D9.1 – Prototype of the BIG-MAP App Store

VERSION

VERSION	DATE
1.2	02-26-2021

PROJECT INFORMATION

GRANT AGREEMENT NUMBER	957189
PROJECT FULL TITLE	Battery Interface Genome - Materials Acceleration Platform
PROJECT ACRONYM	BIG-MAP
START DATE OF THE PROJECT	1/9-2020
DURATION	3 years
CALL IDENTIFIER	H2020-LC-BAT-2020-3
PROJECT WEBSITE	big-map.eu

DELIVERABLE INFORMATION

WP NO.	9
WP LEADER	EPFL
CONTRIBUTING PARTNERS	KIT, EPFL, DTU
NATURE	Report
AUTHORS	Wolfgang Wenzel, Simon Adorf, Nicola Marzari, Giovanni Pizzi, Joerg Schaarschmidt, Jie Yuan, Celso R. C. Rêgo, Ivano E. Castelli, Eibar Flores ; Tejs Vegge
CONTRIBUTORS	Same as authors
CONTRACTUAL DEADLINE	2-28-2021
DELIVERY DATE TO EC	2-28-2021
DISSEMINATION LEVEL (PU/CO)	PU

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957189.

ABSTRACT

Workflows are an integral part of the strategy of the BIG-MAP project to disseminate complex computational protocols, and at later stages, integrated experimental-computational protocols to the researchers within the project, researchers in other projects of the Battery 2030+ family and ultimately to the battery research community at large. To this end complicated computational protocols have to be encapsulated in transferable modules using workflow-technologies that the couple control of the workflow and computational. To this end we have identified several established, high performance workflow engines, which will be used for this purpose. In order to make these protocols available to select the target audiences defined above, the protocols need to be discoverable and accessible. To this end we have designed and implemented in App Store where researchers developing the protocols can easily deposit the results of the work, while other researchers can find, download and reuse the protocols to accelerate battery research in Europe.

TABLE OF CONTENTS

1. Introduction

Materials are an essential basis for the development of new technological solutions in the fields of energy and environment, health, information and communication, manufacturing, or security and transport, but their development and adaptation remains often a time and resource intensive endeavor. Experimental efforts to develop and design new materials are increasingly complemented by computational strategies. This mirrors the trend in many other application areas, where the computer aided design has significantly accelerated product development, while often reducing the cost at the same time. Examples are the automotive, aerospace and electronics industry, where the development of novel products is unthinkable without computer aided design. A prerequisite for the successful application of such a strategy is the availability of predictive simulation protocols, which can be used as digital twins for devices in the context of development and design.

Materials design is still lagging behind other fields in the application of computer aided design strategies, not for lack of effort, but because of the complexity of the underlying task. Material properties are typically queried at the scale of micrometers or beyond, but manipulated at the nanometer scale or below. The field is lacking a monolithic computational framework to cover all of these scales, in both space and time, with attainable computational resources. Computational materials development necessarily requires the combination of several methods to generate a predictive materials model or digital twin. There is an enormous literature on the development and application of the components and their integration into complex protocols[1–5]. There have been several recent high-level initiatives, such as the Materials Genome Initiative or the Materials Project[5,6], Materials design at the Exascale (MAX, http://www.max-centre.eu), or the Harvard clean energy initiative[7] that aim at the computational development of novel materials. However, there are relatively few reports on tools that may be used to formalize the execution of the underlying protocols, which we call workflows in the following. In addition to the actual computation, various other steps, including retrieval, preprocessing, transformation, analysis, and deposition of data are an integral part of the process, which need to be addressed in a scientific workflow.

2. Scientific workflows

Scientific workflows can be viewed as a formal modeling method rooted in both simulation and data analytics, which can be used to describe complex physical systems and phenomena. A workflow represents the coordinated execution of repeatable computational steps while accounting for dependencies and concurrency of tasks. While in computational science the workflow approach has a long established tradition[8–10], it has only started to gain traction in computational materials science and computational chemistry within the last decade[6,11–13]. A workflow can be formally described by a directed acyclic graph, in which vertices denote the actions, and the edges denote the execution and data dependencies (known as control flow and data flow). A conceptual overview of the components of a workflow framework is given in Figure 1, and examples for concrete workflows will be given below. While the design and study of digital twins does not necessarily require a workflow framework, there are many compelling reasons for their utilization. By providing a layer of abstraction, workflows enable scientists to design and conduct studies without in-depth knowledge of the software deployment on computational resources. Furthermore, they improve the transfer of knowledge within groups, between collaborators or with customers by providing a

concise description of the input data, utilized software and scripts, as well as the respective parameters and settings used for a specific project. In consequence, applying the same methodology to a new system or scaling up a study only requires more of a conceptual understanding of the subject and the adjustment of input data or run parameters. Especially with increased efforts for scientific data to adhere to the FAIR guidelines[14], which require scientific data to be findable, accessible, interoperable, and reusable, workflow frameworks provide an important tool to improve the interoperability and reusability of published data.

