workflows with the community, facilitating the exchange and replication of scientific methodologies. This article discusses the inherent challenges of adapting workflow concepts, providing the perspective on developing and using workflows in the respective domain of the various funded projects. Additionally, it introduces the Workflow Store's role within the initiative and outlines a future roadmap for the PMD workflow group, aiming to further refine and expand the role of scientific workflows as a means to advance digital transformation and foster collaborative research within material science.

1. Introduction


Digital workflows are becoming increasingly important in materials science and engineering, following the general trend that data handling shifted from analog documents to digital files and databases. Furthermore, the amount of data generated or processed within the domain increases drastically, also due to improved measurement techniques and simulation approaches.

The MaterialDigital initiative addresses this aspect on a broad basis by building a general framework for the digitalization of materials science based on an IT infrastructure featuring ontological data representation and workflows as central connecting components.^[1] The prominent position of workflows is due to the fact that they are omnipresent as soon as information and data are available in digital form and need to be transferred or transformed. Along the value chain of data, this spans

1) Data acquisition, where data is obtained

S. Bekemeier, U. Saikia, B. Bayerlein, B. Mieller, J. F. Unger, T. Hickel
 Bundesanstalt für Materialforschung- und prüfung (BAM)
 Unter den Eichen 87, 12205 Berlin, Germany

S. Bekemeier
 Hochschule Bielefeld - University of Applied Sciences and Arts (HSBI)
 Interaktion 1, 33619 Bielefeld, Germany

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/adem.202402149>.

© 2025 The Author(s). Advanced Engineering Materials published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/adem.202402149

C. R. Caldeira Rêgo, J. Schaarschmidt
 Institute of Nanotechnology (INT)
 Karlsruhe Institute of Technology (KIT)
 Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
 E-mail: joerg.schaarschmidt@kit.edu

H. L. Mai, O. Waseda, T. Hickel
 Max Planck Institute for Sustainable Materials
 Max-Planck-Str. 1, 40237 Düsseldorf, Germany

M. Apel, L. Koschmieder
 Access e.V.
 Intzestr. 5, 52072 Aachen, Germany

from the physical world and integrated into digital data spaces (e.g., an ontology); 2) Process description, where the steps to achieve experimental results are digitally represented (e.g., in an electronic lab notebook) or even steered by a digital infrastructure (e.g., in self-driving labs); 3) Computer simulations that obtain data from virtual representations of real-world objects (e.g., in ab initio simulations); 4) Data-processing pipelines that connect acquisition and/or simulation with analytics, visualization and interpretation as endpoint (e.g., in AI/ML approaches).

While every scientist uses workflows in a broader sense, they differ significantly in their traceability and reproducibility. Many are manually executed using rudimentary tools like Excel sheets and chain arbitrary undocumented transformation steps that elude the comprehensibility of others and often even the originator himself after some time. This results-focused view of scientific data processing not only leads to many errors and poor efficiency, but also prevents the establishment of findable, accessible, interoperable, and reusable (FAIR) principles^[2] for scientific data and methods. To overcome this, digital workflows in a narrower sense describe every step in a human-readable and machine-actionable way that allows shareable and reproducible definitions of data processing and software integration. Since the boundary conditions and requirements vary considerably, this still results in individual solutions (e.g., shell scripts) that are only theoretically

interoperable and reproducible and often fail in practice due to differently configured software environments and data structures.

To manage a workflow as a chain of well-documented process steps that generate or process data for a specific problem and deliver a specific set of results, workflow environments provide a harmonized approach by enforcing a certain structure and providing tooling around it. The advantages of using workflow environments are: 1) provide integration and easy access to high-performance computing (HPC) resources; 2) record complex individual calculation processes for documentation and distribution (e.g., for a paper, an IP application, a collaboration); 3) provide a user-friendly interface to a variety of tools; 4) enable interoperability of workflows by making complex links between individual software tools accessible; 5) automate the execution of workflows, for example, for high-throughput applications; 6) handle storage of the final results and all relevant intermediate steps (e.g., in database systems, repositories); 7) connect to community-wide semantics and knowledge graphs (ontologies) by describing the in- and output of individual tools within a workflow chain.

However, workflow environments are typically created within a specific community determined by its scientific field (e.g., life sciences, chemistry, solid-state physics, engineering, data science) and requirements (integrated tools, HPC application, graphical user interface, common marketplace, etc.). Inter-

F. Arendt, M. Sierka
Otto-Schott-Institut für Materialforschung
Friedrich-Schiller-Universität Jena
Löbdergraben 32, 07743 Jena, Germany

A. Aschemann, U. Giese
Deutsches Institut für Kautschuktechnologie e. V.
Eupener Straße 33, 30519 Hannover, Germany

R. Courant, J. Maas, J. Mertens
Mechatronic Systems Lab
Technische Universität Berlin
Hardenbergstraße 36, 10623 Berlin, Germany

G. Dziwis, K. Junghanns
Institute for Applied Informatics Association (InfAI)
Goerdelerring 9, 04109 Leipzig, Germany

F. Fuchs, J. Schuster
Fraunhofer ENAS
Technologie-Campus 3, 09126 Chemnitz, Germany

F. Fuchs, P. Schulze, J. Schuster
Chemnitz University of Technology
09107 Chemnitz, Germany

M. Kamal
Institute of Digital and Autonomous Construction
Hamburg University of Technology
Blohmstraße 15, 21079 Hamburg, Germany

S. Leineweber, M. Lukas, L. Overmeyer
Institut für Transport- und Automatisierungstechnik
Gottfried Wilhelm Leibniz Universität Hannover
An der Universität 2, 30823 Garbsen, Germany

M. Luger
Technology Campus Hutthurm
Deggendorf Institute of Technology
Hochleiten 1, 94116 Hutthurm, Germany

N. Pirch
Fraunhofer ILT
Steinbachstr. 15, 52074 Aachen, Germany

J. Reimann
Department of Mechanical Engineering
Production Technology Group
Technische Universität Ilmenau
Gustav-Kirchhoff-Platz 2, 98693 Ilmenau, Germany

S. Schröck
Institute of Applied Informatics
Deggendorf Institute of Technology
Grafenauerstr. 22, D-94078 Freyung, Germany

P. Schulze
Institute of Mathematics
Technische Universität Berlin
10623 Berlin, Germany

A. Seidel
TUM School of Engineering and Design
Chair of Carbon Composites
Technical University of Munich
Boltzmannstr. 15, 85748 Garching, Germany

O. Shchyglo, M. Tegeler
OpenPhase Solutions GmbH
Universitätsstraße 136, 44799 Bochum, Germany

O. Shchyglo
ICAMS
Ruhr-Universität Bochum
Universitätsstraße 150, 44801 Bochum, Germany

F. Silje
OSCAR PLT GmbH
Hamburger Ring 11, 01665 Klipphausen, Germany

S. Stier
Fraunhofer Institut für Silicatforschung ISC
Neunerplatz 2, 97082 Würzburg, Germany

M. Weber
Fraunhofer Institute for Mechanics of Materials IWM
Wöhlerstrasse 11, 79108 Freiburg, Germany

operability beyond the own ecosystem is mostly a secondary aspect. Especially semantic/ontology-based descriptions of workflow steps are a fairly new aspect for most initiatives. As a result, there is an abundance of incompatible solutions (<https://github.com/meirwah/awesome-workflow-engines>) covering different features and application scenarios.

As a framework for digital material science, the Platform MaterialDigital (PMD) therefore focuses on workflow environments that support methodologies and tools established in this domain by covering use cases represented by research projects within the platform. These include, among others, research on glass, metal alloys, polymers, concrete, or functional materials.

To enhance interoperability across various research projects, we support two dedicated workflow environments: pyiron^[3] and SimStack.^[4] These well-established environments address critical requirements within the materials science domain. In this article, we consolidate these solutions within the workflow landscape and explore their practical applications across various research fields. Furthermore, we offer an overview of a recently deployed common workflow store, intended to facilitate the transition to workflows that comply with FAIR criteria,^[2] along with insights into its ongoing development related to semantic self-description, standardization, and industry adoption.

2. Workflows

2.1. Definitions

The term workflow is used with different annotations in the materials science community. In the present article, we refer to a workflow as an abstract layer on top of individual software components or digital interfaces to experimental devices. Within this abstract layer, a workflow consists of a chain of well-documented process steps to generate or handle data for a specific materials science problem in order to deliver a particular set of outputs. Accordingly, data acquisition from an experimental device into an electronic lab notebook and subsequently into an ontology is a workflow for us, similar to the combination of two software tools in a computer simulation.

The different steps of the digital process are implemented as nodes. The combination of these nodes including their in- and outputs, as well as the connections of the output of nodes to the input of other nodes, forms the workflow. In principle, such a workflow node can also consist of several nodes that are combined in a macro-node. An atomic node, on the contrary, cannot be split into individual steps. An atomic node alone is therefore the smallest possible implementation of a workflow, still consisting of an input, the respective (even complex) processing task, and an output. A typical example is the computation of a single total energy for a given crystal structure with an ab initio software tool.

2.1.1. Required Features of Workflow Nodes

To implement this abstract concept in a workflow engine, one has to specify the features of these atomic nodes. In the first place, an atomic node is required to be represented by a pure function without internal state variables that needs to be transferable to a remote compute server. To be part of a workflow, it is

further required that such a node has predefined, serializable, and well-documented inputs and outputs.

In a workflow engine, these inputs and outputs will be simple parameters represented in basic data types such as integers, floats, vectors, arrays, or strings. An ab initio calculation can, for instance, require a set of atomic coordinates (array) as input in order to deliver an energy (float) as an output. However, since even a simple crystal structure contains not only coordinates but also the information about the atomic species, more complex input formats, such as dictionaries, or structured and hierarchical objects (e.g., HDF5) need to be supported inputs as well.

For the sake of efficiency, these objects could, similar to the simple parameters, be exchanged between different nodes in the shared memory of the compute environment. For a workflow engine that operates in Python (e.g., pyiron), we expect that inputs and outputs are exchanged as Python objects. Since a workflow node can also be represented as a Python object, it can thus be an input of another workflow node. However, our definition of workflow nodes does also include that in- and outputs can also be stored in files, be it structured file formats (e.g., json) or even plain text files.

The file format is, in particular, required if the individual workflow nodes need to be executed in different environments, which might even have conflicting dependencies of libraries or software packages. It is therefore also required that each atomic node carries the information about its execution environment including the employed library versions as specific as necessary to make the workflow reproducible (e.g., in the form of conda environments or docker images).

Finally, it is also required that the workflow node exposes all relevant error and even warning messages, generated by the lower-level software tools during execution, to the workflow environment. Thereby, it enables the workflow environment to document possible error propagation and to assist the user in identifying and fixing potential bugs in the workflow.

2.1.2. Further Concepts for Workflow Nodes

One of the main objectives of the MaterialDigital workflow group is the semantic annotation of workflows to achieve the interoperability of workflow nodes. One way is the enforcement of an ontological typing of all inputs and outputs of the nodes. In this way, workflow frameworks can ensure that during transfer of information from one node to the next not only the expected data types align (e.g., float value) but also that the transfer is semantically correct. Based on this information, it is even possible to achieve an automatized generation of workflows from a semantic description.

To handle the increasing complexity of workflows in materials science, the implementation of unit tests is important to ensure functionality and reliability. Unit tests execute parts of the workflow (atomic nodes and their combination) with specific input and can be included in a continuous integration strategy.

There are several other optional features of workflow nodes that are specific to certain implementations of workflow engines,^[5–7] such as task managers within a node, remote monitoring of the status of output files,^[8] the description of nodes as Python wrappers,^[3] and many more. However, to keep the description of this review article concise, we do not aim for

completeness of their description but rather focus on the features that are central to the use cases in MaterialDigital.

2.2. Implementation

The use cases described in detail in Section 3 all address problems in materials science, but still originate from a diverse set of researchers. As a consequence, a variety of established workflow environments are used. This reflects the diversity of the underlying constraints imposed by the scientific questions. For example, computational workflows based on *ab initio* methods can often be executed within a single and consistent software environment. However, the simulation of a macroscopic device often requires the use of different software environments, which in most cases requires file-based communication. In the following, we provide a categorization of the different workflow implementations used in MaterialDigital, while details can only be provided for those workflow environments that are supported as services to the community.

2.2.1. Command-Line-Wrapper

A command-line-wrapper aims to replace a sequence of stand-alone command line operations with an encapsulated and formal sequence definition. Consequently, data transfer is based on arguments and files. A command-line-workflow environment typically supports control structures like loops and conditions, global parameters, and variables that are set from return values and can be reused in subsequent commands. The advantage is the easy integration of various software tools as long as they expose a command-line interface. On the downside, this approach introduces a custom wrapper language with limited expressiveness (in comparison to a full-fledged programming language). Examples for command-line wrappers are *retflow*, *Snakemake*, and *SimStack* (WaNos).

2.2.2. Code Wrapper

A code wrapper like *pyiron* aims to replace a sequence of function calls within the same language. In doing so, data transfer is typically based on in-memory arguments and return values. Usually a code-workflow environment focuses less on control structures and variables, since these are already provided by the underlying programming language, but more on logging of function calls and persisting of intermediate and final results. On the pro side, this does not require learning an additional language and allows to reuse all existing language/IDE features, like doc strings, auto-completion, or unit tests. However, working in a specific language could make it more difficult to integrate other languages, if not provided by existing tool sets (e.g., *ctypes* in Python). In addition, there is also an overhead when integrating other components in machine code (e.g., binary command-line executables).

2.2.3. Low Code and Graphical Workflows

Workflows implementations that require only a few (low code) or even no programming skills at all (no code) usually provide a

graphical user interface to build a workflow by parameterizing and connecting building blocks. From the user perspective, data transfer occurs by abstract wiring of building blocks. These are subsequently automatically translated into command-line sequences or program code in the backend. This approach has the obvious advantage that users do not have to learn any programming language. However, building blocks in the UI have limited expressiveness and extending feature requires more effort. Due to the complexity, this is typically out of scope for most end users. Furthermore, complex flow logic is difficult to handle and may lead to inefficient back-end-generated code. Examples of such implementations are the *SimStack-UI* and *Node-RED*.

2.2.4. Deployment, Orchestration, and Management

While workflow development may take place locally, deployment to a containerized or cloud/HPC environment is typically used in production. On the one hand, this makes sure that the workflow code and dependencies are bundled correctly. On the other hand, this helps to scale the workflow on a suitable hardware infrastructure and allows 24/7 operations.

Automated execution of workflows on remote infrastructure implies the need for orchestration and management solutions. Typical solutions are web services that allow to trigger workflow runs (e.g., via UI or REST-API), track their state executions, and access their results and logs. Optional features may include workflow parameterization and management of secrets, access keys, etc.

2.3. Specific Workflow Environments in PMD

As outlined earlier, the sustainability and interoperability of the software solutions and the uniform access to data and tools are decisive for the workflow activities in the PMD. The platform is therefore providing the two workflow environments “*pyiron*” and “*SimStack*” (Figure 1). The associated projects are expected to design their workflow solutions in a way that they can be made executable within one of these workflow environments. Their availability should allow the user to focus on science rather than on technical details such as input/output formats of codes and tools. As mentioned earlier and in several reviews,^[7] there exist various other workflow environments, including commercial solutions, but covering them extensively outside of the specific use cases is beyond the scope of the PMD and this article.

2.3.1. Pyiron

The workflow environment *pyiron* is designed as an integrated development environment (IDE) for the implementation, execution, and analysis of scientific workflows. As such, it provides all the necessary tools to integrate existing methods into a common platform, to interactively execute complex simulation protocols combining different computer codes, and to perform millions of separate calculations on powerful computer clusters. At the same time, *pyiron* goes beyond the management and execution of a chain of previously available workflow steps by allowing the user to interactively develop, implement, and analyze new

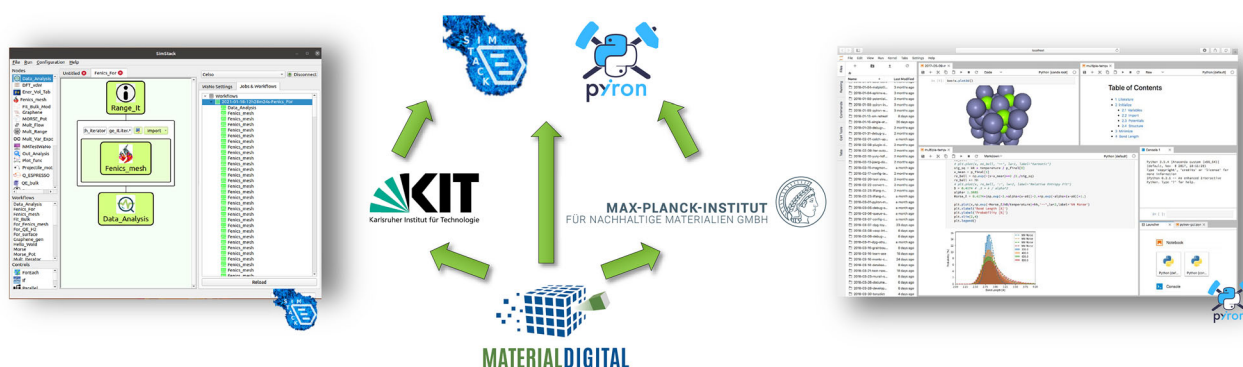


Figure 1. The Platform MaterialDigital (PMD) supports two workflow frameworks through its partner institutions: SimStack via KIT and pyron via MPI SusMat. SimStack (left) offers an easy-to-use graphical user interface, while pyron (right) provides a more powerful but more demanding interface through the use of Jupyter notebooks.

workflow steps in order to address novel scientific questions. This requires a high level of flexibility for the user, while the advantages of workflow environments listed in Section 1 including the automated handling of technical tasks in the background, should still apply.

This ambivalence is resolved in pyron by employing Jupyter notebooks. They provide the right balance between a graphical user interface (GUI) and the freedom to write completely new code. The advantage is a user-friendly structuring of the code, including section titles and the collapsing/folding of certain sections. It furthermore allows a detailed documentation of the workflow, using the flexible Markdown language, in which even mathematical equations can be represented. Most importantly, however, a graphical representation of all intermediate results allows the scientist to directly interact with the scientific performance of the workflow.

To keep the workflow well-structured, the pyron IDE is built of an abstract class of Python objects, which can be combined like building blocks. This structure supports the node design of workflows and the replacement of nodes with alternative ones. One of the objects is the pyronTable, which allows the efficient aggregation of large datasets in Pandas tables. Other objects handle the parsing of input and output for software tools and Python bindings to ensure their straightforward integration.

The basic idea behind pyron is to provide a single tool with a uniform interface for various simulation codes used in MaterialDigital as well as analysis and visualization tools. Currently, the majority of the applications of pyron are in the field of ab initio thermodynamics,^[9] in the atomistic description of defects^[10] and the development of electronic structure methods.^[11] Continuous efforts are made to integrate developments made in the different projects as outlined in Section 3. To reflect the diversity of materials-scientific challenges in PMD, pyron offers different sub-packages (and corresponding github repositories) for classes of software packages, including pyron_experimental, pyron_atomistics, and pyron_continuum. How these sub-packages can be combined is addressed in the UseCase StahlDigital Section 3.12. At the same time, the projects contributed to improving the functionality of pyron, as, for example, demonstrated by the front-end developments in the usecase of DiMoGraph (Section 3.3).