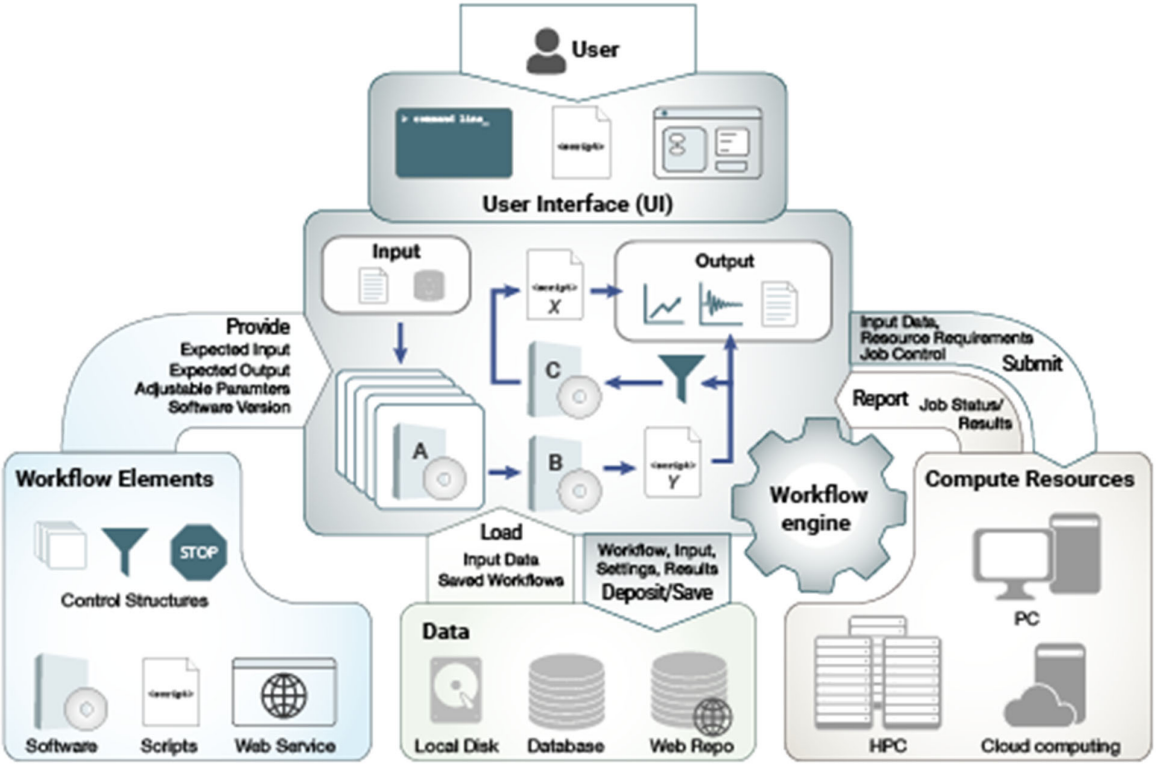

Figure 1. Schematic representation of the components of a workflow framework. A workflow consist of several interconnected data operations (center). Dependencies and data flow (indicated by arrows) between these operations are captured in the workflow and handled by the workflow framework. Furthermore, the workflow framework handles the interaction with various data and compute resources. The user interacts with the workflow framework via a user interface (UI), which enables the user to edit the workflow architecture, interact with data storage elements, target compute resource, as well as define and adjust parameters and settings of the individual workflow elements.

2.1 Benefits of workflows

The main benefits of using workflows in multiscale and high-throughput simulation can be summarized as follows:

1. Automation: Workflows can be executed automatically using a workflow engine which schedules the steps according to their interdependencies, collects the output of preceding

steps and passes the input to subsequent steps. Some workflow systems support the execution of process steps on different compute resources.

2. Scalability & HPC readiness: Workflows automatically exploit concurrency of steps that do not exchange data, i.e., that are not directly connected in the workflow graph. High workflow concurrency can be used to scale up the application on an HPC cluster or on a large number of distributed resources (for example in computing clouds).
3. Reusability and complexity reduction: Data produced in every workflow step can be seamlessly reused in subsequent steps. Thus, workflows can be nested, i.e. a sub-workflow can often be reused without modifications in a workflow for another application.
4. Provenance: Workflows are persistent objects providing metadata and methods to track the origin of data and code. A workflow can be automatically reproduced and thus be used for validation purposes. Thus, a workflow documents a simulation or data analysis and consequently improves reproducibility.
5. Reliability and resilience: Workflows provide mechanisms to authenticate users on the resources, track errors and recover from failure. Failures of single steps do not lead to invalidation of the whole workflow but only affected steps and their descendants have to be recovered.
6. Rapid prototyping: The workflow method enables multiscale modelling by reusing existing codes in a very flexible 'drag-and-drop' fashion using a specialized workflow editor and increases model development productivity.

3. Workflow frameworks in BIG-MAP

Despite these significant advantages, many academic groups still rely on script-based approaches to implement increasingly complex computational protocols. This is in part due to the lack of information on existing workflow frameworks that have been specifically developed for application in the natural sciences. In order to help close this gap we review here a few representative workflow environments, without any claim to be exhaustive.

3.1 Aiida

The Automated Interactive Infrastructure and Database for Computational Science (AiiDA - http://www.aiida.net) is an open-source Python infrastructure to help researchers with automating, managing, persisting, sharing and reproducing the complex workflows associated with modern computational science and all associated data.

AiiDA is built to support and streamline the four core pillars of the ADES model: Automation, Data, Environment, and Sharing. Key features include:

- **Workflows**: AiiDA allows to build and execute complex, auto-documenting workflows linked to multiple codes on local and remote computers.
- **High-throughput**: AiiDA's event-based workflow engine supports tens of thousands of processes per hour with full check-pointing.
- **Data provenance**: AiiDA automatically tracks and records inputs, outputs and metadata of all calculations and workflows in extensive provenance graphs that preserve the full lineage of all data.
- **Advanced queries**: AiiDA's query language enables fast graph queries on millions of nodes.
- **Plugin interface**: AiiDA can support via plugins any computational code and data analytics tool, data type, scheduler, connection mode, etc. (see public plugin repository)
- **HPC interface**: AiiDA can seamlessly deal with heterogeneous and remote computing resources; it works with many schedulers out of the box (SLURM, PBS Pro, torque, SGE or LSF).
- **Open science**: AiiDA allows to export both full databases and selected subsets, to be shared with collaborators or made available and browsable online on the Archive and Explore sections of Materials Cloud.
- **Open source**: AiiDA is released under the MIT open-source license.

AiiDA features a set of Python classes that provide access to the core AiiDA objects, namely calculations, code and data, and provides a high-level interface to computational resources via the AiiDA Daemon as well as Data handling by utilizing an Object-Relational Mapper (ORM) that maps AiiDA objects onto Python classes. Users can interact with AiiDA by using the command-line client verdi, an interactive Python shell, Python scripts, or the user-friendly AiiDAlab jupyer enivonrment. New components are incorporated using a Plugin registry (https://aiidateam.github.io/aiida-registry/) in the form of Python modules using subclasses of the AiiDA classes. Data can easily be shared between various AiiDA instances through import and export. Public AiiDA databases are also accessible through the Materials Cloud (www.materialscloud.org) that provides archival dissemination of raw or curated data. The full description of AiiDA can be found in[11,15], and of AiiDAlab in[16].

3.2 ASE

The Atomic Simulation Environment (ASE) has been developed as a tool to work with structures independently from the calculation engine in use. ASE currently supports 43 atomistic codes (https://wiki.fysik.dtu.dk/ase/). Thanks to its capability of reading, working, and writing a variety of

structure formats, ASE is highly interoperable and can already work with some of the workflow engines described here. While other workflow engines, such as AiiDA, include structure manipulation, workflow systems, error handling, data storage, in a single package, ASE focuses on the structural manipulation and interface with the calculation engine. MyQueue (https://myqueue.readthedocs.io/en/latest/) is the workflow system, which is interfaced with ASE. Thanks to the modularity of ASE, MyQueue can be interfaced with all the codes working with ASE. At the moment, examples of MyQueue workflows have been developed for GPAW and VASP. MyQueue is entirely written in Python and easily allows to concatenate tasks, check the status of calculations, and resubmit calculations if simple errors occur (time-out/out of memory). By not storing data, MyQueue does not need the installation of a database system, a provenance track has not been implemented yet.

3.3 Pipeline Pilot

Pipeline Pilot is a commercial Workflow engine initially developed by SciTegic, but subsequently acquired by BIOVIA, which is now part of Dassault Systèmes (http://www.3dsbiovia.com/products/collaborative-science/biovia-pipeline-pilot/)[17]. Pipeline Pilot provides a graphical User interface in which components are combined via drag and drop into workflows. Components represent data handling actions such as loading, filtering, combination or manipulation of data. Complete workflows can be incorporated as a component into other workflows. The connection between components are referred to as pipes and represent the Dataflow. Materials Studio (http://www.3dsbiovia.com/products/collaborative-science/biovia-materials-studio/) provides a large set of components relevant for material simulation and modelling that can be combined into larger protocols with Pipeline Pilot. Next to existing components, external applications can be included as components in various ways (http://www.3dsbiovia.com/products/datasheets/integration-collection.pdf) including the incorporation of web services via the Simple Object Access Protocol (SOAP)[18]. Due to the commercial distribution and the graphical user interface, ease of use is a major focus of Pipeline Pilot. However reusability is limited as a license for the software is required.