2.3.2. SimStack

The workflow environment SimStack allows users to construct and modify workflows through a drag-and-drop GUI. This minimizes the need for extensive computational expertise, allowing users to focus on scientific inquiry. It is particularly beneficial in fields such as materials science that require integrating various computational tools and methods.^[4,7]

The basic building block (node) of a SimStack workflow is referred to as Workflow active Node (WaNo). A WaNo is an XML file that describes the expected input, configurable parameters, the output generated by the WaNo, and the code to be executed. These WaNos are the atomic nodes in a workflow implementation, providing a structured and modular approach to defining the computational task. Using a simple drag-and-drop interface, users can quickly create new workflows from available building blocks or modify existing workflows to tailor a custom solution to a scientific problem. Using XML for describing user input, coupled with a templating engine (Jinja), allows for quickly incorporating user-defined parameters into the workflow. This system enables the simple integration of an arbitrary software or script, enhancing the flexibility and adaptability of the workflow to meet diverse computational needs.

A core strength of SimStack is its ability to connect to remote high-performance computing (HPC) resources with a single click, automating data transfer and job execution. This client-server model enhances the scalability, reproducibility, and flexibility of simulation protocols.

SimStack can support a variety of simulation methods and has been used in various applications, demonstrating its versatility and effectiveness. Examples include umbrella sampling to calculate binding free energies, exciton dynamics to study organic light-emitting diodes (OLED), dihedral scanning to parameterize molecular dynamics, emission spectra calculations for organic molecules,^[4,7] an active learning approach to model solid-electrolyte interphase formation in battery materials,^[12] the study of thermoelectric materials,^[13] as well as the design of organic semiconductors based on metal-organic frameworks.^[14] These use cases illustrate SimStack's capacity to address complex scientific problems by facilitating the integration of different computational techniques and tools into coherent workflows.

Detailed tutorials and installation documentation are provided at simstack.readthedocs.io/en.

2.3.3. Comparison of Pyiron and SimStack

Pyiron and SimStack are both frameworks designed to streamline computational workflows (see Figure S1, Supporting Information). For the IDE pyiron, Python interfaces such as Jupyter notebooks offer a high-level interface for the implementation, execution, and analysis of scientific simulations. SimStack is a framework that uses a graphical user interface, automatically generated from the XML description of its building blocks (WaNos) to set up and submit complex computational jobs. With proper integration of workflows to both frameworks, users can choose between a simple, easy-to-use GUI or a more powerful IDE as a front-end, while execution is handled by the available back-end.

3. Use Cases

Next to the PMD, the associated initiative includes several projects that address specific fields in materials science. Together with the platform, these projects aim to develop prototypical solutions for the digitalization of their respective fields. This results in a variety of perspectives and approaches to the implementation of FAIR workflows within the initiative. Further, a variety of tools along the data value chain have been used, benefiting from the advantages and fulfilling the different needs that are connected with workflow environments (cf. the list in the Section 1). To provide examples for these advantages, a selection of corresponding use cases from the projects is presented in the following. The matrix in **Figure 2** serves as a guide, in which use case (rows)

and for which stage along the data value chain (colors) which aspect of workflows (columns) is addressed.

3.1. DIGIT RUBBER: From Data Acquisition to Real-Time AI-Based Control

In this project, an example of a workflow from data acquisition on a rubber extruder through the identification of cause–effect relationships is implemented. Challenges related to storage and automatization are addressed (cf. Figure 2), outlining the implementation of a data mining algorithm to a real-time capable AI control system.

In other domains, significant progress has been made in identifying cause–effect relationships, underscoring the transformative potential of data-driven models. For instance, Ghosh's work provides a pivotal perspective on integrating physics-informed machine learning into materials science, effectively bridging theoretical frameworks with experimental observations to optimize materials and evaluate their performance.^[15] In another domain, Miklin et al. introduce an entropic approach to understanding causality in quantum mechanics. Using Bayesian networks and counterfactual variables, their study derives inequalities that offer a characterization of causal correlations across both bipartite and multipartite scenarios.^[16] In their study, Saha et al. demonstrate the superiority of span-based models over traditional sequence tagging methods for extracting causality from text. By utilizing pre-trained language model embeddings such as BERT, their approach consistently outperforms state-of-the-art methods across diverse datasets.^[17] These studies highlight the potential of data-driven models to identify cause-and-effect relationships in various fields. However, they also reveal a gap in utilizing these relationships for real-time implementation

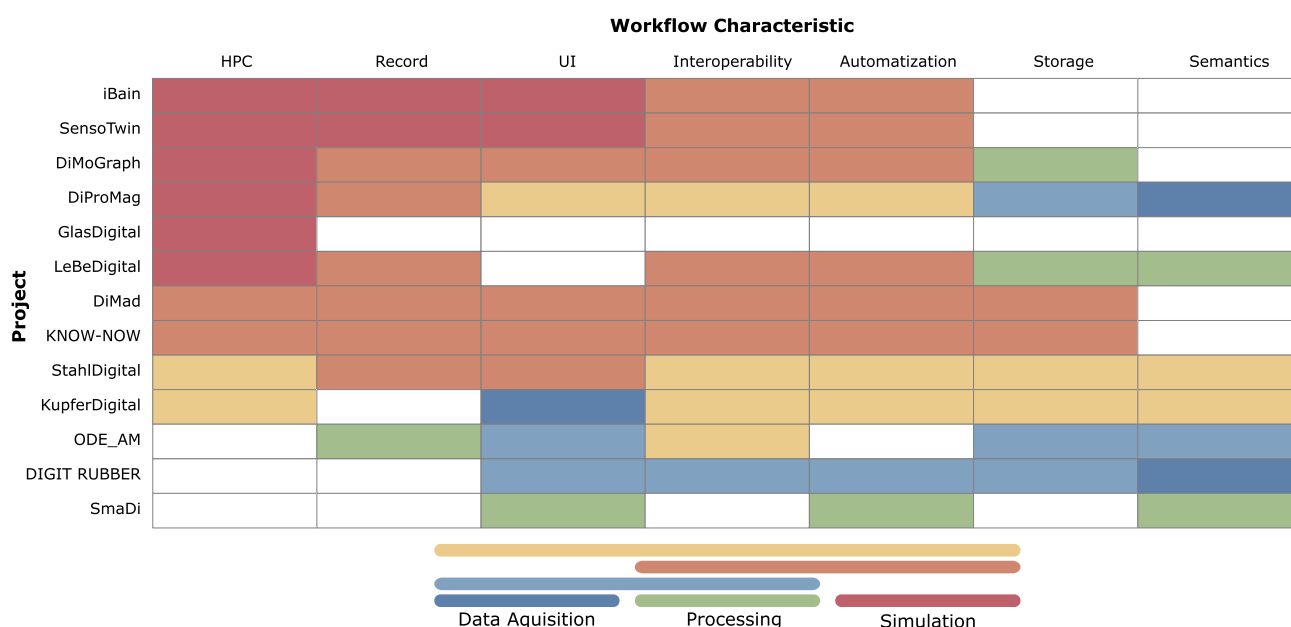


Figure 2. Overview of the workflow characteristics of the use cases presented. The advantages of workflow environments are shown on the horizontal axis (for reference see Section 1). Each highlighted cell marks the aspects that are covered (column) by the use case of a participant project (row). Each cell is colour coded to indicate the stage along the data value chain (for reference see Section 1) at which the tool delivers the given benefit.

in automated process control. To bridge this gap, our article introduces a workflow that integrates data-driven models into a real-time automated control system, improving efficiency and adaptability in dynamic industrial environments.

Using a rubber extrusion line from the 1980s as an example, the path to digitalizing the systems as a workflow is presented here. The aim is to transfer the described method of using neural networks to identify process-inherent cause-and-effect relationships and a feedback of AI-based control variables in real time to other industrial systems in the future, regardless of the process and material. The extrusion unit has been expanded with sensors for temperature and pressure recording, as well as for recording geometries of the extrudate as indicated in **Figure 3**.

In addition to control variables for temperature and speed control, these measured variables are managed in a time series database. This database serves as fundamental training data for the AI-based control system. Various rubber compounds based on ethylene-propylene-diene rubber (EPDM) are being investigated. Real-time data recording, processing, storage, and feedback to the extrusion unit represent a central challenge.

A data mining algorithm based on neural networks (NN) is used to identify correlations between control and measurement variables. The correct configuration of the NN is a key challenge. If measured values deviate from previously defined target values, AI-based control systems predict control variables in such a way that measured variables relevant to quality move back within a previously defined corridor. The detected correlations provide the basis for predicting control variables. The AI can be trained offline on a basis of recorded data and integrated into the real process chain once sufficient training progress has been made.

To realize real-time capable process control, a corresponding hardware interface for data transmission was created for the extruder and the conveyor belt. An Arduino microcontroller

(UNO R3) was integrated into the existing control module so that a network connection can be established via ethernet.

The BITMOTECOSystem^[18] is connected to the network for data transmission and decentralized data storage. This system contains various open source modules, including an integrated “Message Queuing Telemetry Transport” (MQTT) broker and the graphical network tool Node-RED,^[19] which are the basis for automating and managing the data streams. Furthermore, the time series-based database management system InfluxDB^[20] is used for data storage, as shown in Figure 3.

In this way, the measured process data from the extruder, conveyor belt, and other measuring instruments can be transmitted to the database via MQTT messages and following data processing in Node-RED.

The AI-based control system is designed to ensure adherence to user-defined tolerance limits, adapting to the variability of rubber material and external influences. To achieve this, a data mining algorithm that accurately predicts outcomes from extrusion inputs was developed based on a dataset of 14 923 measurement points across various tests for training. Through a combination of iterative development and the hyperopt framework, the feedforward neural network-based algorithm achieves a high accuracy of more than 99% for temperature predictions throughout the extrusion process.^[21–23] The algorithm’s applicability was further demonstrated by successfully extending its predictive capabilities to the rubber-mixing process, achieving an accuracy equivalent to that observed in the extrusion line. Upon exceeding a tolerance threshold, the system uses both the data mining model and its inverse to compute the necessary input variables. The inverse model computes the input variables necessary to meet the defined tolerance limits, while the data mining model validates that these inputs produce the desired output variables, ensuring compliance with the specified target values. This

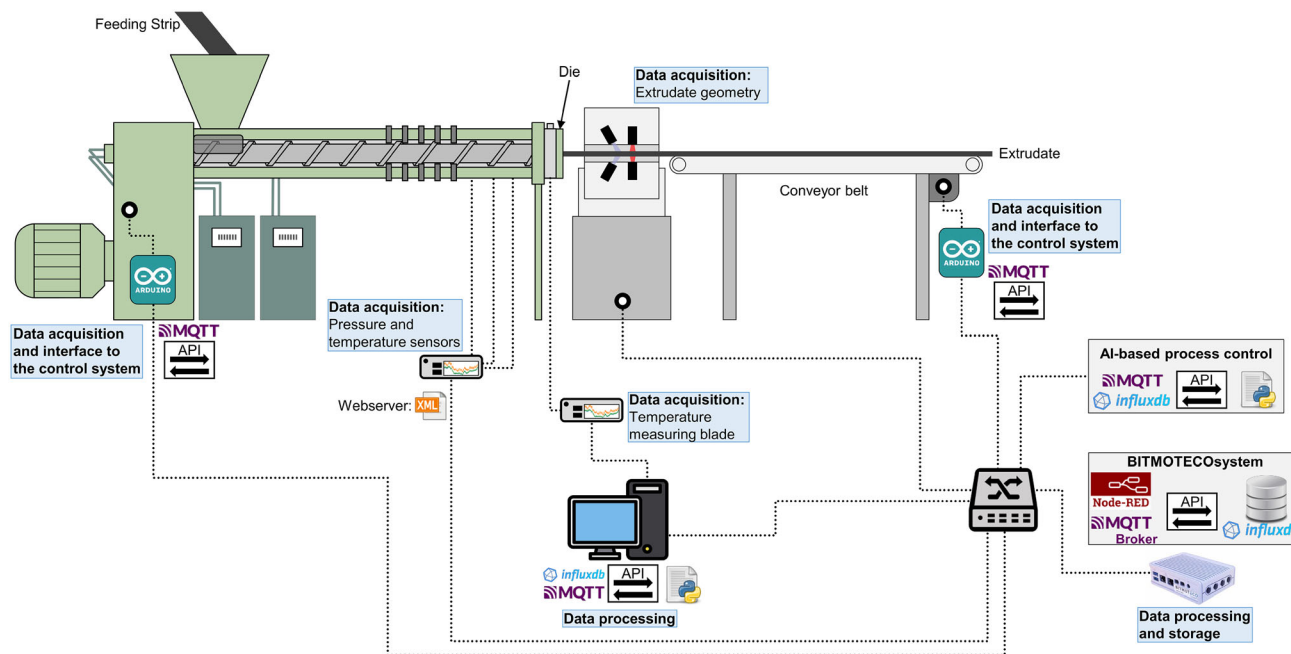


Figure 3. Visualization of the workflow from data acquisition, data processing to data storage during the extrusion process.

approach allows for precise adjustment of the target parameter, regardless of whether it needs to be reduced or increased, and ensures controlled maintenance within tolerance thresholds. Real-time application is facilitated through an InfluxDB for data collection in 1-s time intervals and an Arduino with MQTT broker for efficient parameter communication to the extrusion line. Since each inference per model requires ≈ 12 ms on the utilized hardware, the total time for a single iteration is 24 ms. With an average of two iterations needed to converge, the modeling process is completed in 48 ms. Taking into account network latency, which averages around 20 ms for variable transmission, the system achieves a total response time of 68 ms. The AI-based control system has been validated through both simulations and practical in-line applications on the extrusion line, showcasing its ability to precisely regulate temperatures.^[24] By dynamically adjusting to specific batch conditions and adhering to defined tolerance thresholds, the system proves its robustness and ability to reduce scrap rates in different operating scenarios.

Using the example of a rubber extruder, it was shown for the first time how an inventory extrusion line can be modified in terms of hardware and software to digitalize and implement an AI-based control system. The workflow for recording, processing, storing, and returning data to the extruder was then described. In particular, the functionality of the AI-based control algorithm was highlighted. In the future, this technology will need to be transferred from the laboratory level to industrial extrusion lines. This will also result in a larger data base for further developing AI. Additional measurement techniques can also be implemented to improve quality prediction. The aim is also to achieve cross-process networking through AI-based control. In the example of rubber extrusion, this would also include mixing, milling, and vulcanization. The digitalization of rubber extrusion is intended as an example process and can be transferred to other industrial plants. The developed system presents a framework that facilitates the integration of AI-based workflows into the comprehensive MaterialDigital workflow environments. For this purpose, the PMD platform is used to provide measurement data as well as the code for data mining and AI-based control.

3.2. DiMad: Advancing Wire-Based Additive Manufacturing Through Integrated Simulation Workflows

The DiMad project addresses the challenge of developing a robust digital framework for wire-based additive manufacturing using electric arc and laser radiation as heat sources. The current state of our workflow integrates experimental studies with advanced simulation tools, realized within the MaterialDigital (PMD) infrastructure. Challenges arise for the automatization and interoperability of diverse tools (cf. Figure 2), data transfer, and collaborators across institutions, making seamless integration and reproducibility critical.

From the material point of view, two high-alloy steels, the austenitic stainless steel 316L and the duplex steel grade Duplex 2205, are used in a comprehensive experimental study eventually resulting in the development of a demonstrator component accompanied by a digital representation of the experimental workflow together with numerical simulations.

Our research focuses on establishing a connection between processing parameters and the resulting quality features—geometrical accuracy, surface roughness, material properties, and defect density—of the manufactured parts. A significant aspect for the digital material description is the inclusion of both the microstructure of the material and its resultant properties. The digital representation facilitates a simulation workflow that elucidates the local process–microstructure–property relationships within the as-built component, enabling an assessment of material performance variations at the scale of the part itself.

The exploration of wire arc additive manufacturing (WAAM) and wire laser additive manufacturing (WLAM) processes aims to deepen the current understanding of process–microstructure relationships for the selected steel alloys. It involves studying the evolution of solidification morphologies and the microstructural changes due to solid-state phase transformations. To this end, we employ and combine different advanced simulation tools, including CALPHAD-based thermodynamic calculations (e.g., Scheil–Gulliver calculations with Thermo-Calc^[25]), phase-field simulations (with MICRESS^[26]), and thermomechanical simulations on the process scale using a custom FEM software for laser metal deposition (LMD).^[27]

Collaboration among different research groups from industry and academia is a cornerstone of the project, facilitated by the Multi-User Integrated Computational Materials Engineering (ICME) environment, AixViPMaP,^[28] and underpinned by the PMD-Server framework (PMD-S). This setup enhances user authentication and integrates web-based tools, including cloud storage, to streamline data sharing among institutions. Also, High-performance computing (HPC) plays a crucial role in managing computationally demanding simulations, with Jupyter Notebooks^[29] serving as an interface for job definition, task execution, and results visualization, **Figure 4**.

A noteworthy achievement is the development of a workflow that bridges thermal continuum simulation at the melt pool scale with spatially resolved solidification simulation at the dendrite scale, **Figure 5**. This integration is meticulously documented

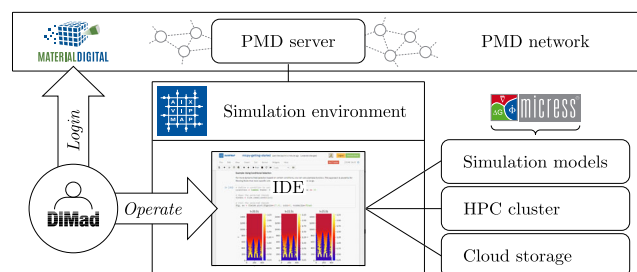


Figure 4. DiMad workflow integration within the MaterialDigital infrastructure: Users from DiMad partners leverage the MaterialDigital Single Sign On (SSO) for seamless access to the simulation environment. A PMD server serves as a gateway to the MaterialDigital network, facilitating secure and efficient operations. Within this ecosystem, the Jupyter Notebook is utilized as an integrated development environment (IDE) that orchestrates the simulation models, manages operations on the high-performance computing (HPC) cluster, and access cloud storage. This integrated platform is designed to facilitate the execution of complex simulation workflows, enhancing productivity and collaboration.