3.4 SimStack

Simstack (www.simstack.eu) implements a server-client workflow architecture to realize automation, reusability, reproducibility, transferability, and rapid prototyping in simulation protocols. SimStack features a graphical workflow editor based on PyQt which runs on several platforms (Linux, Windows, Mac). It allows efficient implementation, adoption, and execution of complex and extensive simulation workflows. SimStack saves time, hides the complexity of high-performance computing on remote resources, and allows users in the academy or industry to incorporate competitive edge models and scalable scientific simulations into their virtual design process. Furthermore, it provides a highly flexible drag-and-drop environment that allows the quick adaptation of existing workflows to develop custom solutions fitting the user needs.

The workflow elements are incorporated using Workflow Active Nodes (WaNos), comprising XML files defining the expected input and output and adjustable parameters. Thereby, users can easily develop and include WaNos in SimStack Workflows without requiring extensive programming skills. This feature permits rapid prototyping in the sense that it is easy to incorporate new modules ad hoc into the system.

SimStack splits the whole virtual design process into Client and Server operations. The Client is executed on the laptop and is dedicated to define and control the workflows. In the client, many modules are connected into complex workflows using drag and drop, and the main parameters are set using the automatically generated graphical user interface (GUI). The SimStack Client automatically connects via *SSH* to the SimStack Server installed on computational resources. The server handles the simulations managing file transfer, submission, monitoring workflows, and downloading the results files to the client user machine.

Figure 2 introduces one of Simstack use cases, where the complexity of the multiscale problem is hidden from the user by automatically interfacing the required applications for each step. After the initial workflow assembly, the only significant remaining input is the structure of the initial molecule.

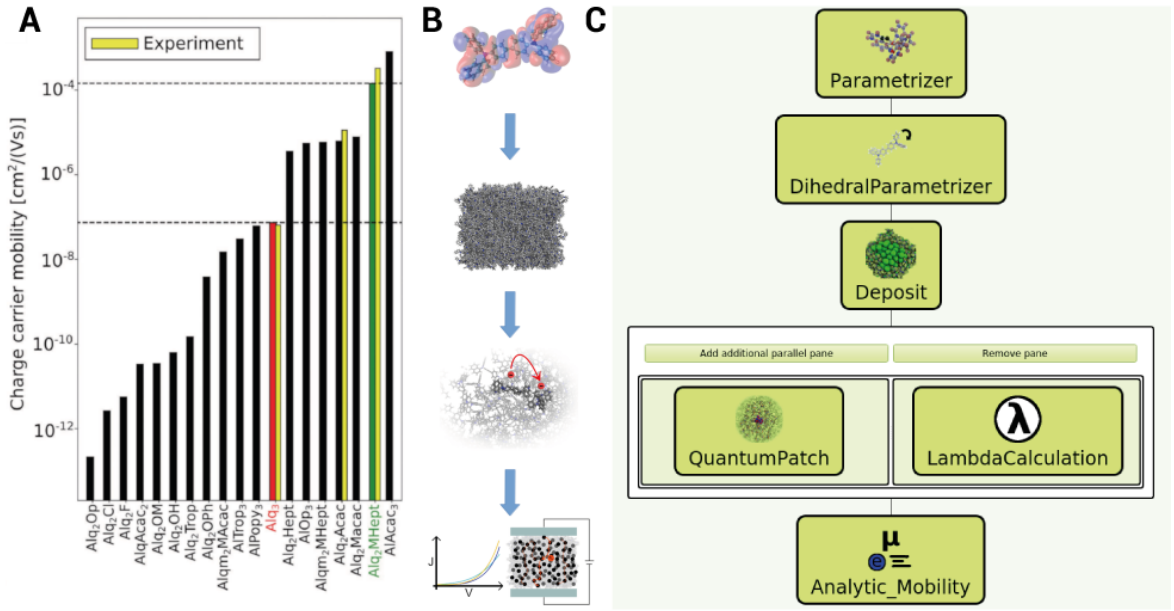

Figure 2. Screening of the charge carrier mobility of derivatives of the ALQ3 material. A Using a multi-scale workflow a novel material (green) could be developed exhibiting a two orders of magnitude higher mobility than the original material (red)[19]. B The workflow consists of a parametrization stage based on single molecule DFT. A forcefield is generated, which is then used to generate thin-film morphology using simulated PVD[20]. For each pair in this morphology the electronic structure is relaxed in a self-consistent way[21]. Finally the mobility is calculated using a generalized effective medium model[22]. C Implementation of the workflow in the SimStack workflow application.

4. BIG-MAP App Store

4.1 Concepts

The BIG-MAP App Store (https://big-map.github.io/big-map-registry/) (see Figure 2) comprises the following three components:

1. A public GitHub repository serving as a primary registry for the different apps

2. An automatically generated front-end page for the BIG-MAP App Store listing all the available apps and capabilities
3. A well-defined schema for the metadata associated with the app and the documentation

The BIG-MAP App Store can be reached from the BIG-MAP website and Sharepoint. At this moment, being at its infancy, it will be accessible only for the partners of BIG-MAP. The App Store will however be made accessible to the community with different level of openness: from publicly available data and codes, to limited access to the BIG-MAP members, to restricted access, which allow only the codes to run, but not to download the raw data. This multi-level openness ensure a protection of the confidential data.

4.1.1 Public GitHub repository

The BIG-MAP GitHub registry for an app is available at https://github.com/BIG-MAP/big-map-registry. Apps are added to the registry by adding an entry to the apps.yaml file within this repository, and any partner can also propose a new app category to be added to category.yaml before or after adding the app.

The protocol for a new entry to be added is this:

1. BIG-MAP partner creates a pull request to this repository that adds a new entry to the apps.yaml file, e.g., by editing the file directly in the browser. An example, following the schema mentioned above and detailed further below, would be:

```yaml
my-big-map-app:
  metadata:
    title: MyBIG-MAP app
    description: |
        My BIG-MAP app helps to promote accelerated discovery
        of novel battery materials.
    authors: A. Doe, B. Doe
    external_url: http://my-app.example.com
    documentation_url: https://my-big-map-app.readthedocs.io
    logo: https://github.com/my-org/my-big-map-app/raw/main/logo.png
    state: development
    version: '1.1'
  categories:
    - technology-aiida
    - technology-ase
    - quantum
```

1. Note: Only the metadata fields title and description are mandatory.

2. The app will show up in the BIG-MAP App Store once the pull request is approved and merged by the BIG-MAP maintainers.

The app store supports the $ref syntax to reference externally hosted documents. That means one can reference metadata that is hosted at a different location, which makes it easier to dynamically update it. For example, if one places a metadata.yaml file within the app repository, then one can reference that file in the app store with:

```yaml
my-big-map-app:
  metadata:
    $ref: https://github.com/my-org/my-big-map-app/raw/main/metadata.yaml
```

One can even reference only parts of the metadata, as in:

```yaml
my-big-map-app:
  metadata:
    title: MyBIG-MAP app
    description:
      $ref: https://github.com/my-org/my-big-map-app/raw/main/metadata.yaml#description
```

The app store will assume that external references are in JSON format unless the referenced path ends with .yaml or .yml.
These below are the valid keys for app entries in apps.yaml:

Key	Requirement	Description
metadata	Mandatory	General description of the app (see below).
categories	Optional	If provided, must be one of the valid categories specified in categories.yaml.
git_url	Optional	Link to the source code git repository.

While these are the valid keys for the app metadata:

Key	Requirement	Description
title	Mandatory	The title will be displayed in the list of apps in the application manager.
description	Mandatory	The description will be displayed on the detail page of your app.
authors	Optional	Comma-separated list of authors.
logo	Optional	Url to a logo file (png or jpg).

state	Optional	One of - registered: lowest level - app may not yet be in a working state. Use this to secure a specific name. - development: app is under active development, expect the occasional bug. - stable: app can be used in production.
documentation_url	Optional	The link to the online documentation of the app (e.g. on Read The Docs).
external_url	Optional	General homepage for your app.

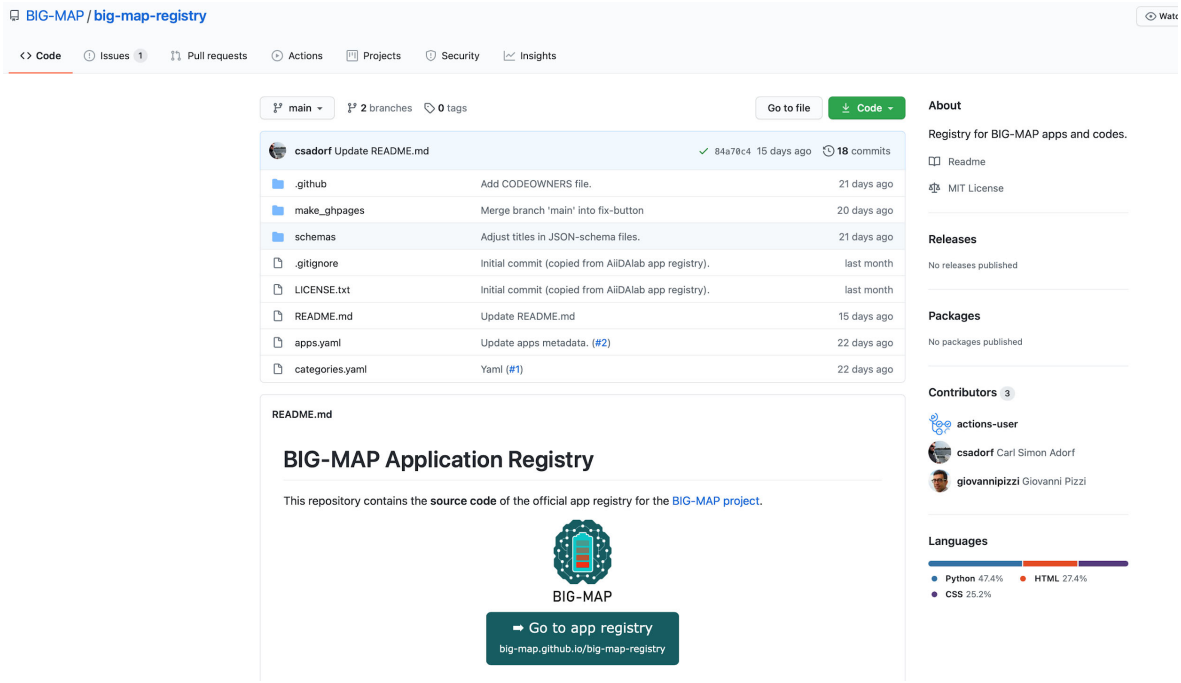

Figure 2. BIG-MAP Application Registry on GitHub.

4.1.2 Front-end for the BIG-MAP App Store

The front-end for the BIG-MAP App Store is automatically generated at https://big-map.github.io/big-map-registry/ and shows all the applications that have been uploaded and approved by the moderators after checking for functionality.

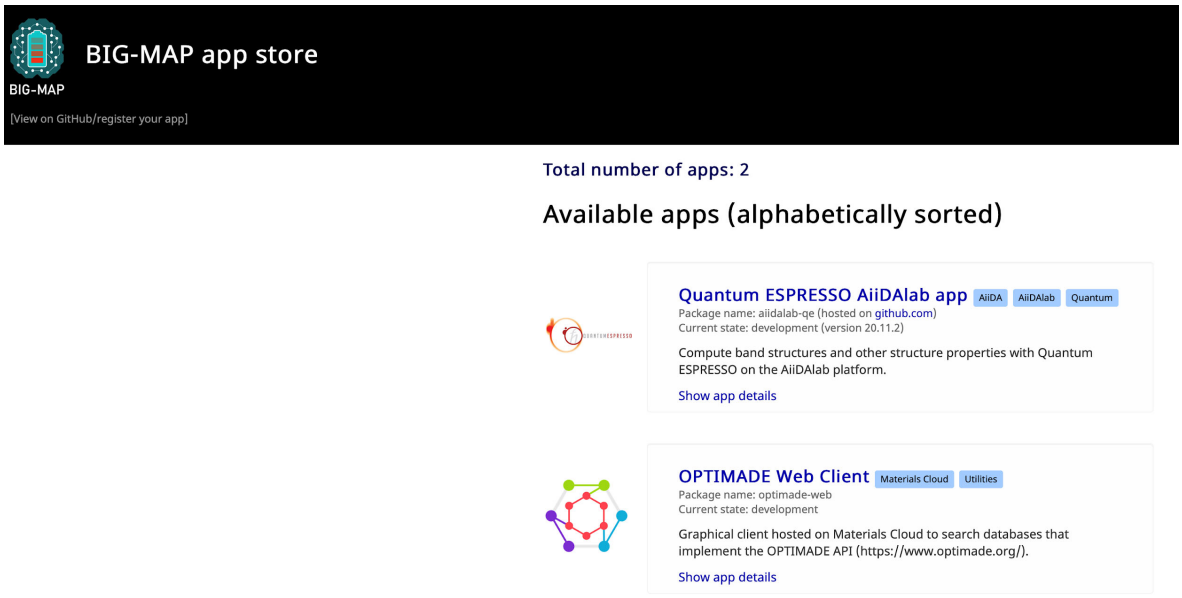

Figure 3. Landing page of the BIG-MAP App Store.

4.1.3 Schema for the metadata and for the documentation

The schema for metadata associated with apps and documentation are defined as versioned JSON-schema documents (https://json-schema.org/specification.html). These schemas are automatically published as part of the app store website, and are used to automatically validate all new entries prior to adding them to the app registry, ensuring that the registry is meeting the defined standard at all times. This is to ensure consistency and serves as an API contract between the app store host and consumers.