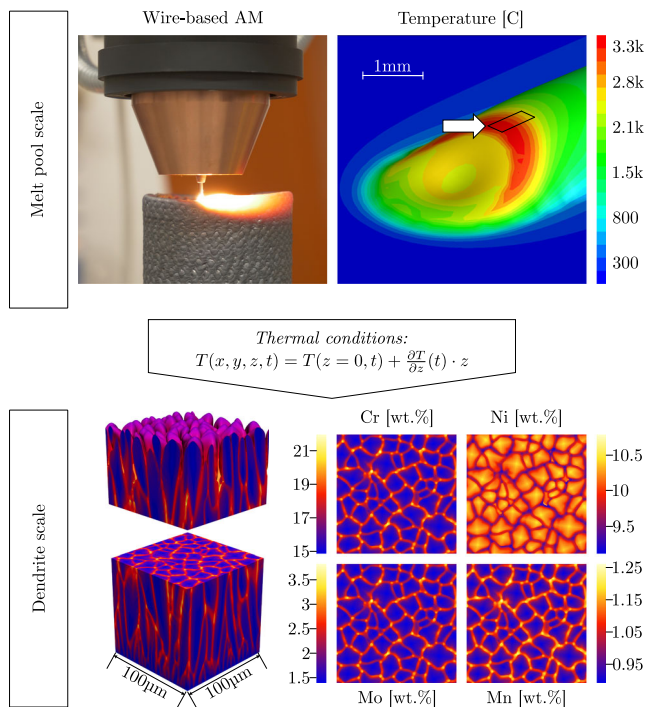


Figure 5. Top left: The photo shows the wire-based additive manufacturing process in action capturing the molten metal as it is deposited in real time. Top right: A thermal simulation illustrates the temperature distribution along a single track of the material deposition process. The color gradient on the image indicates the temperature range in degrees Celsius. Bottom left: The graphic represents a phase-field simulation of dendritic solidification of 316L. The simulation is based on the thermal conditions of the marked area in the thermal simulation above, and the varying colors denote the chromium (Cr) composition. Bottom right: This section displays the microsegregation of elements chromium (Cr), nickel (Ni), molybdenum (Mo), and manganese (Mn) on a plane perpendicular to the direction of solidification and in a part of the material that has completely solidified. The distinct color palettes in each panel reflect the concentration gradients of the respective elements, expressed as weight percentages.

in a Jupyter Notebook, ensuring reproducibility and fostering further collaboration. The sharing of the methodologies developed in this project with a broader community is achieved through the publication of a simplified workflow on the MaterialDigital Workflow Store (<https://workflows.material-digital.de/workflow/95>).

In conclusion, the DiMad project exemplifies the potential for increased automation, interoperability and reproducibility in additive manufacturing research. By leveraging the PMD-S framework, we streamline data transfer and enhance collaboration between project partners. Looking ahead, we are exploring the integration of the pyiron framework^[3] to achieve a more structured workflow management approach. In contrast to the relatively flexible nature of Jupyter Notebooks, pyiron offers specific interfaces that can provide a more standardized and tightly controlled approach to our workflow management. This could lead to an even more robust and systematic handling of our simulation processes.

3.3. DiMoGraph: Toward Efficient Models for Graphene-Based Conductor Materials

Our aim is to create and combine numerical tools to describe graphene-based conductor materials in a computationally efficient way. Very different handling of software tools, lack of documentation, and no interfaces between tools demand an improved and more automatized approach. In this use case, we develop a pyiron workflow to train a fast machine learning model to overcome these shortcomings. The required training data have been calculated using a network model of the entire macroscopic conductor material.^[30,31] With this model, a huge number of individual graphene flakes, which are arranged layer-wise, is studied using nodal analysis (See Figure 6 for a paradigmatic model system.). The electrical conductivity is then extracted as the final result. While the model is capable of extracting material-property-relationships for a large parameter space, it is computationally too expensive to allow real-time estimations. A Gaussian process regression model^[32] is our alternative approach to the network model. It is not only faster, but can give the uncertainty of the prediction. As a result, we can add new data points to the training set whenever the uncertainty is too big. The entire workflow is depicted in Figure 6.

We use the pyiron workflow manager for two purposes. First, it is used when creating the training data using the network model by orchestrating the calculations and submitting them to a high-performance cluster using the Slurm Workload Manager. Secondly, pyiron is used to perform the training of the Gaussian process regression model. Here, the fact that pyiron is python-based allows a smooth implementation of established machine learning libraries such as scikit-learn.^[33]

A specific front-end to use pyiron for our solutions has been written. The network model has its own executable, while the training workflow directly uses python functions from within pyiron. The data exchange between network model and training data is so far realized using csv-files, but more automatic approaches, using, for example, data stored in a knowledge graph, are currently underway.

In conclusion, automatization of workflows using pyiron led to a more structured and better documented work process. Predefined functionalities of pyiron simplified some typical simulation tasks, such as the submission and handling of calculation jobs on a computer cluster. The implementation of the front-end in case of the network model required significant implementation efforts, but migration from existing, already python-based scripts, was relatively smooth. The training workflow, in which pyiron and scikit-learn libraries were combined, offers an example of utilizing pyiron for machine learning purposes to the MaterialDigital community. Future work will focus on combining the creation of training data and the training itself. For example, whenever there is a lack of training data for a certain parameter combination, the network model could be automatically triggered to create the missing training data. The extension to other training data, for example, data from experiments, is planned, too. Finally, we believe that it would be worth attempting a more generalized training process such that the training workflow could be applied to arbitrary training data from other application scenarios, which are relevant within the scope of the

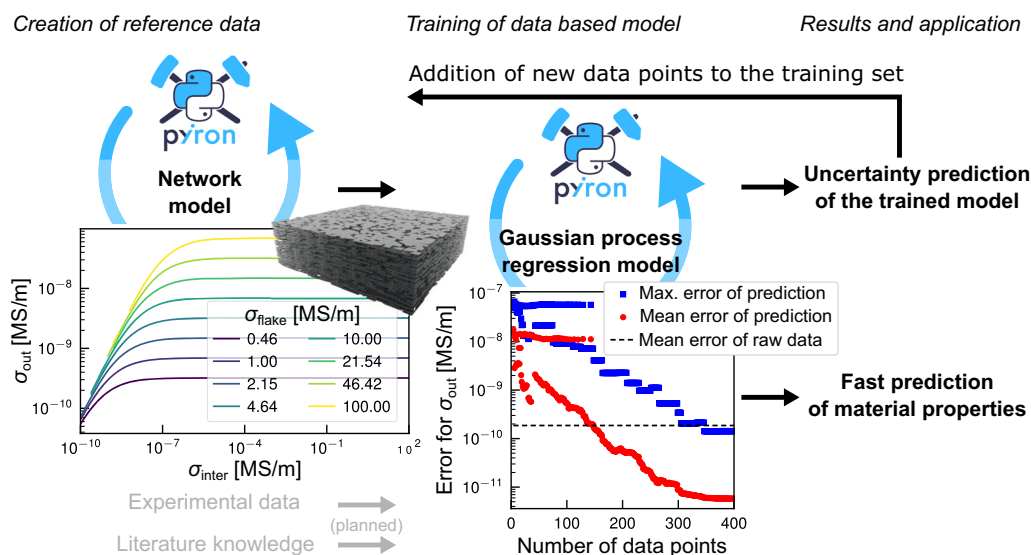


Figure 6. Overview over the key workflow in the DiMoGraph project aiming to develop a data-based model. Left: Reference data are created using a network model, which gives, among others, the relation between microscopic and macroscopic conductivities. Including also experimental data and literature knowledge is planned in the future. Middle: A Gaussian Process Regression Model is trained. The graph shows the reduction of the error by increasing the number of data points. Right: The trained model allows fast prediction of material properties, including also the corresponding uncertainty of the prediction. This uncertainty can be used when selecting new data points for the training set.

PMD. Using the workflow store, a suitable platform is available to share the training workflow with the community.

3.4. DiProMag: Magnetocaloric Materials Discovery through Automated Research Workflows

The DiProMag project aims to streamline the discovery of high-performance magnetocaloric materials. Therefore, experimental and simulation workflows are implemented with the common goal of evaluating the magnetocaloric effect (the change in entropy for a magnetic transition) of given Heusler alloys. The goal of this use case is to build two separate workflows, one experimental and one simulation-based, that start from the same knowledge point, namely a compound description, and want to arrive at the same knowledge point, namely the entropy change in the given magnetocaloric alloy (see **Figure 7**). In the future, the combination of these workflows will make it

possible to bring the experimental process of material characterization and the simulation process closer together, eventually leading to an easier combination of both parts of the materials discovery process, experiment and simulation, not only in a final academic publication but also within the discovery process itself. The complete documentation of the workflows allows both sides, experimental and theoretic physicists, to use each other's tools and data and to easily chain different simulation and analysis tools. This can only be achieved by having FAIR data and FAIR workflows that transform these data.

The overall aim of DiProMag is to identify useful magnetocaloric alloys for use in productive applications such as cooling. Therefore, the magnetocaloric effect is being modeled theoretically in order to quantify it and compare it between different alloys. Furthermore, physical samples of different alloys are measured experimentally and their magnetocaloric effect is quantified based on the experimental data.

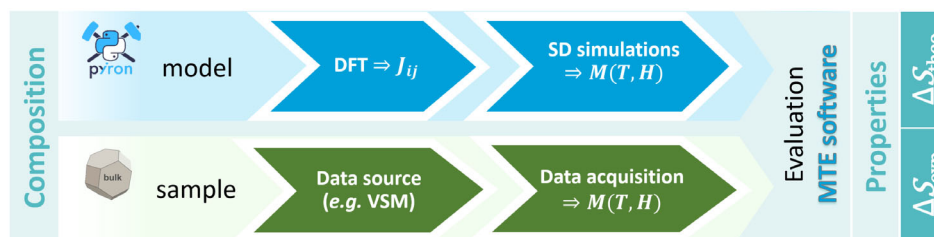


Figure 7. The two DiProMag workflows, built to combine the simulation and experimental branches. The simulation workflow (top) starts with the atomistic material model, extracts exchange coupling coefficients J_{ij} from DFT results for a spin dynamics (SD) simulation, resulting in magnetization curves $M(T, H)$ for the given material. The experimental workflow (below) starts with a sample of the described material characterized by measurement (e.g., VSM), the results of which are automatically canonicalized and recorded in an electronic lab notebook. The result of both individual workflows is a dataset that can eventually be used to evaluate the magnetic phase transition (MTE) to calculate the magnetocaloric entropy change of the material.

The motivation for developing formalized workflows is that the development of different simulation steps can be decoupled, making it easier for interdisciplinary teams to work together. Reproducibility is enhanced as workflows and their results become transferable between groups, encouraging collaborative testing and code development. Automation reduces human intervention in mundane tasks, such as data handling, allowing seamless ingestion and transformation of experimental data for use in simulations and further analysis.

Using pyiron^[3] as the workflow framework, regular tasks such as the programmatic definition of atomic structures, which have already been solved by other groups, can be done using built-in libraries and do not need to be reimplemented.

In the simulation workflow, atomic structures are initially described, and electronic structures are derived using density functional theory (DFT) via pyiron with Sphinx.^[34] The free energy is calculated, and the exchange coupling coefficients are obtained by regression fitting of an effective Heisenberg model, all without the need for further “translations” between tools, as pyiron generic data structures can be reused. This also allows to switch to other DFT tools, which are already integrated into pyiron, for example, VASP, as the generic implementation makes input and output for the same class of tools mostly identical.

A custom Metropolis Monte Carlo spin dynamics simulation code, CINOLA, interfaces with pyiron-specific features, like the data structure for defining atomic structures. All required inputs to CINOLA are derived via a wrapper from the already available pyiron data structures. CINOLA simulation results are written back to the pyiron-style data structure using HDF5.

Experimental data acquisition involves the acquisition and canonicalisation of magnetization curves from a proprietary format (from a VSM magnetometer, QD MPMS) into a centralized electronic laboratory notebook (ELN). This data is automatically transformed into a structured format based on the DiProMag application ontology and stored for easy access. Further automated workflows can access the structured data to automatically perform further analysis, for example, evaluating the magnetic phase transition to calculate the entropy change of the measurement sample.

DiProMag demonstrates the use of structured workflows to facilitate collaboration between experimental and computational physicists in the discovery of magnetocaloric materials. Lessons learned include the challenges of defining precise semantics for quantities and the limitations of using Jupyter notebooks for reusable workflows.

Precise definitions of the semantics of a given quantity can be difficult to obtain. For example, the coefficient of exchange coupling can be given with several different semantics: in several different units, but also, for example, normalized by the number of atoms or not normalized. It is usually difficult to determine from code, code interfaces, and even research publications which way of describing exchange coupling has been used or is required to use the given software. As a first approach to tackling this problem, we see the need for a widely accepted culture of explicitly annotating all occurrences of such quantities, whether as literal values or symbols in the literature or as variables or interfaces in software. Eventually, this issue should be covered

by semantic annotation using some ontology, for example, based on the PMDco.^[35]

Jupyter notebooks used for prototyping workflows and intended for interactive use may achieve reproducibility, but they are not optimal for software modularity and reuse. This is due to the challenge of distinguishing between code components that are applicable to multiple instances of a general workflow, and those that are tightly bound to the specific context of the exact workflow envisioned by the original author. Often, these components are intricately intertwined, making separation difficult. This inherent nature of Python notebooks, which typically prioritize rapid prototyping over adherence to robust software engineering practices, is a well-known problem in software engineering.

The described spin dynamics workflow is published in the PMD Workflow Store (<https://workflows.material-digital.de/workflow/76>). All other described workflow components will be published in the Workflow Store soon.

3.5. GlasDigital: Glass HT-MD Simulation and ML Modeling Workflow

Glass design traditionally relies on a costly trial-and-error approach. Using high-throughput molecular dynamics (HT-MD) simulations, the composition range for optimizing a specific property can be reduced, thus narrowing the possible compositional space for a desired property and accelerating glass development. A workflow driven and modular approach for glass MD simulations enables a seamless exchange of force-fields, adjustment of parameters for the melt-quench process, and a simple selection of specific target properties, such as mechanical and thermal properties, or glass density. The calculated properties are then used to train ML models that are able to combine experimental and simulated data in a single coherent prediction model for glass properties.

Initially, challenges in implementing the MD glass simulation workflow were encountered due to limitations in existing tools. Specifically, pyiron, a computational framework, lacked certain features, such as support for custom force-fields and simulation files. However, through the work of the pyiron development team, the necessary functionalities were integrated, enabling the future utilization of pyiron alongside a Python package tailored for HT-MD simulations of oxide glasses, which has not yet been published.

Figure 8 shows the workflow from glass composition to the prediction of glass properties with ML models. First, if no experimental density is available, the initial density of the simulation cell is estimated as the average of the crystalline glass components weighted by their mol%. Next, random glass structures are generated and treated with the melt-quench technique using the LAMMPS software package.^[36] Mechanical or thermal properties such as specific heat at constant pressure or coefficient of thermal expansion can be calculated in addition to the glass density, depending on the input flags entered. The glass properties obtained are used to train ML models, either with purely simulated data or, if sufficient measurements are available, in combination with experimental data. The models trained in this way are then able to make predictions for the particular

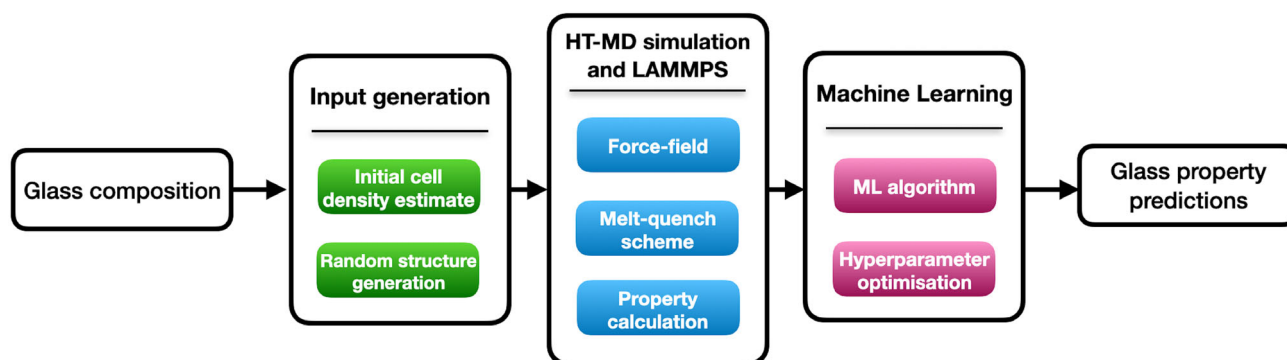


Figure 8. Schematic representation of the glass HT-MD simulation and ML modeling workflow.

combination of glass system and target property, in the case of probabilistic models even with the corresponding prediction uncertainty.

The approach presented demonstrates the potential of employing computational data-driven workflows utilizing HT-MD and ML to accelerate glass development. A key lesson learned from working with this approach is that modularity is crucial not only in efficient development but also in future applicability. In later versions, for example, it enables extensions to other force fields, the calculation of different properties, or the integration of novel ML algorithms. Moreover, another key takeaway is to facilitate the further integration and implementation of the existing pyiron framework, PMD resources, and utilize the opportunities of the Workflow Store.

3.6. iBain: Intelligent-Data-Guided Process Design for Fatigue-Resistant Steel Components Using Bainitic Microstructure as an Example

Within the iBain project the phase-field simulation software OpenPhase^[37,38] was utilized for simulation of martensite, tempered martensite and bainite. Part of the project is the inclusion of OpenPhase in a workflow tool such as pyiron. The implementation of OpenPhase as a workflow module in pyiron is done in two distinct ways, which we call the black-box method and the library method indicating the challenges and advantages of making software tools interoperable within a workflow environment.

In the black-box method pyiron is used as way to create input files for OpenPhase together with an OpenPhase executable that allows the simulation of martensitic transformation. This way the user can only modify process parameters such as cooling rate, holding time, and holding temperature. This black-box approach is easy to use without knowledge of technical aspects of phase-field simulations and easy to maintain. However, it severely lacks flexibility. Therefore, pybind11^[39] was used to create python bindings of the relevant OpenPhase functions and classes that are used in martensite and bainite simulations. However, this approach requires significant knowledge on the part of the user, as they need not only knowledge of the physical processes, but also of the numerical aspect of the phase-field simulations and a base knowledge of OpenPhase itself. However, using the full flexibility of the python bindings, it is possible to add

on-the-fly microstructure analysis and to adapt process parameters. As an example, the OpenPhase module has been combined with an image analysis tool and a fatigue lifetime prediction tool, both developed at ICAMS.

The image analysis tool evaluates morphological features based on 2D slices of the 3D output data of the phase-field simulation. This allows comparison with EBSD data from experiments. The morphological features are then passed into a Kocks–Mecking-type model for the estimation of the fatigue lifetime.

Using this in a workflow schematically depicted in **Figure 9** enables optimization of process parameters, for example, the cooling rate and holding time and temperature, in order to increase the fatigue lifetime.

Working on the workflow design for the iBain project, which required the integration of a number of different tools, showed that such a task is not always easy or straightforward. Different software interfaces, different data formats and different work-principles of different tools make it necessary to use a common environment to facilitate their inter-operation.

Continuing our workflow design started under the guidance of the PMD the python bindings of the OpenPhase will be expanded to cover most of the library and the scope of the pyiron module will be expanded beyond martensite simulations. The resulting generalized workflow will be deposited to the PMD Workflow Store. Such an extension of OpenPhase will facilitate rapid material's development by computer-aided design and will significantly reduce the efforts to incorporate computer simulation tools into the conventional material's design practices.

3.7. KNOW-NOW: SimStack Workflow for DEM Script Jobs

The satellite project KNOW-NOW is centered on exploring ceramic materials. In ceramics, understanding how powders behave during critical processes such as mixing, grinding, and pressing is crucial. To achieve this, the discrete element method (DEM) serves as a powerful tool to model the behavior of the powder at the particle level. However, the challenge arises with the prevalent DEM libraries, which are script-based and can be daunting for those unfamiliar with coding. To bridge this gap, we present an innovative approach that leverages the graphical user interface of the SimStack workflow environment. This

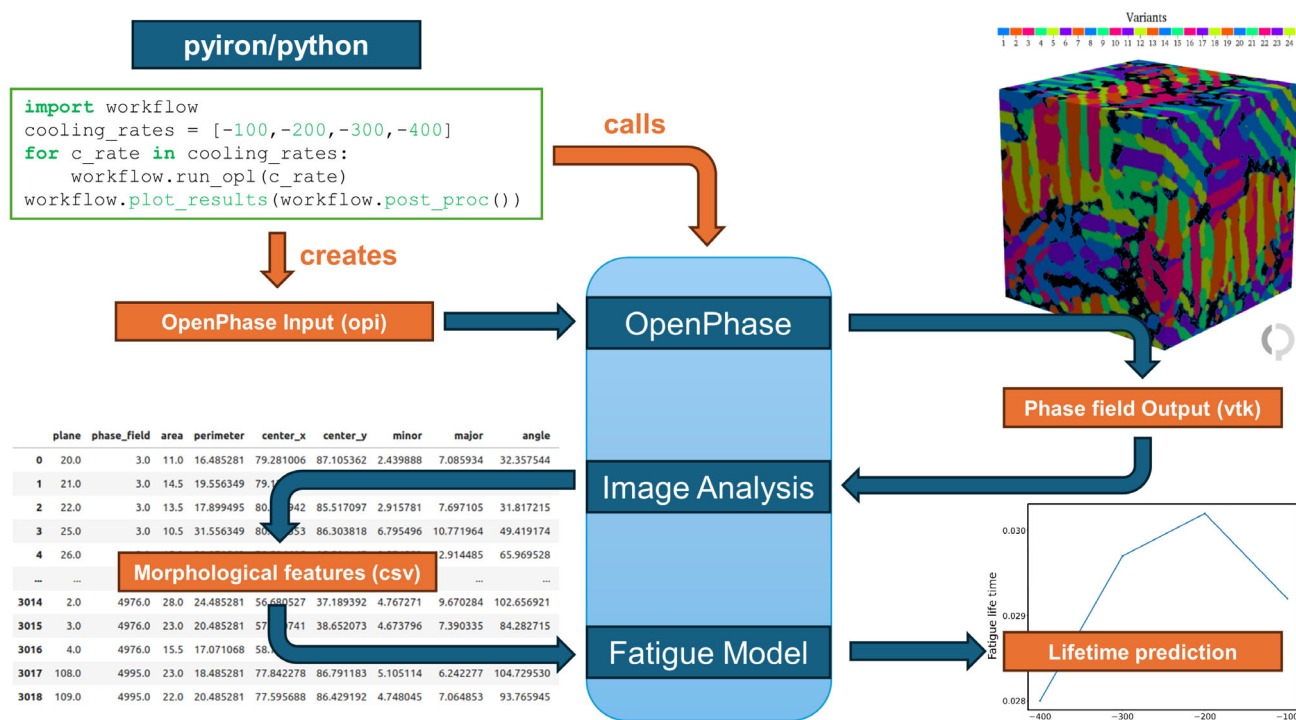


Figure 9. Representation of the workflow for the fatigue lifetime prediction. OpenPhase simulates the martensitic transformation and passes the resulting microstructure to an image analysis tool, which evaluates the morphological features of the microstructure. This data is then used in the lifetime prediction tool.

approach simplifies the simulation of complex, technical issues like powder pressing. Utilizing a SimStack workflow, we could effectively conduct and analyze powder pressing simulations with the open-source DEM code LIGGGHTS, making it accessible for users without coding expertise. The goal was to streamline generating input scripts, running simulation jobs, and analyzing results. At the heart of the workflow we discuss here is the LIGGGHTS WaNo, which automates the creation of LIGGGHTS input scripts. It executes a Python script that adjusts specific variables within a predefined template. This template includes all the commands and parameters for a LIGGGHTS simulation, with key variables such as particle size and number, timesteps, and programmable dump rate. These variables are made accessible to users through a GUI in the SimStack client, enhancing usability and interaction with the LIGGGHTS WaNo. This approach enables users to set their preferred values and easily create tailored LIGGGHTS scripts. The process is streamlined by the SimStack client, which automates the transfer of LIGGGHTS jobs to a designated HPC server. Once the pressing process calculations are completed on the server, the resulting data is sent to the DataDive WaNo for analysis. This data includes detailed information on each particle's position, and the forces exerted on the pressing plates at every timestep. Utilizing this information, the DataDive WaNo calculates a stress-strain curve, considering the contact pressure defined by the user. Typically, a single stress-strain curve is derived, but the system's advanced looping capabilities and the SimStack client's ability to leverage previous results allow for the efficient modeling and calculation

of an entire test series in one go. To expedite the analysis of large series, the DB-Generator WaNo compiles the results of each loop into a single database file. It also allows users to upload this file directly to a GitHub repository, ensuring the results are readily and widely available for collaborative work, such as in Jupyter notebooks or Google Colab. This workflow is exceptionally beneficial for the rapid prototyping of script jobs, allowing for quick testing and development of the script template used in the WaNo. An example workflow depicted in **Figure 10** demonstrates the application of these WaNos to examine how specified contact pressure affects stress-strain curves. The workflow is available at the PMD Workflow Store (<https://workflows.material-digital.de/workflow/91>) and GitHub (<https://github.com/BjoernMie/KNOW-NOW>).

3.8. KupferDigital: Data Ecosystem for an Ontology-Driven Material Development of Copper Alloys

The central challenge of the project KupferDigital is to express and describe simulation and experimental data and process knowledge along the entire lifecycle of copper materials and across all involved domains and company boundaries, each with their own highly individual tools. Moreover, all data and pipelines had to be delivered in one common data ecosystem.

To achieve the project goal within the project funding time of three years, that is, to be able to at least in a fundamental manner represent all domains of the copper lifecycle ontologically, at the beginning of the project the following requirements to workflows

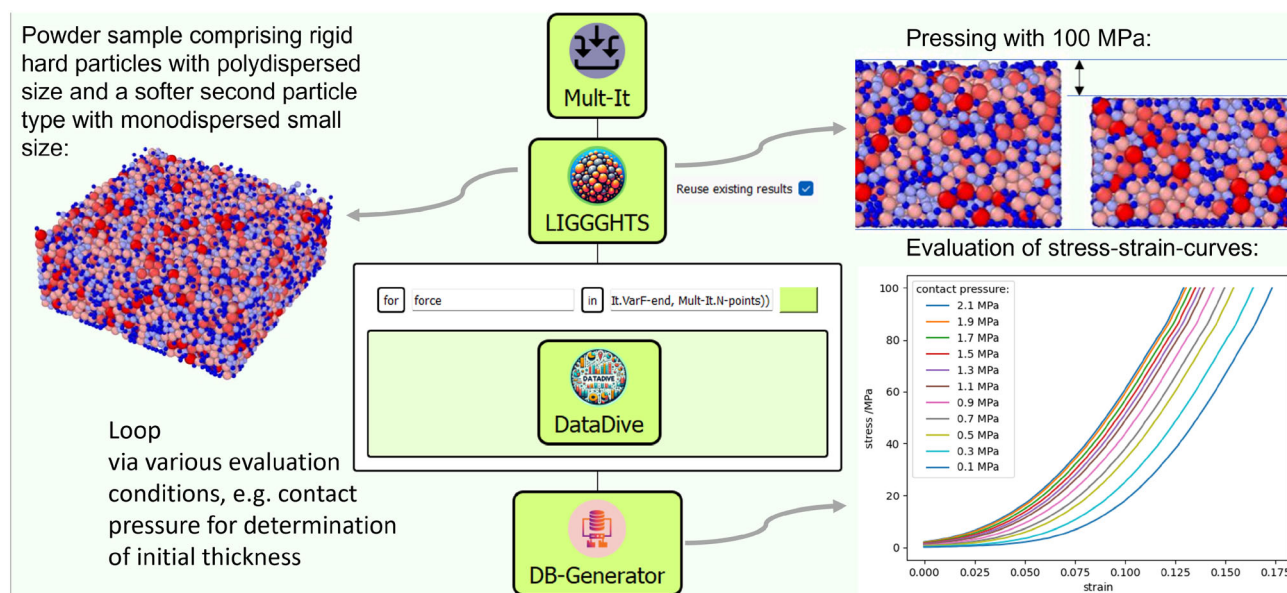


Figure 10. This SimStack workflow integrates four WaNos within a loop to investigate the impact of different contact pressures on the stress–strain outcome of a discrete element method (DEM) simulation of powder compression. The process begins with constructing a powder sample containing rigid, polydispersed, and softer, monodispersed particles. This assembly is compressed under a 100 MPa force, utilizing LIGGGHTS for DEM simulations, Mult-It for iterative processing, DataDive for data examination, and DB-Generator for database compilation. The computed stress–strain curves highlight the material’s deformation characteristics in response to varying contact pressures.

were identified as crucial: 1) Rapid Integration of all Consortium Partners: Existing data workflows of the experts, with the properties mentioned in the Section 1, must all be reliably executable within the first year. 2) Use of Existing Environments: Since the workflows of the partners have individual requirements regarding licenses, libraries, interfaces, or runtime environments, the data must be able to run in the environments in which they were developed or used. 3) Technical Feasibility of Tool Consolidation and Scaling: The solution must be suitable for migration or scaling without problems or restrictions, for which a container-based approach was chosen. 4) Interoperability with the Comprehensive Knowledge Archive Network (CKAN) Data Space Solution of the Data Ecosystem: Another requirement was that the workflows for data transformations and processing must be executable within the CKAN instance. The robust and extensively tested community solution Nextflow seemed best suited for this and was therefore chosen.

These requirements show that due to the size and number of domains involved and their demands, PMD workflow concepts with their strengths, particularly the central and generic implementation, including the provision by PMD, could not yet be utilized in the relevant project stage to achieve the project goals of KupferDigital. However, for the long-term goals of follow-up projects or the upcoming expansion of the data ecosystem, the re-use of the projects results was always valid. Iteratively in the project lifetime, the integration with the PMD was discussed and at a semantic level included into the ecosystem.

By the developed concepts the KupferDigital project overcomes central problems in material development and the circular economy by using a collaboratively developed technology stack in combination with commonly utilized top- and mid-level

ontologies. The project realizes a digitalized alloy screening method (based on the work published in [40]) that combines experimental and simulation data as a demonstrator process with the goal of evaluating and measuring the suitability and quality of the workflow concept at the end of the project. The data catalog is realized with CKAN.^[41]

Addressing the copper life cycle involves balancing technical, economic, and scientific factors, necessitating interdisciplinary collaborations and the exchange of material data across domains. Developed methods should abstractly process varied data formats from multiple scientific fields to facilitate collaborative efforts across networks. Therefore, workflows in KupferDigital must operate without constraints in decentralized environments, handle diverse data types, ensure interoperability, maintain security to open standards, and exhibit scalability for performance in large data volumes. Key future challenges for the application of workflows include secure data transfer, correct access rights, and security mechanisms during decentralized use. Partially realized goals include traceability and comprehensive logging for result verification.

The alloy development process, significant in the project since it involves all given main aspects, integrates existing tools from different domains, requiring seamless workflow execution across varied environments, from personal computers to cloud systems, see **Figure 11** where the semantic connections between subprocesses are drafted. KupferDigital leverages Nextflow^[42] and the minipod^[43] container system for workflow management, supporting diverse scripts and ensuring scalability and sustainability.

The process is also represented as a Knowledge graph (KG), which itself is graphically expressed with drawio and provided to

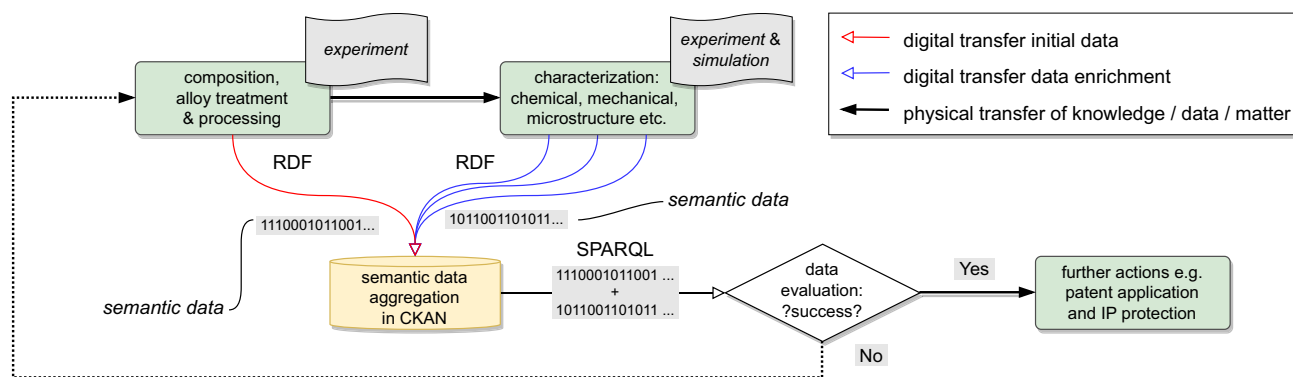


Figure 11. Workflow in the KupferDigital project for the alloy development process via screening based on semantic data. All data from all domains were converted into their semantic representations using individual pipelines with different programs or scripting languages. The pipelines were integrated as add-in into CKAN.

the workflows with Chowik.^[44] The diagrams are maintained in a GitLab repository. As the execution environment, a comprehensive data processing pipeline is used, orchestrating the transformation from source files to a KG. This is conducted using Nextflow. Each discrete step within this workflow is exclusively encapsulated within dedicated containers, ensuring modularity and reproducibility. These encapsulated processes include the conversion of diagrams to RDF, mapping using the RDF-Diagram-Mapper, merging RDF files, reasoning, and the subsequent publication of RDF data to both a triple store and Fuseki.^[45]

In response to any modifications, a continuous integration job is automatically triggered, initiating the execution of the Nextflow workflow, thus ensuring seamless adaptation of the pipeline to evolving requirements. Using semantic technologies and containerization, the exchange of specific tools is less of a problem and the technical environment is set which does not introduce new technical conflicts over time per se. Additionally the access to the workflows and the data is restricted and secured via HTTPS and SSH communication, thus the requirements for data protection are fulfilled.

As part of the workflow environment, a CKAN instance and a triple store are used to store and provide the KG. One interface of

the KG is the tool Sparklis.^[46] The deployment specifics of these components are meticulously documented using minipod, guaranteeing a standardized and reproducible setup for the workflow. This robust infrastructure underpins the efficiency and reliability of the entire workflow, from data processing to the construction of the KG. **Figure 12** represents what has been mentioned.

The generic interfaces and technologies used in PMD and KupferDigital lead to an inherent compatibility, the focus on simulations in real environments of the PMD activities can be enhanced with the generality of Nextflow workflows and the available semantic data. Notable among these technologies are DCAT,^[47] CKAN, RDF, and SPARQL.^[48] Especially DCAT and RDF in use in the copper industry, is a step toward the industry adoption of semantic technologies and thus PMD related efforts. Instead of a static tool list, the KupferDigital project, concurrently with developments in PMD, advanced alternative solutions that are interchangeable or integrable through the named interfaces and technologies. Being specific, pyiron can easily be integrated into or in addition to Nextflow workflows, and the output data can be expressed with RDF and stored in CKAN as one example. Another possible approach for integrating the projects results in PMD technologies is to use the semantic outcome (data

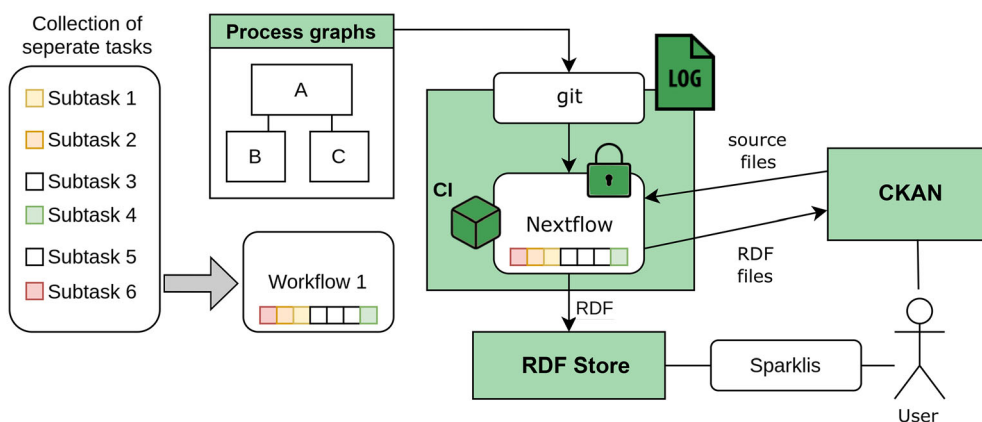


Figure 12. Overview of workflows and connected systems. It shows that a workflow consists of multiple steps and how the systems are connected.

and services) as a base for workflow input data and workflow internally used interfaces. The reconciliation process took place in a targeted manner through various cooperations and is especially visible at the data catalog level based on CKAN. The reason is that CKAN is expected to play a central role in the future PMD efforts and also that CKAN, as a widely used data catalog, is built to be used in a distributed environment. As all of this cannot be a replacement for PMD solutions, it was heavily influenced by a specific restriction at the start of KupferDigital, as mentioned at the beginning of this subsection. In the context of contributions to the MaterialDigital workflow activities, lessons learned from the use case highlight the essential role of connecting workflows or their triggers to GUIs for potential utilization.

The successful combination of respective work steps into workflows, addressing previously identified challenges, underscores the effectiveness of execution environments. In particular, DCAT and CKAN are expected to play significant roles in future collaborations and projects. Similar to the DiProMag project, it was beneficial to have precise semantics when working with, for example, quantities in the modeling phase. Furthermore, the generic usage of workflows, as well as the transparency of the workflow executions, the high usage of semantic technologies, the low barrier for defining plus mappings graphs and the central data catalog, are strengths of KupferDigital and show the relevance of the projects results. Using a main unit system, modeled in RDF, additionally increased the adoption to our workflow technology.

As a technical review, the Workflow Management System supports complex, scalable data workflows by encapsulating steps in processes and using channels for advanced data flow, enabling parallel execution and adaptive patterns like branching and looping. It ensures reproducibility and portability across diverse environments through containerization, functioning seamlessly on laptops, HPC clusters, and in the cloud without modifications. The KupferDigital workflows have advanced significantly in involving domain experts with minimal or no prior computing experience. However, barriers to entry remain, which require that a significant portion of efforts be dedicated to training partners and developing new tools. The suitable application of dynamic knowledge graph management and the usage of git in workflows was similar to the LebeDigital project. The combined use of Nextflow and DSMS proved to be a viable solution also utilized within the StahlDigital project. Regarding the evolution of underlying ontologies, our approach to model especially the process graphs in a usable GUI while providing the namespaces and identifiers enables and encourages updating the workflow artifacts to new versions of the ontologies.

Regarding the challenges faced, the application of Nextflow in continuous integration settings on Gitlab presented accessibility issues for non-developers. The development of a specific toolset, using minipod, for deployment on server infrastructure was a strategic response to ensure working environments and the fulfillment of technical requirements.