4.1 Deposition of apps

- Each app entry must have a title and a description as the only mandatory fields required for each entry. Any additional metadata, such as a version, or a URL, are optional to keep the entry barrier for adding an app to the registry as small as possible.
- Certain app store consumers, such as an AiiDAlab deployment, might require additional keys, such as a repository URL (to download the app).
- The app metadata schema can be revised over the lifetime of the project in a backwards compatible manner, e.g., via the addition of additional optional keys, to enable new capabilities.
- The app registry supports the use of references using the special $ref key. This allows app providers to dynamically update their app registry entry by hosting the app metadata (or

parts of it) in a different location under their control, without the need to go through the general app approval process for each change.

5. Examples for workflows and workflow components

5.1 Example 1: Quantum ESPRESSO AiiDAlab app

The Quantum ESPRESSO AiiDAlab app (*short:* QE app) is a Jupyter-based web application developed and executable within the AiiDAlab environment[16]. The app features an intuitive and simple interface that enables users who are not necessarily familiar with the intrinsics of executing DFT calculations with Quantum ESPRESSO, to compute, e.g., band structures for structures obtained from various input sources.

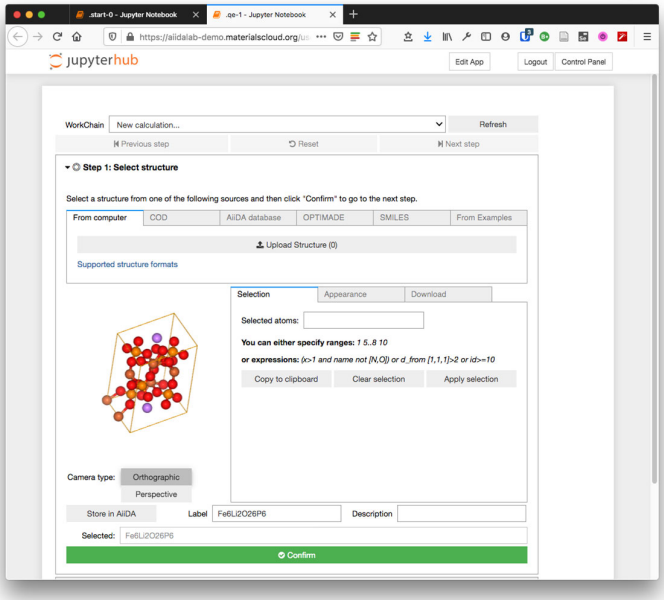

Figure 4: The Quantum ESPRESSO AiiDAlab app allows for the simple and interactive calculation of materials properties using the Quantum ESPRESSO on structures from various input sources, including uploaded files in all common formats, the OPTIMADE client, among others.

5.2 Example 2: OPTIMADE App and Web Client

The OPTIMADE App and Web Client allows to probe any public database complying with the universal OPTIMADE REST API (https://www.optimade.org). The specifications are agreed upon and updated yearly by a consortium of European and American partners that include most or all of the major players in the field (The Materials Project, AFLOW, OWMD, Materials Cloud, NoMAD, ….).

The full specifications, from the website above, are available here:

https://petstore.swagger.io/?url=https://raw.githubusercontent.com/Materials-Consortia/OPTIMADE/master/schemas/openapi_schema.json

We present in Figure 5 an example, where an inorganic material containing Li and Co is searched for in the Materials Project, giving rise to 3322 entries that can be browsed or downloaded. We are currently working at the integration of this with the QE App, allowing for the seamless capability to do an actual quantum mechanical simulation on the desired structure.

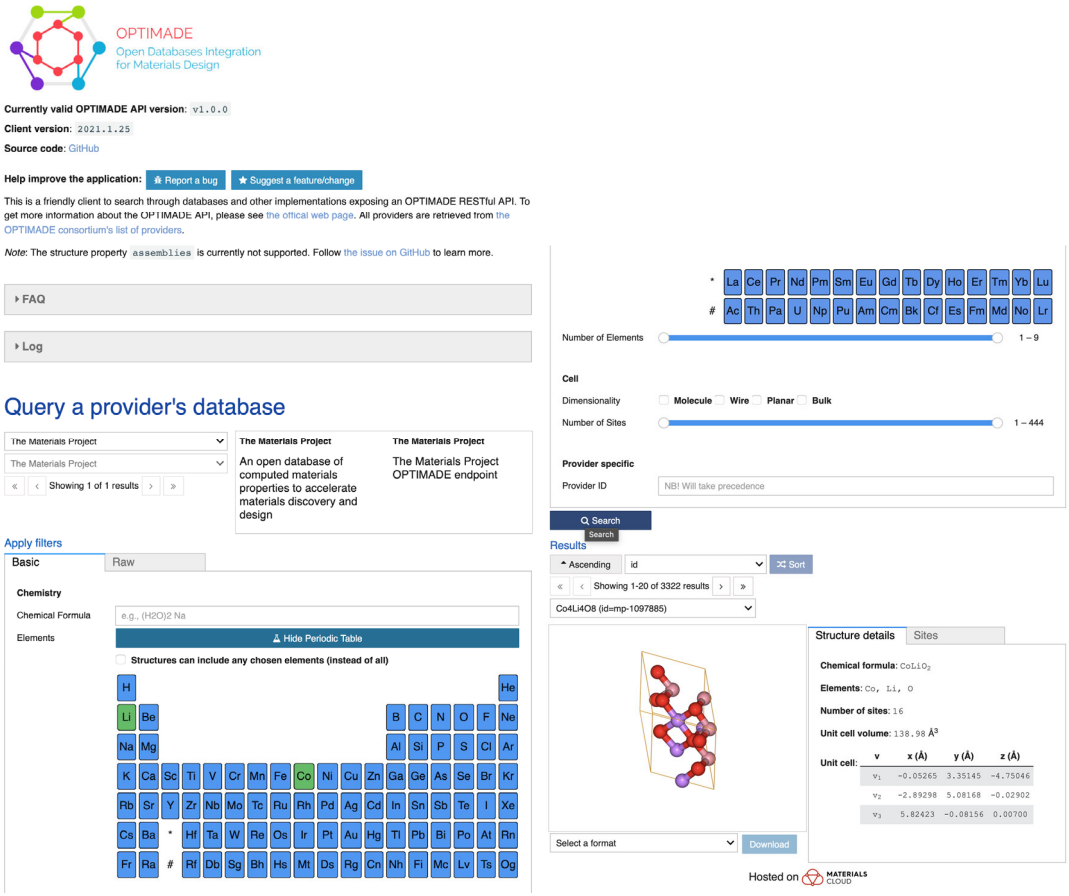

Figure 5. An interactive search of the Materials Project structure data (one of the many providers supporting the OPTIMADE universal REST API) through the OPTIMADE App. Searching for Li- and Co-containing inorganics has led to 3322 possible different entries.

5.3 Example 3: Workflow Component for Density Functional Calculations

The DFT_VASP WaNo implements a wide range of methods available within the VASP code, one of the most widely used electronic structure programs[23,24]. This WaNo affords experienced and inexperienced users to perform DFT calculations without requiring a deep understanding of VASP functionalities and specifications. The POSCAR file is the single mandatory file as the input of the WaNo. All the remaining VASP input files are automatically generated.