Future work in KupferDigital emphasizes a strategic pivot from direct publishing to triple stores toward constructing a knowledge graph based on the DCAT Catalog with an assembler file, controlled by a CKAN-integration. This approach advocates for the separation of workflow definitions from tools and source files, currently consolidated within a

monorepo, to enhance modularization. From the project perspective, the reuse for PMD should center on enabling more workflow environments and making use of semantic data and technologies. The insights of the project highlight the need for the adoption of specified generic interfaces and technologies, enhancing interoperability and integration within the digital ecosystem.

3.9. LebeDigital: Optimizing the Process Chain for the Production of Concrete and Concrete Structures

The aim of the LebeDigital project was to optimize the production process of concrete and concrete structures.

This use case includes on the one hand the structuring and storage of all relevant data (e.g., for an optimized mix design, a quality control) and a parallel modeling approach to build a simulation model that captures all the relevant phenomena and is able to combine material and structural design in a single workflow. For demonstration purposes, the design of a precast concrete beam was investigated to minimize the global warming potential by changing the mix design (replacing portland cement with slag) and the height of the structure (as another optimization parameter). The coupling of different simulation tools ranging from simple analytical expressions over design codes (EC2) up to complex FEM simulations that compute the spatial and temporal evolution of the load bearing capacity of the structure are included. These models are parameterized (e.g., require material parameters such as Youngs modulus or parameters describing the rate of hydration). These are either calibrated using Bayesian methods including uncertainties or are learned via ML-methods. A detailed description of the demonstrator is given in [49].

The workflow related to the computation of the key performance indicators is illustrated in **Figure 13**.

This workflow was then used first in a calibration loop to identify the material parameters or train the ML models, and then—in a stochastic optimization loop—to identify the optimal set of parameters. After having compared several workflow engines,^[50] the workflow was implemented using snakemake^[51] with individual python scripts each representing a function/module. The advantage of using hierarchical workflows in this project was clearly the ability to combine the expertise of individual contributors that were able to use the work of others as a black box with well-defined interfaces and the ability to easily install the compute environment in a reproducible way on their machine. In order to enforce a consistent usage of units, the interfaces where defined with corresponding units and an automatic conversion was realized using the python package PINT. The latter was sometimes a challenge, since, for example, our FE-simulation tool FEniCS^[52] requires a linux operating system. Just providing a pure conda-environment was not sufficient—the usage of container technologies was difficult both from an administration point of view (docker is not allowed due to admin permissions) and from an implementation point of view—the setup by colleagues that are not using these tools on a regular basis was not straightforward.

A precondition for using calibrated simulation models is the availability of training data that was planned to be queried from a

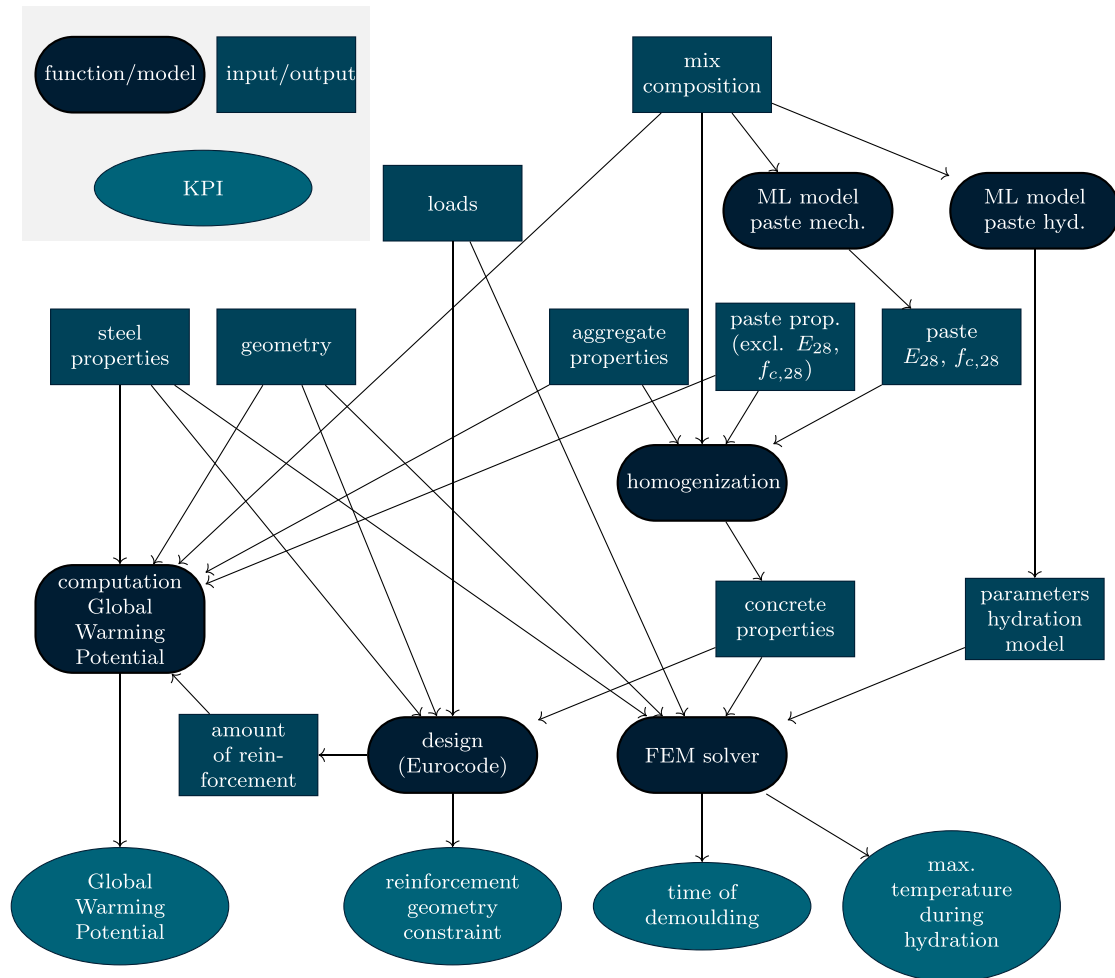


Figure 13. Workflow to compute KPIs from input parameters.

database (i.e., a knowledge graph). The generation of the knowledge graph is a very complex task due to the fact that data of different characterization techniques (e.g., compressive strength, Young's modulus, calorimetry) performed for different processing steps (e.g., related to the age of testing) and different mix compositions (amount of slag) have to be connected in a joint knowledge graph based on standardized Excel sheets as input for each individual test/step. This task was also planned to be automated using hierarchical workflows, such that on performing an additional test the data is incrementally and automatically added to the knowledge graph. In order to build the final workflow, we developed a minimum working example that we decomposed into separate steps that were coded incrementally and combined to a workflow using pydoit.^[53] This workflow was executed in a continuous integration pipeline in git for small sets of data. This allowed us to identify for each merge request potential challenges in the definition of interfaces. In addition to the complete workflow, separate tests for each individual module/script were included. The workflows were always executed in the CI-pipeline or from the commandline and planned to be used in a hierarchical way, thus we did not use interactive tools such as jupyter notebooks.

A challenge in our project was to develop a common philosophy of how git and the CI-pipeline and the minimum working examples were to be used, resulting in the end only in a very limited number of people actively pushing that approach forward. In the retrospective view, this was on the one hand related to insufficient training for colleagues without a coding background and on the other hand the ambivalence between writing good reproducible code and measureable scientific outcome, for example, in terms of papers. In this context, a platform such as the PMD-S that would have allowed to execute all these workflows in a common environment was unfortunately not yet available. Some very compute intensive jobs (in particular, the optimization and calibration) were then performed on an HPC in a partner university, resulting in those parts to be missing in the CI pipeline.

Another challenge was related to the data transfer and acquisition pipeline. In our point of view, a structured list of excel sheets is a good starting point, but that structured differed for the partners (for historic reasons) requiring individual extraction scripts for each partner. This was realized using intermediate json files that contained the relevant information for each experiment. In the future, this information should come directly from

an ELN system, and thus there is a need to integrate tools such as elabFTW^[54] or openBIS^[55] into these workflows to generate the knowledge graphs.

From our perspective, setting up a minimum working example that is integrated in a CI-pipeline really supported the joint development and the understanding of interfaces. Teams should dedicate time (e.g., in workshops) to jointly agree on these interfaces and also ensure that reviews of the code are done by the complete team and not only technically experienced single users. These workflows also helped document the work in case of fluctuating staff. In particular, when tools with additional challenges (HPC, commercial tools with licensing issues, dependency on specific operating systems) are used, it is advisable to generate a platform that allows everyone to execute the complete data processing pipelines in the project in a central location.

3.10. ODE_AM: Data Workflow Through Additive Manufacturing Process Life Cycle

The following case study illustrates an example of workflows in the additive manufacturing process, highlighting the use of ontologies for data management across both experimental and simulation workflows. A finite element simulation coupled with a regression model is used to predict the mechanical properties of additively manufactured parts, using thermal measurements. Concurrently, an experimental workflow is used to measure the mechanical properties and validate the simulation results.

There are numerous challenges in the additive manufacturing process that require structuring and formalizing workflows to achieve both automation and reproducibility. First, multiple workflows are required to capture the required material properties; for example, in Figure 14, both experimental and simulation

workflows are utilized to measure and predict the relation between process parameters and the final product's mechanical properties. Multiple workflows generate huge amounts of heterogeneous data throughout the life cycle of the material, from the raw material of the additive manufacturing process to the final product. Second, the geographical distribution of different additive manufacturing processes, mechanical testing, and simulation presents obstacles to data and knowledge sharing. Finally, there is a lack of personnel knowledge due to the multi-domain knowledge required to simulate the additive manufacturing process accurately.

Within the ODE_AM project, both experimental and simulation workflows have been utilized to measure and predict the mechanical properties of additively manufactured steel structures and to infer the relation between additive manufacturing process parameters and the produced mechanical properties. Figure 14 shows workflows for the Directed Energy Deposition (DED-arc) process, where a simulation workflow that utilizes FEM thermomechanical simulation to predict thermal history in the manufactured part and uses the thermal history data and the corresponding mechanical testing data set as input for a regression model to predict the mechanical properties of the part. A parallel experimental workflow that uses mechanical testing to measure the mechanical properties of the manufactured part for validation purposes is implemented. A semantic layer based on an ontology has been used to organize and manage data through workflows. Python scripts have been used to import data from simulation and testing reports to the semantic layer, while SPARQL queries have been used to export the required data for different steps in the workflow. A web app is developed to facilitate user interaction with the semantic layer for data import and retrieval.

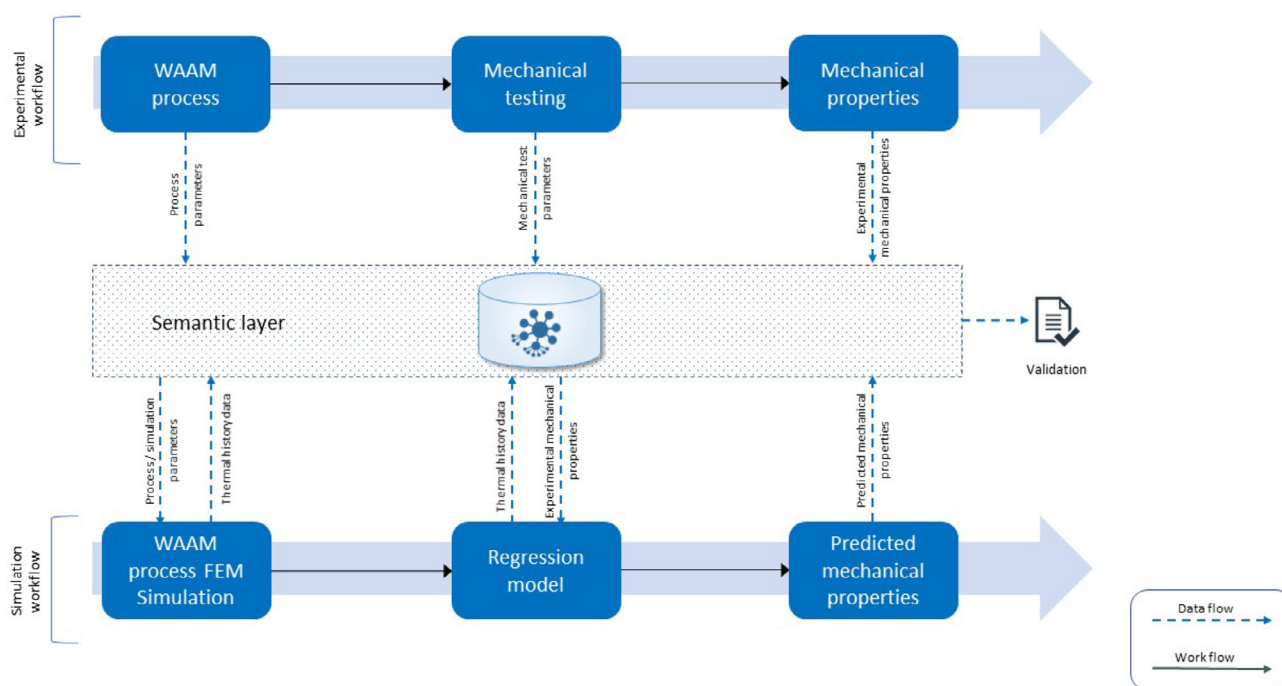


Figure 14. Experimental and simulation workflows of the Directed Energy Deposition process (DED-arc).

This use case can thus be summarized into the following four main points: 1) The integrated approach of combined FEM simulation and a regression model or ML model shows a successful prediction for mechanical properties of additively manufactured parts, which could reduce the cost of experimental tests and accelerate the optimization of manufacturing process parameters. 2) A semantic layer based on an ontology is an efficient data management solution for data management between multiple workflows. The semantic layer ensures a formal representation of workflow metadata, which in turn ensures automation and reproducibility of workflows. 3) Adoption of workflows with interaction with ontologies confirms the MaterialDigital goal to achieve the FAIR concept of data where the workflow is well documented and also to ensure the availability of workflows to be used by single users rather than the requirement of technical experts. 4) Separation of the entire workflow pipeline into modular workflows to meet the needs of different projects may be part of future work to facilitate flexibility.

3.11. SensoTwin: Sensor-Integrated Digital Twin for High-Performance Fiber Composite Applications

The project SensoTwin aims to develop a digital representation of fiber-reinforced plastics utilized in the rotor blades of modern-day wind turbines. These blades are designed for a lifetime of 20–25 years. At the end of this period, a large amount of rotor blades is decommissioned without further evaluation, although their structural health would still allow for an additional operation time. By modeling the manufacturing process (including process-induced defects) as well as the blade's structural operation, a remaining lifetime prediction with an increased accuracy

shall be implemented. The workflow described herein models the life span of a single blade of a wind turbine until its end of service and is depicted in **Figure 15**.

In the scope of the project, the most commonly used manufacturing process vacuum assisted resin transfer molding (VARTM) for glass fiber-reinforced polymers (GFRPs) in modern-day rotor blades^[56] is included. Herein, dry fibers are placed in a mold, packed in an air-tight vacuum bag, impregnated with liquid resin, and cured at elevated temperatures. Due to the polymerization of the matrix during curing and different coefficients of thermal expansion of the constituents, residual stresses evolve in the heterogeneous material after cooling down to room temperature and can negatively influence the material's mechanical performance. This process is implemented in the workflow by means of numerical multiscale models and is based upon experimentally determined time-temperature curves and the micro- and mesoscale architecture of the composite. Manufacturing defects can additionally be defined by the user in the form of, for example, local fiber orientation misalignment, ply waviness, or stacking irregularities. The process simulation results are mapped onto a structural finite element (FE) model of the blade. The material stresses during operation can be calculated based on real-world wind data. Therefore, simulations for varying wind speeds are carried out while imposing aerodynamic, gravitational, and centrifugal forces onto each simulation. The continuous sequence of varying stresses can be converted into blocks of equivalent constant amplitude stresses using the rainflow-counting method (described in, e.g., [57]). The material's fatigue behavior (determined in experiments as SN-curves or perspective also in microscale simulations) is subsequently used to determine the partial damage in every load block. Summing over all partial damages in all load blocks until the current point in the

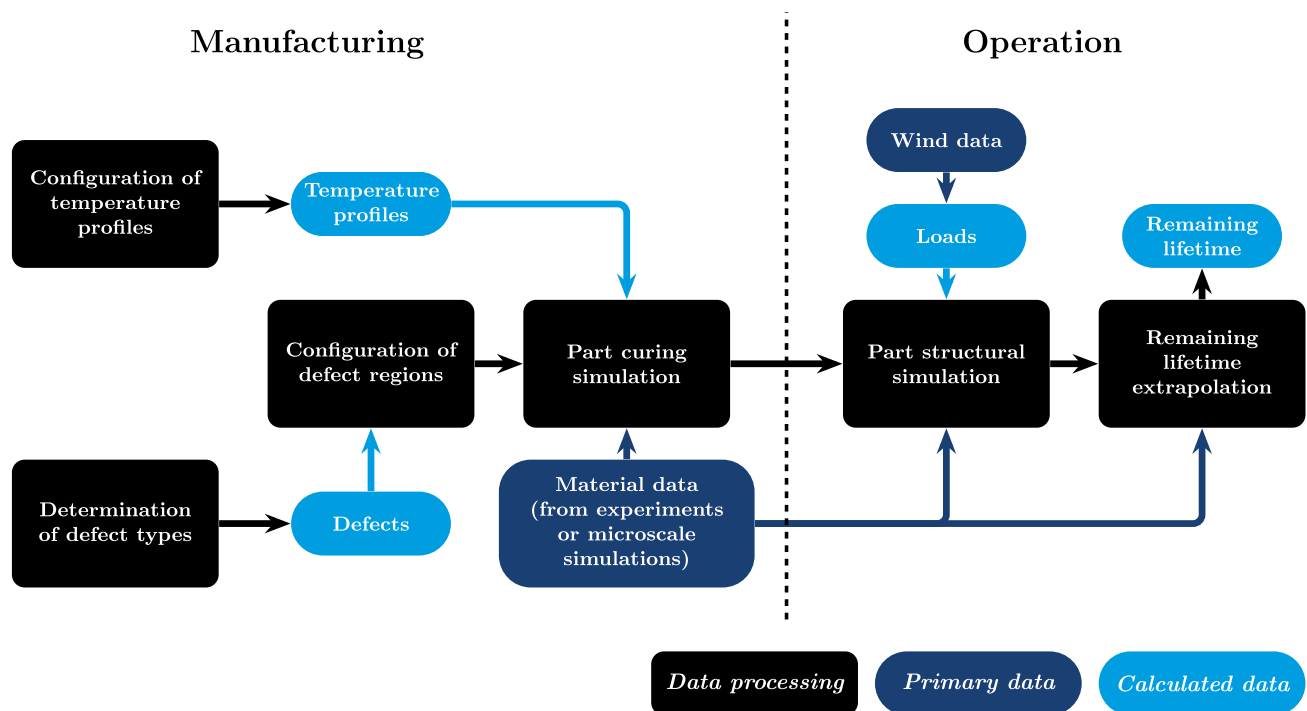


Figure 15. SensoTwin workflow modeling the life cycle of a wind turbine rotor blade from production to expected failure.

time of evaluation ultimately leads to the remaining lifetime at every material point of the model.

The technical foundation of the developed workflow are Jupyter notebooks with Pyiron workflows. Every individual step has its own GUI inside a notebook that is created via the Jupyter widget library `ipywidgets`.^[58] Representation and interactivity with 3D objects within the notebook is achieved via the `PyVista`^[59] library and the workflows connect to our ontology via the `owlready2` python package.^[60] A vital point of the workflow is using the proprietary software `Abaqus`^[61] from Dassault Systèmes to numerically re-stage applied temperature profiles to create the resulting process-dependent material properties and possible defect types for the main workflow itself. Open-source alternatives, incorporating, for example, `FEniCS`^[52] and `CalculiX`,^[62] are currently evaluated, and moreover suitable models depicting the effects of the curing process are developed.

The introduction of interactively rendered 3D models and the automatic generation of input files eliminates a time-intensive and error-prone step from a previously manual process. Beyond improved visualization of interim results, the developed workflow also overcomes the requirement to be familiar with specific input formats of the used simulation software and thus broadens the applicable audience of users.

One of the major pain points of the project was the necessary combination of propriety third-party software into this single workflow, for example, the simulation of the material curing process used the commercial plugin `COMPRO`.^[63] Due to version conflicts, using two different versions of `Abaqus` in a single workflow was necessary. The initial plan of distributing different containers with `Abaqus` that interface with the developed software proved workable but unwieldy in addition to the additional effort that licensing entails.

In line with the spirit of the PMD concerning open software and data, an equivalent solution using `CalculiX` is currently being evaluated after the initial scientific results were validated using `Abaqus`. While the upfront development cost in terms of time for a simulation using `CalculiX` is significantly higher, avoiding being locked to specific versions of closed-source software solutions is a worthwhile goal in the long term and is a point of focus for `SensoTwin`. The developed workflow is shared with the wider material science community via the PMD workflow store (<https://workflows.material-digital.de/workflow/92>).

3.12. StahlDigital: From Steel Sheet to Crash Safety with Innovative Digital Strategies

The `StahlDigital` project builds new digital strategies for the design and optimization of steels through ontology-based interoperable workflows. In this way, the multiscale and multiphysics character of steels development process is addressed, which requires the combination of data from various computational and experimental methods as well as the combination and interoperability of different existing simulation tools. The methodological background and performance of this approach are demonstrated here for the example of a multiple rolling simulation of steel sheets with `DAMASK`, which is realized in the `pyiron` workflow environment.

Such a `DAMASK` simulation needs computational input parameters (e.g., grid), process parameters (e.g., load, rolling speed), and material-specific information such as the elastic and plastic properties of the material. The latter parameters can be obtained with an atomistic code like `LAMMPS` (or a DFT code like `VASP`), unless they are taken from a tensile test experiment. Hence, we need to combine two different software tools, of which the output from one tool/method is provided as an input to the other. This prototypical example of linking software solutions from different communities is one of the workflow contributions of `StahlDigital` to the PMD.

To this end, we use the workflow environment `pyiron` and its different sub-packages (cf. Section 2.3.1) for different software requirements. The `pyiron_base` package manages workflows, contains a hierarchical storage interface based on `HDF5`, and supports HPC computing. Similarly, various atomistic simulation, continuum simulation and experimental data analysis tools are available in the `pyiron_atomistic`, `pyiron_continuum` and `pyiron_experimental` packages, respectively. Correspondingly, we integrated the multi-physics crystal plasticity simulation package `DAMASK` in the `pyiron_continuum` sub-package. All required tools for `DAMASK` simulations, starting from the preparation of input files, different grid solvers for the simulation, to post-processing of outputs are available via the `DAMASK` python package. The user can now setup, run, and analyze a `DAMASK` simulation via `pyiron` wrapper calls, which under the hood convert user instructions provided in generic `pyiron` terminology to `DAMASK` specific instructions. Once the simulation is finished, the important information related to the simulation is either parsed and stored in the `HDF5` file of the `pyiron` job or remain in the `HDF5` file of the `DAMASK` calculation.

There are various atomistic codes (e.g., `VASP`, `SPHInX`, `LAMMPS`) available in the `pyiron_atomistic` sub-package that are combined with a `pyiron ElasticMaster` job to calculate the elastic parameters of a material. We have reused this job class for the purpose of `StahlDigital`. We have also developed and integrated the `TensileTest` job in `pyiron`, which takes tensile test experimental data as input and returns elastic and plastic parameters of the material. The availability of these different codes/methods within the same environment makes it possible to interoperably combine them to new workflows. As shown in **Figure 16**, the user can exploit the different functionality and design the workflow by providing minimal instructions/information via a Jupyter interface. All steps associated with data transfer and data/metadata storage are taken care of internally by `pyiron` and shielded from the user. All important data/metadata related to the entire simulation process are stored by `pyiron` to a `HDF5` file, and are available to the user via the same Jupyter interface. As a result, the workflow replaces all the incompatible and non-transparent scripts that are commonly used for data processing and simulation execution, and which limit reproducibility and impose knowledge loss in the process.

Another goal of `StahlDigital` is to map the steps of the `pyiron` workflow into an ontology to further enhance reproducibility. A prerequisite for additionally ensuring interoperability is that this mapping is consistent with the rules that are defined by the Platform MaterialDigital core ontology (PMDco). To achieve an application ontology for the workflow, we first create a meta-data table for the `pyiron` simulation environment with different

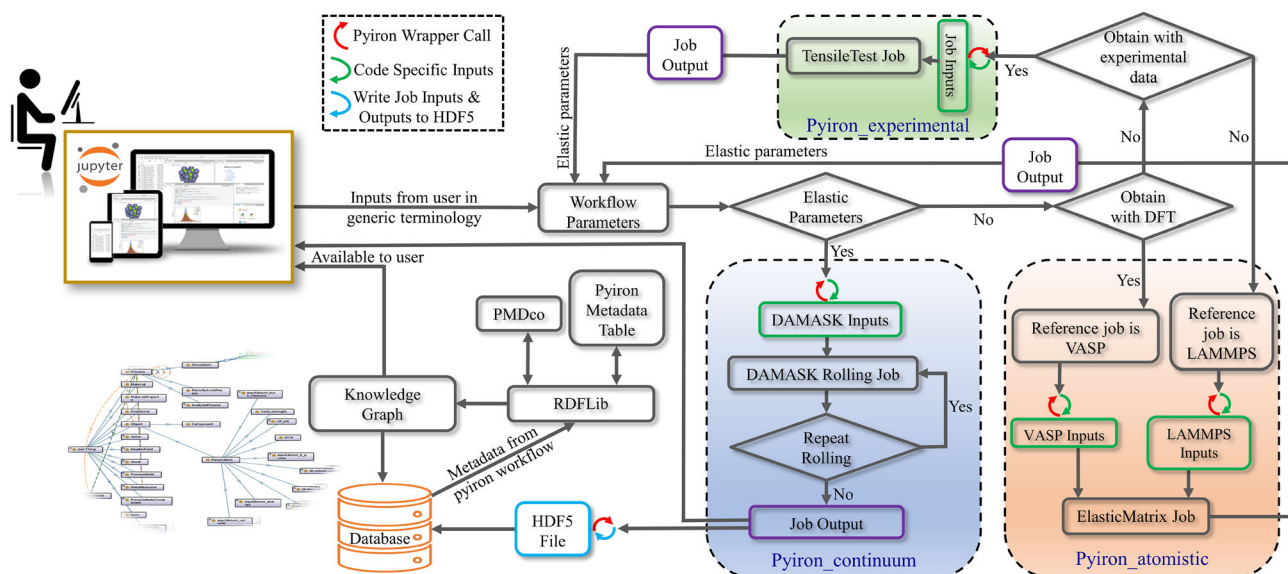


Figure 16. Schematic representation for the integration of DAMASK into pyiron workflows, as used to perform multiple rolling simulations and to derive a knowledge graph of the entire simulation workflow from pyiron metadata.

sections on (i) codes, (ii) the input and output parameters, (iii) the reference jobs and (iv) the physical units. The metadata table is then mapped to the PMDCo in class and subclass relationships using RDFlib, where we add Codes to the PMDCo class SimulationProcess and Parameters to ValueObject. The simulation data from the hdf5 file of the finished pyiron workflow is extracted by running a PyironTable job. We compare the pyiron job input/output with metadata information and map those data in the application ontology that are defined in the metadata table. We afterwards follow the PMDCo guidelines to derive correct classes, object property and data property relations.

In addition to the pyiron workflow described above, in StahlDigital, other workflows have also been developed such as for integrating and processing data from mechanical experiments, including tensile tests, bulge tests, Nakajima tests etc. These workflows are supported by a dedicated platform powered by a dataspace management system (DSMS) developed at Fraunhofer IWM [https://stahldigital.materials-data.space/]. The DSMS allows for the setup and management of ontology-based dataspace solutions. Data pipelines related to these workflows semantically enrich tabular data from experimental measurements and transform relevant meta-information into RDF format using open-source libraries such as data2rdf [https://pypi.org/project/data2rdf/] and the DSMS-sdk [https://pypi.org/project/dsms-sdk/].

We can therefore conclude that the project StahlDigital provides several contributions to the development of workflows in the PMD, of which two have been particularly highlighted here. A first achievement is the implementation of multi-physics workflows in pyiron combining software tools from different communities. Beyond the above-mentioned tools, OpenPhase is also part of the StahlDigital workflows, providing the potential for establishing a link to the iBain activities outlined in Section 3.6 in the future. A second achievement is the connection of the workflows

used in this project with the PMDCo ontology,^[35] therewith linking the two major developments of the PMD. This part of StahlDigital deserves further attention in future projects.

3.13. SmaDi: Workflows in an OBDMA System

The aim of the project is an easy, scale-bridging data and model access for four exemplary subclasses of smart materials: thermal and magnetic shape memory alloys (SMA and MSMA), piezoelectric ceramics (PC), and dielectric elastomers (DE). Smart materials respond to physical fields (e.g., thermal fields, magnetic, and electrical) and are often used as bidirectional transducers.^[64] As these materials are a comparatively young research field, data sets are often heterogeneous, distributed over different tools, and partly inconsistent, especially when considering the dependence on the manufacturing process. The already established ontology-based data access (OBDA) is a suitable approach to allow easy common access to different databases of material subclasses via a common ontology and a mapping to the database.^[65] By extending this approach to an ontology-based data and model access (OBDMA), we combine data access with workflow steps for data processing, enabling, for example, a scale-bridging data access. One of our implemented use cases is the calculation of typical transducer characteristics based on the different lower scale material parameters. This enables users to pre-select suitable materials for their application without the need for expert knowledge of the different subclasses.

The special nature of the OBDMA-approach is characterized by the fact that the data is stored in conventional databases that are queried using an ontology. The predefined vocabulary stored in the TBox, which is used for the query, is provided with ontological information in a rewriting step and translated into the database vocabulary with the help of the mapping. The mapping thus enables access to the database and the ABox is generated

only virtually. In addition, this approach enables knowledge to be stored at different system levels (query, ontology, mapping, and database), resulting in a lightweight ontology. The unique feature of OBDMA is an ontology-driven workflow, where model functions are stored as user-defined function (UDFs) in the database. Access to the models is defined at the mapping level, while the workflow steps, including the functions and calculations, are defined at the database level. The system is based on Ontop, a widely used implementation of OBDA,^[66] which is also available as a plugin-reasoner in the ontology editor Protégé. The languages used on all four levels of the system are standards of the world wide web consortium (W3C):^[67] SPARQL for the query, OWL2QL for the ontology, R2ML for the mappings and SQL as the database language. The developed system is published on GitHub under the following URL: <https://github.com/SmaDi-OBDMASmaDi-OBDMASystem>.

The blocking stress is one of the common transducer characteristics implemented and is defined as the stress that results from the application of the appropriate physical field while simultaneously preventing deformation. It is defined slightly differently for all material subclasses and illustrated in Figure 17 for DE material to demonstrate the working principle of the OBDMA system. The query is rewritten and unfolded using the ontology and mappings to be answered on database level. Parameters that are not pre-stored can be computed using

UDFs. For DE the blocking stress σ_B can be calculated with the simple equation $\sigma_B = \epsilon E^2$, where ϵ is the permittivity and E the applied electrical field. The electrical field E is stored with a default value $E = 0.5E_{BFS}$, where E_{BFS} represents the dielectric strength of the material. Through the mappings, the result is fed back into the virtual ABox and used to generate the response.

The developed OBDMA system offers two decisive advantages. Firstly, it enables a shared access to potentially heterogeneous databases. Secondly, the included model access allows data processing, such as conversion of material parameters to transducer characteristics, as demonstrated for the blocking stress. This allows easy access to properties of transducers based on material parameters of the different subclasses without the need of expert knowledge. In particular, the ability to apply models to a variety of different materials helps the end user to select suitable materials. While the demonstrated use case is based on one simple equation, there are more complex equations and calculation steps with more than one workflow step implemented. Complex python scripts can also be accessed via user-defined functions, such as those used for the identification of the elastic parameters of DE.^[68] Nevertheless, as the system is developed to perform calculations for a large number of different material parameters simultaneously, it should focus on simpler functions to keep calculation times low. For complex workflows, like FEM-calculations, a combination of the OBDMA system with

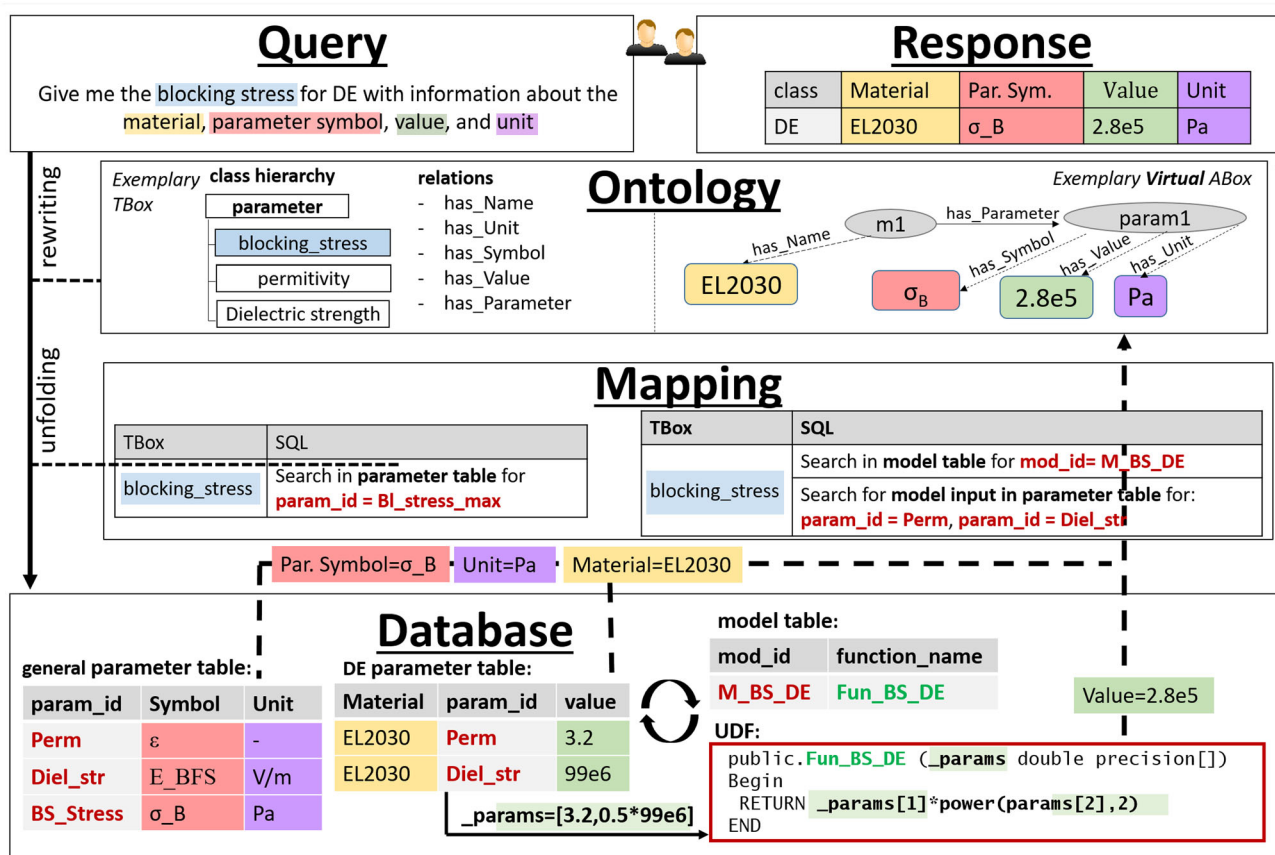


Figure 17. Working principle of the OBDMA system explained on the use cases blocking stress.

workflow environments such as pyiron or sim-stack could be advantageous. For a more detailed description of the OBDMA system, its advantages and other application examples, see also.^[69] As our system handles ontology-driven workflows, the entire OBDMA system, including the ontology, mapping, and database, will be provided in the PMD workflow store and is therefore fully accessible to all users and can, for example, be easily supplemented with additional data on the database level.

3.14. Conclusion

The project structure of the MaterialDigital initiative reflects the diversity of research activities typical of the community of materials science and engineering. Nevertheless, the involvement of workflows is a common feature of all these activities. In some projects, computational workflows form a central part of the activities. In other projects, the relevance of workflows is less transparent. However, there is no project in which the processing of data from one state (e.g., proprietary raw data of a device) to another format (e.g., the visualization of data) does not play a role. Despite the variety of implemented solutions, some general aspects have been identified that can and should be covered by the workflow environments and all implemented workflows.

One is the automation of repetitive work that can be covered by digital processes and triggers to these processes to reduce manual work and even reduce the number of touch points where manual starting of processes is required. The need for automation can be identified throughout the value chain of data. Firstly, at the point of data generation the process of acquiring data into a digital system should be automated, whether it is experimental data or data that are the result of a previous digital process. Secondly, the further processing of data can be automated, be it purely technical requirements such as the transformation of data formats or additional analysis and post-processing of said data. Finally, the generation of visualizations and other representations of the said data that transfer knowledge can be automated. At each stage, where it is possible to define all the necessary process parameters in advance, triggering subsequent process steps automatically relieves researchers from technical and repetitive tasks. It further reduces the danger of human errors and makes data handling protocols more transparent and reproducible.

One specific instance of automation involves the integration of interfaces with HPC facilities, enabling the utilization of computational resources seamlessly, without manual definition and execution of compute jobs for different compute facilities and workflow steps. This extends to automating the provisioning of compute environments for a single workflow step, whether on individual researcher's local setups or centralized facilities.

Another overarching need is the ability to easily collaborate and share data and workflows on all levels, for example, between individuals, groups, institutions, project partners, or the whole community (e.g., together with publications). This need extends from the development of workflows, where it should be easy to develop workflow steps separately through the use of specified software interfaces, to the execution of workflows from other authors, to the collaborative use of data generated by workflows in further workflows or manual processing. It is further expected to play a major role in training machine learning models over distributed

data. Achieving this requires user-friendly interfaces for accessing data and triggering automated workflows, alongside agreements on interoperable data structures and eventually ontological descriptions of workflows and their input and output data. Along these lines, both SimStack and pyiron provide valuable solutions.