Figure 6 shows that this WaNo has the INCAR, KPOINTS, Analysis, and Files_Run tabs. The first one aims to define the INCAR file. This tab might be changed whenever the problem requires more

input parameters, which is attained by adding the necessary flags in the XML file. The remaining tabs create the KPOINTS file, allows the user to perform DOS and Bader Charge Analysis, and the Files_Run controls the VASP compilation types (vasp_std, vasp_gam, and vasp_ncl). It loads the POSCAR file and, as an option, may load INCAR, POTCAR, KPOINTS too.

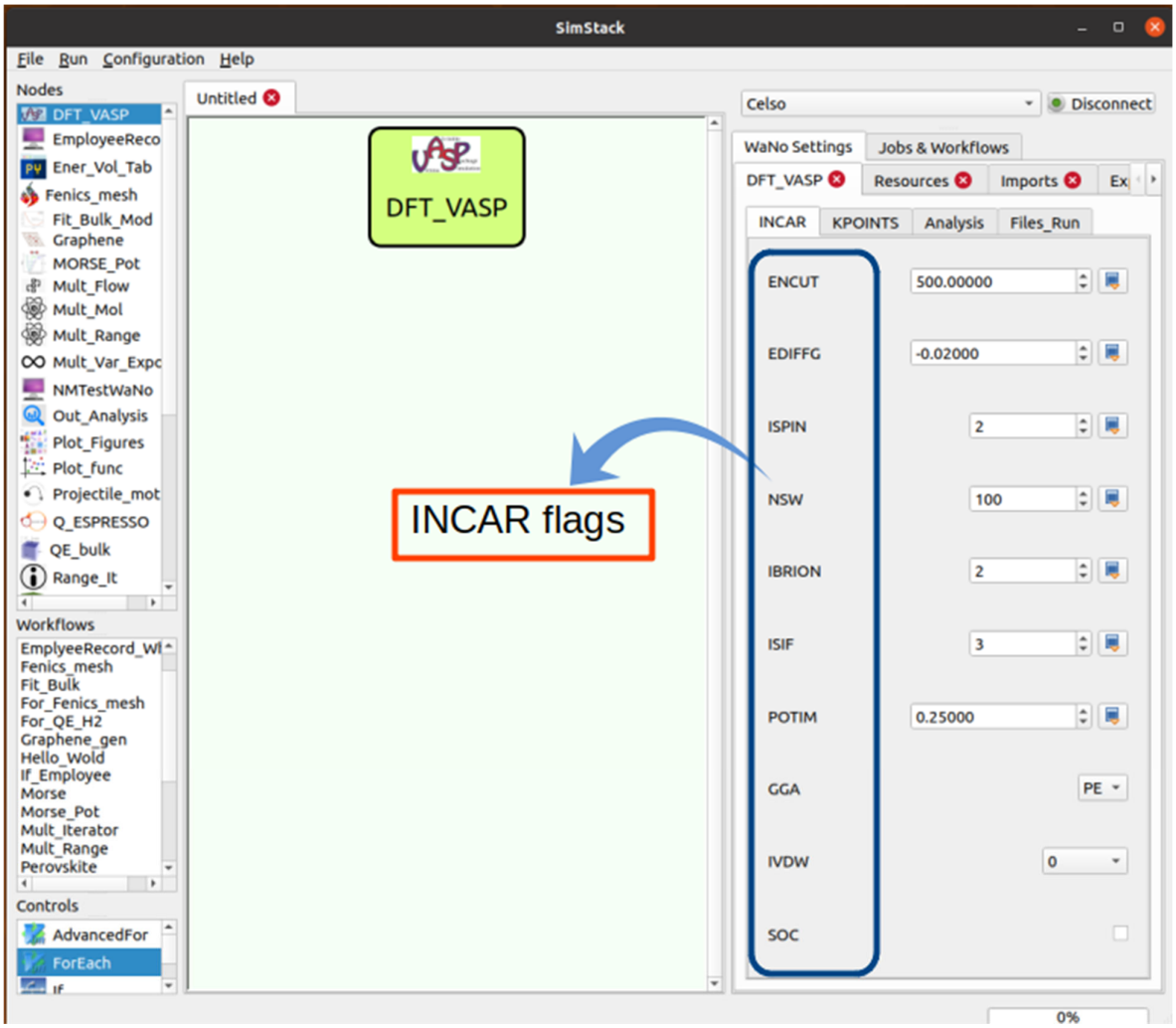

Figure 6. The DFT_VASP WaNo performs DFT calculation using Vasp code. In this WaNo, the POTCAR might be automatized after reading the POSCAR file. In the GUI, we can set the KPOINTS and INCAR files, but there is also the option to load the inputs file in the Files_Run tab.

5.4 Example 4: Surface Energy Workflow

In this workflow, we use the SimStack framework features to perform as an option a single shot DFT calculation of molecules absorbing on a surface. Here, we combine four different WaNos: Mult_Mol, Surface, DFT_VASP, and Table_Generator, to set up a molecule position on the surface, surface type, load molecules file structures, and choose the methods embedded in the DFT approach using VASP code. A table containing the system's total energy, molecule label, and molecule position on the surface is the expected output of this protocol.

Using the drag-and-drop environment of SimStack, we can build the Workflow depicted in Figure 7 in four steps. The Mult_mol WaNo accounts for the number of different positions for each molecule on the surface. In the second step, we add the Surface WaNo inside the ForEach loop control to generate the POSCAR files of adsorbed molecules to the chosen surface. In the third step, we insert the DFT_VASP WaNo, which will receive the generated files from the previous WaNo. At this step, we can take advantage of the parallelization in the HPC remote resources once the ForEach loop control is designed for this end. Table_Generator WaNo extracts three variable values on the OUTCAR file: the output file of steps two and three. This WaNo builds a table named Table_var in CSV format at the end of the protocol.

Using SimStack framework features, we built a simple, intuitive, and powerful workflow. Allowing us to explore the nature of molecules adsorbing on a surface, whether by assessing several geometric configurations of the same molecule or evaluating the energy of different molecules adsorbed on a chosen surface.

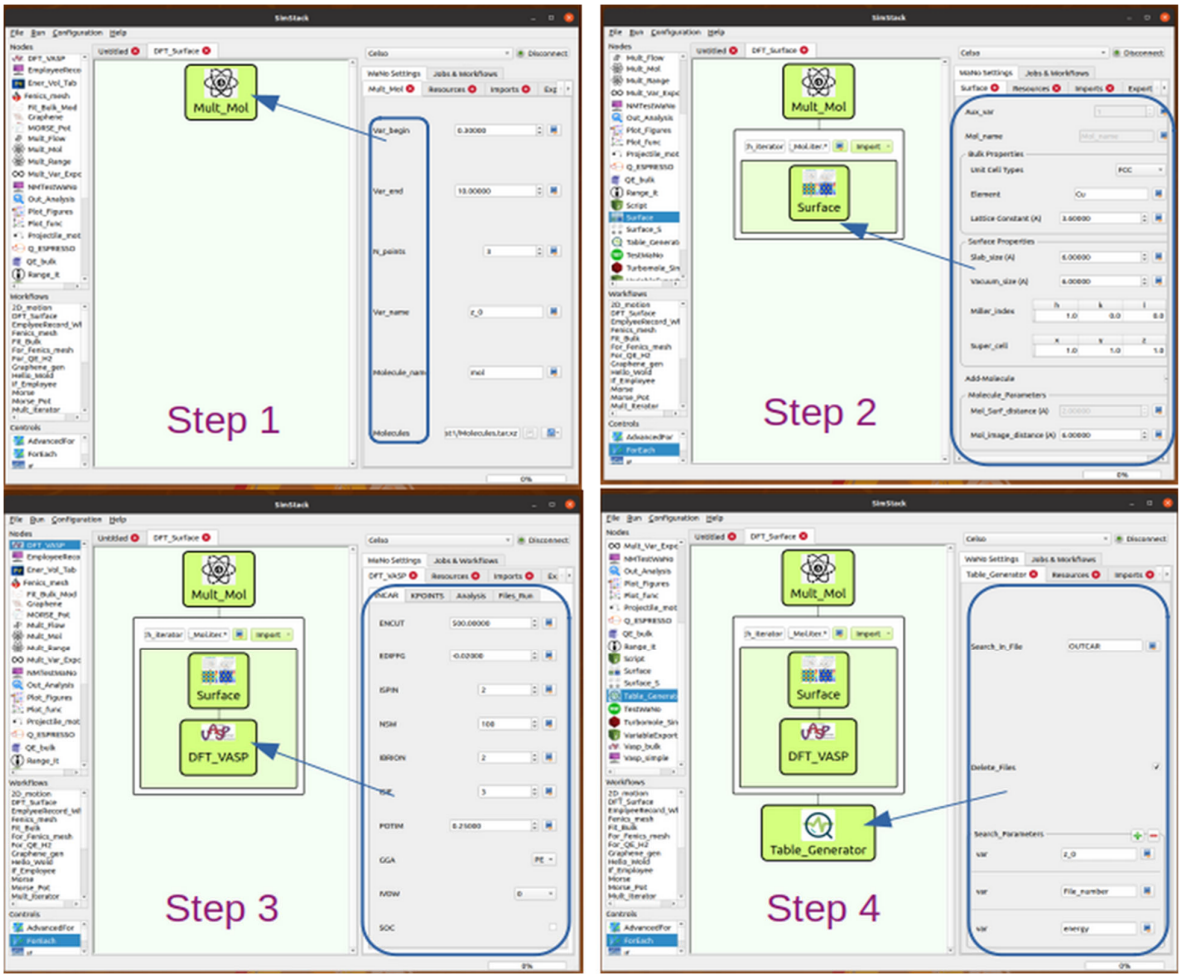

Figure 7. This workflow aims to perform several DFT calculations of molecules absorbing on a given surface. It is composed of Mult_Mol, Surface, DFT_VASP, and Table_Generator WaNos connected by the ForEach loop control. In step 1, we generate the number of points over the surface, where the molecule will be added. Steps 2 and 3 define the surface type and the DFT calculation methods employed in the simulation. The WaNo in the last step extracts the inquired variables of the output file from the previous steps.

5.5 Example 5: HTSpectra - a software for high-throughput analysis of spectra

Operando spectroscopies are essential tools for exploring the dynamic processes unfolding during the operation of Li battery materials. Currently available spectroscopic hardware enable recording hundreds of spectra during a single cycle; however, in most cases only few samples are manually analyzed. Manual spectral analysis is cumbersome, overlooks rich details hidden in the spectral trends and might hamper reproducibility. As a remedy, we are developing HT-Spectra, a Jupyter-based GUI powdered by a Python-based kernel for visualization and automated analysis of high-throughput (HT) spectra (Figure 8).

Figure 8. The interface of HTSpectra

The interface enables the user to load a spectral dataset, visualize each spectrum and tune processing parameters for baseline subtraction and peak fitting. The optimal parameters are used to process every spectrum in the dataset. Finally, the software outputs the processed spectra and fitting parameters in a ready-to-plot format to investigate the spectral trends. HT-Spectra is:

- User-friendly: The Jupyter GUI only exposes to the user an intuitive layout of widgets - no code.
- Developer-friendly: The code is modular, easily supporting the creation of new modules (e.g. for Machine Learning).
- Reproducible: The parameters used for analysis are saved in a metadata file, so the process can be reproduced from raw data.
- Efficient: In an average laptop PC the Python kernel process hundreds of spectra within seconds.

6. Outlook

With the work reported here we have laid the ground-stone for a growing, communitywide effort to generate transferable and reusable workflows and workflow components for the battery community. The big-map project will take the lead to establish this technology in many work packages, led by WP3. We foresee the incorporation of not only purely simulation-based workflows, but also of workflows integrating the control and evaluation of experiments, towards the end of the project, also on-the-fly.

It should be noted that there is little conceptual difference between the remote execution of an experiment, which relies on inputs and control parameters in a similar fashion as a simulation protocol, and a complex simulation on a remote computational resource. Broadly speaking both return data as a result of the request. Implementing the app store therefore is also the basis to gather components for automated discovery involving experiments.

In terms of exploitation it should be noted that workflow components and workflows come with individual licenses which may be different from the original licenses, as permitted. The BIG-MAP project is committed to generate as much open-source content as possible, but even open-source workflow components could be integrated into commercial workflows, if the license permits such use. For this reason, simulation solutions developed in BIG-MAP can be made available in the EU marketplace projects for the transfer software to partners outside of the consortia, which generated the software.

We see the BIG-MAP app store as a pilot for other initiatives in the context of battery 2030+ and beyond. The technology developed here will be made available, and wherever possible, co-developed with other initiatives to generate novel computational tools of unprecedented power, which can be used outside of the projects which originally generated these computational solutions.

7. References

(1) Giannozzi, P.; Baroni, S.; Bonini, N.; Calandra, M.; Car, R.; Cavazzoni, C.; Ceresoli, D.; Chiarotti, G. L.; Cococcioni, M.; Dabo, I.; Corso, A. D.; Gironcoli, S. de; Fabris, S.; Fratesi, G.; Gebauer, R.; Gerstmann, U.; Gougoussis, C.; Kokalj, A.; Lazzeri, M.; Martin-Samos, L.; Marzari, N.; Mauri, F.; Mazzarello, R.; Paolini, S.; Pasquarello, A.; Paulatto, L.; Sbraccia, C.; Scandolo, S.; Sclauzero, G.; Seitsonen, A. P.; Smogunov, A.; Umari, P.; Wentzcovitch, R. M. QUANTUM ESPRESSO: A Modular and Open-Source Software Project for Quantum Simulations of Materials. *Journal of Physics: Condensed Matter* **2009**, *21* (39), 395502. https://doi.org/10.1088/0953-8984/21/39/395502.