The diversity of the use cases presented in this article represents a wide range of approaches and solutions to address these needs. The various workflow tools available have their specific applications, and there is no universal solution. Hence, continuous community efforts are essential to make the developed and diverse solutions readily available in the form of FAIR and modular workflows that can be reused by the whole materials science community.

Several satellite projects have already leveraged the PMD Workflow Store (see Section 4) to publish their developed workflows, with others planning to do so upon completion of final developments. However, examples for reuse of workflow solutions from one project via the workflow store in another project, which would be the ultimate demonstration of the desired interoperability, are still rare.

Ongoing work, coordinated by Platform MaterialDigital, aims to identify generalizable patterns from the presented and future use cases to find standards for FAIR, reusable, and modular workflows. This effort needs to focus on three separate aspects: standards for abstract, tool-independent workflow definitions; standards for sharing workflow implementations using specific workflow environments; and interoperable standards for storing and transferring workflow data between steps.

4. Advancements in Workflow Management: The MaterialDigital Workflow Store

The Platform MaterialDigital provides the Workflow Store <https://workflows.material-digital.de> as a central component in advancing computational workflow management within the Materials Science and Engineering (MSE) community. The major objective is to facilitate and foster the transition from manual data processing to the utilization of reusable workflows thereby enforcing core principles of the FAIR paradigm,^[2] aiming to enhance the Findability, Accessibility, Interoperability and Reusability not only of data, but also of workflow and software solutions.^[70]

4.1. The Concept of the Workflow Store

The Workflow Store is a digital registry designed to facilitate the storage, sharing, and discovery of computational workflows and workflow modules. This store provides a platform to help researchers find specific workflows, understand their functionality, and apply them to their own projects, fostering a culture of reuse and collaboration.

The Workflow Store encourages researchers to share their computational methods in a way that they can be easily reused by others. By providing a system where workflows and workflow modules are hosted, indexed, and made searchable, the Workflow Store removes barriers to accessing and implementing complex computational processes. This not only saves time and resources but also promotes transparency and reproducibility in scientific research (Figure 18).

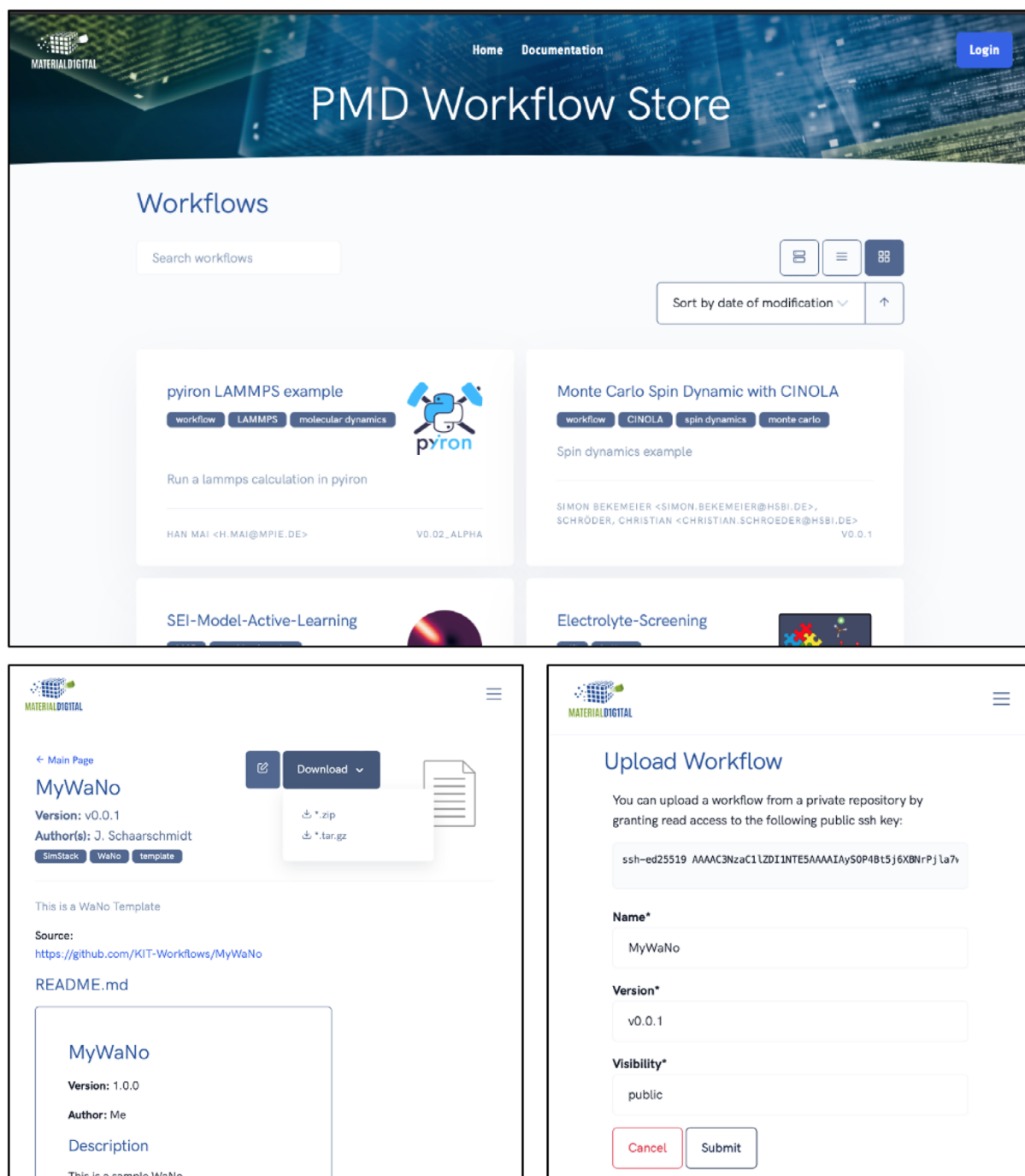


Figure 18. Overview of the PMD Workflow Store. The landing page (top) displays the indexed workflows accessible to the user, along with controls to search and sort them. The workflow page of an individual workflow (bottom left) provides detailed information about the workflow and options to download it. Users can upload and update workflows (bottom right) by setting the display name, specifying the main version as indicated by releases in the source repository, and adjusting the visibility settings.

To streamline the integration of workflows into the store, the store builds on the widely used version control systems Git. The process of adding workflows to the store involves registering existing repositories from hosting services like GitHub or GitLab, ensuring that the community can continue using familiar tools and platforms.

4.2. Metadata and Standards

Key to the functionality of the Workflow Store is the use of standardized metadata and documentation. Each workflow has to include a metadata file with essential information that enhances discoverability and usability. Furthermore, accompanying

documentation, typically provided through a Readme file, offers detailed instructions and insights, enabling users to understand and implement the workflows effectively.

4.3. Enhancing Computational Interoperability and Reproducibility

The Workflow Store advocates for the inclusion of Conda environment.yml files or Dockerfiles, although not mandatory, to detail the computational environment required for the workflow. This practice, alongside the recommendation to provide license and citation files, aligns with proven registry concepts, aiming to standardize and promote reproducibility.

These components are instrumental in ensuring that workflows are not only accessible but also operable in diverse computational settings, embodying the principles of reusability and interoperability. For the stakeholders of Platform MaterialDigital (PMD), a prototypical compute environment is provided at KIT and in terms of a deployment guide for a PMD server <https://materialdigital.github.io/pmd-server/>. Using an instance of this server in your own environment ensures the reliable and interoperable usage of PMD components with local data and resources. The PMD server is based on a docker compose system. For both, the PMD server at KIT and the local instances, docker images for pyiron are readily available. This includes all the sub-packages mentioned in Section 3.12 such as pyiron_base, pyiron_atomistic, pyiron_continuum and pyiron_experimental. They can be conveniently chosen by starting a JupyterHub session on the PMD server (instance). We note that other systems are supported in some of the projects, such as AixViPMaP in DiMad (Section 3.2) or SnakeMake in LebeDigital (Section 3.9).

4.4. Conclusion/Outlook

The introduction of the MaterialDigital Workflow Store marks a significant advancement toward streamlining workflow management in materials science. By leveraging established registry concepts and focusing on the automation and standardization of computational workflows, the store sets a new precedent for collaboration in the MSE domain.

For comprehensive guidelines on contributing to and utilizing the Workflow Store, as well as best practices for workflow management, interested parties are encouraged to consult the official documentation at <https://materialdigital.github.io/workflow-store/>.

Uploading a workflow Following the mentioned best practice guidelines ensures that the published workflow adheres to the FAIR principles.

Although it already provides a first entry point, the MaterialDigital Workflow Store is also set to undergo significant enhancements that will further consolidate its position as an integral tool for the MSE community. Planned features include a deeper integration with established workflow frameworks such as SimStack and pyiron, offering seamless interoperability and streamlined workflow execution. Next to the current checks for available metadata, we furthermore intend to add features, highlighting the quality and adherence to FAIR principles of

available workflows, such as including information about test coverage, proper description of the required execution environment, or usage metrics like number of downloads or number of workflows a workflow node is part of. The store will also introduce more granular access control levels, allowing contributors to fine-tune the sharing settings of their workflows. Additionally, the linkage to scientific ontologies is an important milestone, aiming to improve the semantic association and discoverability of workflows. These advances are expected to enhance the user experience, promoting a more dynamic, collaborative, and efficient research ecosystem.

5. Future Work

One of the foremost areas of enhancing the role of workflows in the MaterialDigital initiative lies in improving user interaction and accessibility. This could involve the development of a more intuitive user interface for the Workflow Store, ensuring that novice and experienced users can easily navigate the platform. At the same time, it is crucial to make workflow environments flexible enough to enable the inclusion of novel features for code developers. In this chapter, we will present various possibilities to explore potential future paths for PMD, as well as possible solutions to the challenges we faced and described in the previous chapters.

5.1. Collaborations and Synergies

An option for improvement is to lower the barrier for the incorporation of solutions in other digitalization initiatives. At the international level, these are large-scale activities such as the MaterialsProject^[71] or NOMAD.^[72] They are often focused on the sharing of data, but less on their provenance and reproducibility. We can therefore provide the workflow tools developed in PMD to flexibly include more physical relations in their database. On the national level of Germany, the National Research Data Initiative (NFDI) consortia such as NFDI-MatWerk and FAIRmat are national partners of the PMD. They have access to a broad community of academic research institutions, together with their cutting-edge research and large user base. More generally, it is a challenge for workflow frameworks such as SimStack or pyiron, that the number of functionalities available beyond the platforms they are based on is limited. This point becomes particularly important when the user already has a well-established workflow relying on a GUI such as ParaView,^[73] Abaqus,^[74] etc.

5.2. Standardization of Workflows

One crucial aspect of improving user experience in workflow design involves establishing a standardized input/output framework. This standardization is pivotal as it simplifies the comprehension of workflows for users, promoting a seamless and intuitive interaction. Moreover, standardization allows for computational frameworks to be able to combine various workflows without further intervention, enabling to bridge simulation tools of various scales and different branches. As a result, while the Workflow Store currently encourages users to base their

workflows on pyiron or SimStack, their interoperability with other platforms, such as AiiDA,^[75,76] atomate,^[77] or other workflow tools, will not only enhance the user experience, but will also potentially nurture collaboration among different platforms, leading to stronger synergy. Another aspect of a standardized input/output framework is the option to include details in the specification such as specific data types or employed unit systems, ensuring only meaningful connections between out- and inputs can be established or data is appropriately transformed in between.

5.3. Workflows and Ontologies

To enhance the reusability of a workflow, it is essential to provide additional specifications and resources beyond simply making them available in workflow stores. Users must have a clear understanding of the purpose of a workflow, facilitated by detailed documentation. This documentation should be self-explanatory, incorporating sample input and output data to ensure optimal utilization. Equally significant are provenance data and the recognition of individuals involved in the scientific citation context.^[78]

Ontologies and related frameworks, which are gaining increasing importance in materials science and engineering, enable detailed descriptions of processes and steps through the semantic annotation of (meta)data.^[79,80] Important ontological resources in this context include PROV-O, a W3C standard specifically developed to describe workflows.^[81] In the context of materials science and engineering, the PMD Core Ontology (PMDco) v2.0.7 is also essential, building upon PROV-O^[35] and extending it within the scope of materials science. In response to industry requirements, PMDco v3.0 is being modularized and enhanced using the standardized top-level basic formal ontology (BFO).^[82] Similarly, the elementary multi-perspective material ontology (EMMO) provides modules^[83] to describe characterization workflows in connection with data processing pipelines. The Common Workflow Language (CWL) offers interoperable workflow definitions, promoting a standardized approach^[84] that already supports references to vocabular terms. Recommendations for a provenance framework have also been published, which could contribute to standardization efforts.^[85]

Using ontologies and integrating crucial contextual information into workflow descriptions can enhance the search functionality of the database and the app store. Searches become more precise, yielding context-specific results. For instance, searching for a specific material can reveal available workflows, along with associated raw data, experiments, simulations, protocols, publications, and even researchers working on similar topics.^[86,87] By interpreting workflows connected by their input and output specification as a network or graph, determining routes through complex data transformations can become an even more sophisticated application of ontologized workflow descriptions.

However, it remains challenging to accurately capture the complex semantics of workflows with ontologies. Current practices may not yet be sufficient to fully represent the relationships between different workflow steps, the contexts in which they are applied, and the interdependencies between various data

elements and processes. While there are examples in the literature where ontologies have improved the reproducibility of workflow steps by semantically representing used parameters,^[88,89] a fully comprehensive approach does still not exist.

5.4. Incorporating Technological Advancements

Emphasizing efficient search capabilities ensures seamless interaction by simplifying workflow comprehension. Integrating natural language descriptions for compiled code, using the existing-workflow descriptions, coupled with large language models (LLM), enhances search precision. This streamlined approach optimizes user interactions and establishes a distinctly user-centric workflow design.

5.5. Workflows and Industry

In the current implementation of the Workflow Store, there is currently only a distinction between public, internal, and private. However, as the number of industrial partners grows, there is a growing demand for more fine-grained access control. In particular, it is indispensable for the industry to be able to make certain parts of their data as well as workflows available while sharing other parts only with a select audience based on individual contracts. In a first step, users can share workflows via the Workflow Store but provide the required software only on their own servers. Subsequently, solutions for other cases such as request-based access or workflows as a service will be implemented. However, especially if workflows require commercial software packages, the author needs to ensure that the user can follow basic instructions to apply for licenses and run the workflow with the acquired licenses.

In digital workflows, the importance of standardization and backward compatibility as catalysts for robust collaboration between academia and industry is set to become increasingly pivotal. The trajectory of this collaboration is steering toward an ecosystem where data and systems interoperability are not merely advantageous, but essential. The strategic foresight of standardization lies in its ability to pre-emptively address the complexities of future technologies. As we delve further into the era of big data and AI, seamlessly integrating diverse datasets, tools, and methodologies becomes crucial. Open and clarified standards are instrumental in this regard.^[90] They foster a common language for collaboration and ensure that emerging technologies can be swiftly incorporated into existing digital ecosystems without causing disruption.

The role of backward compatibility, in harmony with standardization, is to provide a safety net for innovation.^[91] It guarantees that adopting new technologies can entail a partial overhaul of existing systems. This continuity is especially vital in collaborative ventures between academia and industry, where the stakes and scales of investments are significantly high. Integrating new academic research into industrial applications often requires navigating a labyrinth of legacy systems. Backward compatibility ensures that this integration is not a bottleneck but a smooth transition, accelerating innovation and application. SimStack and pyiron emerge as workflow framework solutions that synergize standardization and backward compatibility in this context.

Both tools illustrate how tools designed with these principles can significantly enhance collaborative efforts by accommodating multi-module simulations and reducing setup complexity. It shows that the future of digital workflows lies in developing new technologies and creating environments where these technologies can be effortlessly integrated and leveraged.

The interplay between academia and industry in digital workflows is poised to become more dynamic. With the advent of technologies such as quantum computing and the increasing ubiquity of IoT devices, the volume and complexity of data is set to grow exponentially. In this context, the principles of standardization and backward compatibility will be the bedrock upon which successful, sustainable, and scalable collaborations are built. They will ensure that future digital workflows are robust, efficient, but also inclusive, and progressive, driving the frontier of innovation forward without leaving anyone behind.

5.6. Toward FAIR Workflows

In summary, to achieve FAIR workflows, several key practices should be adopted. First, putting workflows under version control and registering them in platforms like the PMD Workflow Store ensures both Findability and Accessibility. Formalizing workflows within a dedicated workflow environment guarantees basic Reusability by standardizing execution processes. Additionally, adherence to recommended guidelines—such as describing the compute environment in detail and integrating ontologies enhances the reusability and Interoperability of workflows and their constituent nodes. These steps collectively contribute to the creation of robust, FAIR-compliant workflows.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors thank the German Federal Ministry of Education and Research (BMBF) for financial support of the project Innovation-Platform MaterialDigital through project funding grant no. 13XP5094E (BAM), 13XP5094C (MPIE), and 13XP5094A (KIT).

Open Access funding enabled and organized by Projekt DEAL.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Simon Bekemeier: conceptualization (lead); project administration (equal); writing—original draft (equal); writing—review and editing (equal). **Celso Ricardo Caldeira Rêgo:** writing—original draft (equal); writing—review and editing (equal). **Han Lin Mai:** conceptualization (equal); writing—original draft (equal); writing—review and editing (equal). **Ujjal Saikia:** conceptualization (supporting); writing—original draft (supporting); writing—review and editing (supporting). **Osamu Waseda:** conceptualization (lead); writing—original draft (equal); writing—review and editing (equal). **Markus Apel:** writing—original draft

(supporting); writing—review and editing (supporting). **Felix Arendt:** writing—original draft (supporting); writing—review and editing (supporting). **Alexander Aschemann:** writing—original draft (supporting); writing—review and editing (supporting). **Bernd Bayerlein:** conceptualization (supporting); writing—original draft (supporting); writing—review and editing (supporting). **Robert Courant:** writing—original draft (supporting); writing—review and editing (supporting). **Gordian Dziwis:** writing—original draft (supporting); writing—review and editing (supporting). **Florian Fuchs:** conceptualization (supporting); writing—original draft (supporting); writing—review and editing (supporting). **Ulrich Giese:** writing—original draft (supporting); writing—review and editing (supporting). **Kurt Junghanns:** writing—original draft (supporting); writing—review and editing (supporting). **Mohamed Kamal:** writing—original draft (supporting); writing—review and editing (supporting). **Lukas Koschmieder:** writing—original draft (supporting); writing—review and editing (supporting). **Sebastian Leineweber:** writing—original draft (supporting); writing—review and editing (supporting). **Marc Luger:** writing—original draft (supporting); writing—review and editing (supporting). **Marco Lukas:** writing—original draft (supporting); writing—review and editing (supporting). **Jürgen Maas:** writing—original draft (supporting); writing—review and editing (supporting). **Jana Mertens:** writing—original draft (supporting); writing—review and editing (supporting). **Björn Mieller:** writing—original draft (supporting); writing—review and editing (supporting). **Ludger Overmeyer:** writing—original draft (supporting); writing—review and editing (supporting). **Norbert Pirch:** writing—original draft (supporting); writing—review and editing (supporting). **Jan Reimann:** writing—original draft (supporting); writing—review and editing (supporting). **Sebastian Schröck:** writing—original draft (supporting); writing—review and editing (supporting). **Philipp Schulze:** writing—original draft (supporting); writing—review and editing (supporting). **Jörg Schuster:** writing—original draft (supporting); writing—review and editing (supporting). **Alexander Seidel:** writing—original draft (supporting); writing—review and editing (supporting). **Oleg Shchyglo:** writing—original draft (supporting); writing—review and editing (supporting). **Marek Sierka:** writing—original draft (supporting); writing—review and editing (supporting). **Frank Silze:** writing—original draft (supporting); writing—review and editing (supporting). **Simon Stier:** conceptualization (supporting); writing—original draft (equal); writing—review and editing (equal). **Marvin Tegeler:** writing—original draft (supporting); writing—review and editing (supporting). **Jörg F. Unger:** conceptualization (supporting); writing—original draft (supporting); writing—review and editing (supporting). **Matthias Weber:** writing—original draft (supporting); writing—review and editing (supporting). **Tilmann Hickel:** conceptualization (lead); project administration (lead); writing—original draft (lead); writing—review and editing (lead). **Jörg Schaarschmidt:** conceptualization (lead); project administration (lead); writing—original draft (lead); writing—review and editing (lead).

Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Keywords

digitalisation, FAIR principles, MaterialDigital, scientific workflows, semantic interoperability

Received: September 12, 2024

Revised: December 2, 2024

Published online:

- [1] B. Bayerlein, J. Waitelonis, H. Birkholz, M. Jung, M. Schilling, P. V. Hartrott, M. Bruns, O. Waseda, J. Schaarschmidt, K. Beilke, M. Mutz, V. Nebel, V. Königer, L. Beran, T. Kraus, A. Vyas,

- L. Vogt, M. Blum, B. Ell, Y.-F. Chen, T. Waurischk, A. Thomas, A. R. Durmaz, S. Ben Hassine, C. Fresemann, H. N. Beygi, G. Dziwis, T. Hanke, M. Telong, S. Pirsakawetz, M. Kamal, T. Bjarsch, U. Pähler, P. Hofmann, M. Leemhuis, A. Özçep, L.-P. Meyer, B. Skrotzki, J. Neugebauer, W. Wenzel, H. Sack, C. Eberl, P. Portella, T. Hickel, L. Mädler, P. Gumbsch, *Adv. Eng. Mater.* **2024**, 2401092.
- [2] M. Wilkinson, M. Dumontier, I. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. Bonino da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. A. Gonzalez-Beltran, A. J. G. Gray, P. Groth, C. Goble, J. S. Grethe, B. Mons, *Sci. Data* **2016**, 3, 160018.
- [3] J. Janssen, S. Surendralal, Y. Lysogorskiy, M. Todorova, T. Hickel, R. Drautz, J. Neugebauer, *Comput. Mater. Sci.* **2019**, 163, 24.
- [4] C. R. C. Rêgo, J. Schaarschmidt, T. Schlöder, M. Penalzoa-Amion, S. Bag, T. Neumann, T. Strunk, W. Wenzel, *Front. Mater.* **2022**, 9, 877597.
- [5] *Meinwah/Awesome-Workflow-Engines*, **2024**, <https://github.com/meinwah/awesome-workflow-engines> (accessed: May 2024).
- [6] P. Amstutz, M. Mikheev, M. R. Crusoe, N. Tijanic, S. Lampa, *Existing Workflow Systems*, **2024**, <https://s.apache.org/existing-workflow-systems> (accessed: May 2024).
- [7] J. Schaarschmidt, J. Yuan, T. Strunk, I. Kondov, S. P. Huber, G. Pizzi, L. Kahle, F. T. Bölle, I. E. Castelli, T. Vegge, F. Hanke, T. Hickel, J. Neugebauer, C. R. C. Rêgo, W. Wenzel, *Adv. Energy Mater.* **2021**, 12, 17.
- [8] A. Jain, S. P. Ong, W. Chen, B. Medasani, X. Qu, M. Kocher, M. Brafman, G. Petretto, G.-M. Rignanese, G. Hautier, D. Gunter, K. A. Persson, *Concurrency Comput.: Pract. Exper.* **2015**, 27, 5037.
- [9] L.-F. Zhu, J. Janssen, S. Ishibashi, F. Körmann, B. Grabowski, J. Neugebauer, *Comput. Mater. Sci.* **2021**, 187, 110065.
- [10] H. Zhao, L. Huber, W. Lu, N. J. Peter, D. An, F. De Geuser, G. Dehm, D. Ponge, J. Neugebauer, B. Gault, D. Raabe, *Phys. Rev. Lett.* **2020**, 124, 106102.
- [11] C. Freysoldt, A. Mishra, M. Ashton, J. Neugebauer, *Phys. Rev. B* **2020**, 102, 045403.
- [12] M. Soleymanibrojeni, C. R. Caldeira Rego, M. Esmailpour, W. Wenzel, *J. Mater. Chem. A* **2024**, 12, 2249.
- [13] H. Pecinatto, C. R. C. Rêgo, W. Wenzel, C. A. Frota, B. M. S. Perrone, M. J. Piotrowski, D. Guedes-Sobrinho, A. C. Dias, C. Mota, M. S. S. Gusmão, H. O. Frota, *Sci. Rep.* **2023**, 13, 1.
- [14] M. Mostaghimi, C. R. C. Rêgo, R. Haldar, C. Wöll, W. Wenzel, M. Kozłowska, *Front. Mater.* **2022**, 9, 840644.
- [15] A. Ghosh, *Comput. Mater. Sci.* **2024**, 233, 112740.
- [16] N. Miklin, A. A. Abbott, C. Branciard, R. Chaves, C. Budroni, *New J. Phys.* **2017**, 19, 113041.
- [17] A. Saha, O. Hassanzadeh, A. Gittens, J. Ni, K. Srinivas, B. Yener, arXiv preprint arXiv:2308.03891 **2023**.
- [18] BITMOTEC, *Bitmotecosystem*, <https://www.bitmotec.com/technologie> (accessed: February, 2024).
- [19] OpenJS Foundation, *Node-RED*, **2013**, <https://nodered.org> (accessed: February, 2024).
- [20] InfluxData, *Influxdb*, **2013**, <https://www.influxdata.com> (accessed: February, 2024).
- [21] M. Lukas, S. Leineweber, B. Reitz, L. Overmeyer, A. Aschemann, B. Klie, U. Giese, in *Congress of the German Academic Association for Production Technology*, Springer, New York **2023**, pp. 539–549.
- [22] M. Lukas, S. Leineweber, B. Reitz, L. Overmeyer, A. Aschemann, B. Klie, U. Giese, in *Proc. ACS Int. Elastomer Conf.*, Cleveland Rubber Division - American Chemical Society (ACS) **2023**.
- [23] M. Lukas, S. Leineweber, B. Reitz, L. Overmeyer, A. Aschemann, B. Klie, U. Giese, *Rubber Chem. Technol.* **2024**, 97, 371.
- [24] M. Lukas, S. Leineweber, B. Reitz, L. Overmeyer, *Int. J. Adv. Manufact. Technol.* **2024**, 133, 5233.
- [25] J. Andersson, T. Helander, L. Höglund, P. Shi, B. Sundman, *Calphad* **2016**, 26, 273.
- [26] B. Böttger, M. Apel, M. Budnitski, J. Eiken, G. Laschet, B. Zhou, *Comput. Mater. Sci.* **2020**, 184, 109909.
- [27] N. Pirch, S. Linnenbrink, A. Gasser, H. Schleifenbaum, *Int. J. Heat Mass Transfer* **2019**, 143, 273.
- [28] L. Koschmieder, S. Hojda, M. Apel, R. Altenfeld, Y. Bami, C. Haase, M. Lin, A. Vuppala, G. Hirt, G. Schmitz, *Integr. Mater. Manufact. Innov.* **2019**, 8, 122.
- [29] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, in *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (Eds: F. Loizides, B. Schmidt), IOS Press, Amsterdam **2016**, pp. 87–90.
- [30] L. Rizzi, A. Zienert, J. Schuster, M. Köhne, S. E. Schulz, *ACS Appl. Mater. Interfaces* **2018**, 10, 43088.
- [31] L. Rizzi, A. Zienert, J. Schuster, M. Köhne, S. E. Schulz, *Comput. Mater. Sci.* **2019**, 161, 364.
- [32] C. E. Rasmussen, C. K. I. Williams, in *Gaussian Processes for Machine Learning*, The MIT Press, Cambridge **2005**.
- [33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, *J. Mach. Learn. Res.* **2011**, 12, 2825.
- [34] S. Boeck, C. Freysoldt, A. Dick, L. Ismer, J. Neugebauer, *Comput. Phys. Commun.* **2011**, 182, 543.
- [35] B. Bayerlein, M. Schilling, H. Birkholz, M. Jung, J. Waitelonis, L. Mädler, H. Sack, *Mater. Design* **2024**, 237, 112603.
- [36] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, S. J. Plimpton, *Comp. Phys. Commun.* **2022**, 271, 108171.
- [37] M. Tegeler, O. Shchyglo, R. D. Kamachali, A. Monas, I. Steinbach, G. Sutmann, *Comput. Phys. Commun.* **2017**, 215, 173.
- [38] OpenPhase, *OpenPhase Software Library for Phase-Field Simulations*, **2023**, <https://openphase.rub.de>.
- [39] W. Jakob, J. Rhineland, D. Moldovan, *pybind11: Seamless Operability Between C++11 and Python*, **2017**, <https://github.com/pybind/pybind11>.
- [40] K. Ratschbacher, U. E. Klotz, M. Eisenbart, *Mater. Sci. Technol.* **2018**, 36, 925.
- [41] O. K. Foundation, *The World's Leading Open Source Data Management System*, **2024**, <https://ckan.org/> (accessed: March 2024).
- [42] *Nextflow*, **2024**, <https://nextflow.io/> (accessed: March 2024).
- [43] G. Dziwis, *MiniPoD a MINimal PoDman Deployment*, **2024**, <https://gitlab.com/infai/minipod> (accessed: March 2024).
- [44] S. Chávez-Feria, R. García-Castro, M. Poveda-Villalón, in *The Semantic Web* (Eds: P. Groth, M.-E. Vidal, F. Suchanek, P. Szekley, P. Kapanipathi, C. Pesquita, H. Skaf-Molli, M. Tamper), Springer International Publishing, Cham **2022**, pp. 338–352.
- [45] A. S. Foundation, *Apache Jena Fuseki*, **2024**, <https://jena.apache.org/documentation/fuseki2/> (accessed: March 2024).
- [46] S. Ferré, *Sparklis*, **2024**, <https://github.com/sebferre/sparklis> (accessed: March 2024).
- [47] R. Albertoni, D. Browning, S. Cox, A. G. Beltran, A. Perego, P. Winstanley, *Data Catalog Vocabulary (DCAT) - Version 2*, **2024**, <https://www.w3.org/TR/vocab-dcat-2/> (accessed: March 2024).
- [48] S. Harris, A. Seaborne, *E. Prud'hommeaux Language*, **2024**, <https://www.w3.org/TR/sparql11-query/> (accessed: March 2024).

- [49] A. Agrawal, E. Tamsen, P.-S. Koutsourelakis, J. F. Unger, *From Concrete Mixture to Structural Design – A Holistic Optimization Procedure in the Presence of Uncertainties*, **2023**.
- [50] P. Diercks, D. Gläser, *ing.grid* **2023**, 1.
- [51] F. Mölder, K. P. Jablonski, B. Letcher, M. B. Hall, C. H. Tomkins-Tinch, V. Sochat, J. Forster, S. Lee, S. O. Twardziok, A. Kanitz, A. Wilm, M. Holtgrewe, S. Rahmann, S. Nahnsen, J. Köster, *F1000Res*, **2021**, 10, 33.
- [52] M. Alnaes, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. Rognes, G. Wells, *Arch. Num. Softw.* **2015**, 3.
- [53] E. N. Schettino, *pydoit/doit: Task Management & Automation Tool (Python)*, Zenodo **2021**.
- [54] T. O'Brien, *elabFTW*, **2012**, <https://www.elabftw.net/> (accessed: January 2024).
- [55] C. Barillari, D. S. M. Ottoz, J. M. Fuentes-Serna, C. Ramakrishnan, B. Rinn, F. Rudolf, *Bioinformatics* **2015**, 32, 638.
- [56] L. Mishnaevsky, K. Branner, H. N. Petersen, J. Beauson, M. McGugan, B. F. Sørensen, *Materials* **2017**, 10, 11.
- [57] E08 Committee, *Practices for Cycle Counting in Fatigue Analysis*.
- [58] *Jupyter Widgets*, <https://github.com/jupyter-widgets/ipywidgets> (accessed: February 2024).
- [59] C. B. Sullivan, A. Kaszynski, *J. Open Source Softw.* **2019**, 4, 1450.
- [60] J.-B. Lamy, *Artif. Intell. Med.* **2017**, 80, 11.
- [61] D. Systèmes, *Abaqus*, <https://www.3ds.com/products/simulia/abaqus> (accessed: February 2024).
- [62] G. Dhondt, K. Wittig, *Calculix*, <http://www.calculix.de/> (accessed: February 2024).
- [63] CMT Inc., *Compro*, <https://www.convergent.ca/products/compro-simulation-software> (accessed: February 2024).
- [64] W.-G. Drossel, H. Kunze, A. Bucht, L. Weisheit, K. Pagel, *Proc. CIRP* **2015**, 36, 211.
- [65] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, in *Journal on Data Semantics X*, Springer, Cham **2008**, pp. 133–173.
- [66] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, G. Xiao, *Seman. Web* **2017**, 8, 471.
- [67] <https://www.w3.org/>.
- [68] J. Mertens, M. Leemhuis, H. Schmidtke, Ö. Özçep, J. Maas, in *Electroactive Polymer Actuators and Devices (EAPAD) XXV*, 12482, SPIE, Bellingham WA **2023**, pp. 312–319.
- [69] J. Maas, M. Leemhuis, J. Mertens, H. Schmidtke, R. Courant, M. Dahlmann, S. Stark, A. Böhm, K. Pagel, M. Hinze, D. Pinkal, M. Wegener, M. F.-X. Wagner, T. Sattel, H. Neubert, *Adv. Eng. Mater.* **2024**, 2302208.
- [70] M. Barker, N. P. Chue Hong, D. S. Katz, A.-L. Lamprecht, C. Martinez-Ortiz, F. Psomopoulos, J. Harrow, L. J. Castro, M. Gruenpeter, P. A. Martinez, T. Honeyman, *Sci. Data* **2022**, 9, 622.
- [71] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, K. A. Persson, *APL Mater.* **2013**, 1, 1.
- [72] M. Scheidgen, L. Himanen, A. N. Ladines, D. Sikter, M. Nakhaee, A. Fekete, T. Chang, A. Golparvar, J. A. Marquez, S. Brockhauser, S. Brückner, L. M. Ghiringhelli, F. Dietrich, D. Lehmberg, T. Denell, A. Albino, H. Näsström, S. Shabih, F. Dobener, M. Kühbach, R. Mozumder, J. F. Rudzinski, N. Daelman, J. M. Pizarro, M. Kuban, C. Salazar, P. Ondracka, H.-J. Bungartz, C. Draxl, *J. Open Source Softw.* **2023**, 8, 5388.
- [73] J. Ahrens, B. Geveci, C. Law, C. Hansen, C. Johnson, in *The Visualization Handbook*, Vol. 717, Elsevier, Amsterdam **2005**, p. 50038.
- [74] M. Smith **2009**.
- [75] S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky, G. Pizzi, *Sci. Data* **2020**, 7, 300.
- [76] M. Uhrin, S. P. Huber, J. Yu, N. Marzari, G. Pizzi, *Comput. Mater. Sci.* **2021**, 187, 110086.
- [77] K. Mathew, J. H. Montoya, A. Faghaninia, S. Dwarakanath, M. Aykol, H. Tang, I.-H. Chu, T. Smidt, B. Bocklund, M. Horton, J. Dagdelen, B. Wood, Z.-K. Liu, J. Neaton, S. P. Ong, K. Persson, A. Jain, *Comput. Mater. Sci.* **2017**, 139, 140.
- [78] K. Belhajjame, J. Zhao, D. Garijo, M. Gamble, K. Hettne, R. Palma, E. Mina, O. Corcho, J. M. Gómez-Pérez, S. Bechhofer, G. Klyne, C. Goble, *J. Web Seman.* **2015**, 32, 16.
- [79] B. Bayerlein, T. Hanke, T. Muth, J. Riedel, M. Schilling, C. Schweizer, B. Skrotzki, A. Todor, B. Moreno Torres, J. F. Unger, C. Völker, J. Olbricht, *Adv. Eng. Mater.* **2022**, 24, 2101176.
- [80] A. Valdestilhas, B. Bayerlein, B. Moreno Torres, J. Z. Ghezal Ahmad, T. Muth, *Adv. Intell. Syst.* **2023**, 5, 2300051.
- [81] T. Lebo, S. S. Sahoo, D. L. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, J. Zhao, *PROV-O: The PROV Ontology*, **2013**, <https://www.w3.org/TR/2013/REC-prov-dm-20130430/> (accessed: January 2024).
- [82] International Organisation for Standardization, *Information Technology - Top-level Ontologies (TLO) - Part 2: Basic Formal Ontology (BFO), Standard*, International Organisation for Standardization **2021**, <https://www.beuth.de/de/norm/iso-iec-21838-2/348948268>.
- [83] P. Del Nostro, G. Goldbeck, D. Toti, *The Chameo Ontology: Exploiting Emmo's Multiperspective Versatility for Capturing Materials Characterization Procedures*, **2022**, <https://ceur-ws.org/Vol-3240/short3.pdf>.
- [84] P. Amstutz, M. Crusoe, F. Khan, S. Soiland-Reyes, M. Singh, K. Kumar, A. Khodak, P. R. Safont, J. Chilton, T. Hickman Boysha, J. Holland, B. Champman, M. Kotlair, D. Leehr, G. Carrasco, A. Kartashov, J. C. Randall, N. Tijanic, H. Ménager, T. Tanjo, J. J. Porter, G. Molenaar, M. Tong, G. Tagliabue, D. Yuen, A. Barrera, S. Ivkovic, R. Spangler, Zenodo **2018**, <https://zenodo.org/records/1471589> (accessed: January 2024).
- [85] F. Z. Khan, S. Soiland-Reyes, R. O. Sinnott, A. Lonie, C. Goble, M. R. Crusoe, *GigaScience* **2019**, 8, giz095.
- [86] D. Bonino, F. Corno, L. Farinetti, A. Bosca, *WSEAS Trans. Inform. Sci. Appl.* **2004**, 1, 1597.
- [87] T. Schneider, M. Šimkus, *Künstliche Intelligenz* **2020**, 34, 329.
- [88] G. A. J. Zia, T. Hanke, B. Skrotzki, C. Völker, B. Bayerlein, *Integr. Mater. Manufact. Innov.* **2024**, 13, 257.
- [89] S. P. Stier, X. Xu, L. Gold, M. Möckel, *Energy Technol.* **2024**, 12, 2301305.
- [90] Y. Fukami, *Rev. Socionetw. Strat.* **2021**, 15, 535.
- [91] S. Spaeth, S. Niederhöfer, *Electron. Mark.* **2022**, 32, 1891.