(2) Zographos, N.; Zechner, C.; Martin-Bragado, I.; Lee, K.; Oh, Y.-S. Multiscale Modeling of Doping Processes in Advanced Semiconductor Devices. *Materials Science in Semiconductor Processing* **2017**, *62*, 49–61. https://doi.org/10.1016/j.mssp.2016.10.037.

(3) Buehler, M. J. Materials by Design—A Perspective from Atoms to Structures. *MRS Bulletin* **2013**, *38* (2), 169–176. https://doi.org/10.1557/mrs.2013.26.

(4) Lula, R.; Baas, A. *What Makes a Material Function? Let Me Compute the Ways: Modelling in*

FP7 NMP Programme Materials Projects; Publications Office of the European Union: Luxembourg, 2012.

(5) Green, M. L.; Choi, C. L.; Hattrick-Simpers, J. R.; Joshi, A. M.; Takeuchi, I.; Barron, S. C.; Campo, E.; Chiang, T.; Empedocles, S.; Gregoire, J. M.; Kusne, A. G.; Martin, J.; Mehta, A.; Persson, K.; Trautt, Z.; Duren, J. V.; Zakutayev, A. Fulfilling the Promise of the Materials Genome Initiative with High-Throughput Experimental Methodologies. *Applied Physics Reviews* **2017**, *4* (1), 011105. https://doi.org/10.1063/1.4977487.

(6) Jain, A.; Ong, S. P.; Hautier, G.; Chen, W.; Richards, W. D.; Dacek, S.; Cholia, S.; Gunter, D.; Skinner, D.; Ceder, G.; Persson, K. A. Commentary: The Materials Project: A Materials Genome Approach to Accelerating Materials Innovation. *APL Materials* **2013**, *1* (1), 011002. https://doi.org/10.1063/1.4812323.

(7) Hachmann, J.; Olivares-Amaya, R.; Atahan-Evrenk, S.; Amador-Bedolla, C.; Sánchez-Carrera, R. S.; Gold-Parker, A.; Vogt, L.; Brockway, A. M.; Aspuru-Guzik, A. The Harvard Clean Energy Project: Large-Scale Computational Screening and Design of Organic Photovoltaics on the World Community Grid. *The Journal of Physical Chemistry Letters* **2011**, *2* (17), 2241–2251. https://doi.org/10.1021/jz200866s.

(8) Gil, Y.; Deelman, E.; Ellisman, M.; Fahringer, T.; Fox, G.; Gannon, D.; Goble, C.; Livny, M.; Moreau, L.; Myers, J. Examining the Challenges of Scientific Workflows. *Computer* **2007**, *40* (12), 24–32. https://doi.org/10.1109/mc.2007.421.

(9) Deelman, E.; Gannon, D.; Shields, M.; Taylor, I. Workflows and E-Science: An Overview of Workflow System Features and Capabilities. *Future Generation Computer Systems* **2009**, *25* (5), 528–540. https://doi.org/10.1016/j.future.2008.06.012.

(10) Ludäscher, B.; Altintas, I.; Berkley, C.; Higgins, D.; Jaeger, E.; Jones, M.; Lee, E. A.; Tao, J.; Zhao, Y. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice and Experience* **2006**, *18* (10), 1039–1065. https://doi.org/10.1002/cpe.994.

(11) Pizzi, G.; Cepellotti, A.; Sabatini, R.; Marzari, N.; Kozinsky, B. AiiDA: Automated Interactive Infrastructure and Database for Computational Science. *Computational Materials Science* **2016**, *111*, 218–230. https://doi.org/10.1016/j.commatsci.2015.09.013.

(12) Bender, A.; Poschlad, A.; Bozic, S.; Kondov, I. A Service-Oriented Framework for Integration of Domain-Specific Data Models in Scientific Workflows. *Procedia Computer Science* **2013**, *18*, 1087–1096. https://doi.org/10.1016/j.procs.2013.05.274.

(13) Lutz, B.; Sinner, C.; Bozic, S.; Kondov, I.; Schug, A. Native Structure-Based Modeling and Simulation of Biomolecular Systems per Mouse Click. *BMC Bioinformatics* **2014**, *15* (1), 292. https://doi.org/10.1186/1471-2105-15-292.

(14) Wilkinson, M. D.; Dumontier, M.; Aalbersberg, Ij. J.; Appleton, G.; Axton, M.; Baak, A.; Blomberg, N.; Boiten, J.-W.; Santos, L. B. da S.; Bourne, P. E.; Bouwman, J.; Brookes, A. J.; Clark, T.; Crosas, M.; Dillo, I.; Dumon, O.; Edmunds, S.; Evelo, C. T.; Finkers, R.; Gonzalez-Beltran, A.; Gray, A. J. G.; Groth, P.; Goble, C.; Grethe, J. S.; Heringa, J.; Hoen, P. A. C. 't; Hooft, R.; Kuhn, T.; Kok, R.; Kok, J.; Lusher, S. J.; Martone, M. E.; Mons, A.; Packer, A. L.; Persson, B.; Rocca-Serra, P.; Roos, M.; Schaik, R. van; Sansone, S.-A.; Schultes, E.; Sengstag, T.; Slater, T.; Strawn, G.; Swertz, M. A.; Thompson, M.; Lei, J. van der; Mulligen, E. van; Velterop, J.; Waagmeester, A.; Wittenburg, P.; Wolstencroft, K.; Zhao, J.; Mons, B. The FAIR Guiding Principles for Scientific Data Management and Stewardship. *Scientific Data* **2016**, *3* (1). https://doi.org/10.1038/sdata.2016.18.

(15) Huber, S. P.; Zoupanos, S.; Uhrin, M.; Talirz, L.; Kahle, L.; Häuselmann, R.; Gresch, D.; Müller, T.; Yakutovich, A. V.; Andersen, C. W.; Ramirez, F. F.; Adorf, C. S.; Gargiulo, F.; Kumbhar, S.;

Passaro, E.; Johnston, C.; Merkys, A.; Cepellotti, A.; Mounet, N.; Marzari, N.; Kozinsky, B.; Pizzi, G. AiiDA 1.0, a Scalable Computational Infrastructure for Automated Reproducible Workflows and Data Provenance. *Scientific Data* **2020**, *7* (1). https://doi.org/10.1038/s41597-020-00638-4.

(16) Yakutovich, A. V.; Eimre, K.; Schütt, O.; Talirz, L.; Adorf, C. S.; Andersen, C. W.; Ditler, E.; Du, D.; Passerone, D.; Smit, B.; Marzari, N.; Pizzi, G.; Pignedoli, C. A. AiiDAlab – an Ecosystem for Developing, Executing, and Sharing Scientific Workflows. *Computational Materials Science* **2021**, *188*, 110165. https://doi.org/10.1016/j.commatsci.2020.110165.

(17) Warr, W. A. Scientific Workflow Systems: Pipeline Pilot and KNIME. *Journal of Computer-Aided Molecular Design* **2012**, *26* (7), 801–804. https://doi.org/10.1007/s10822-012-9577-7.

(18) Symalla, F.; Friederich, P.; Massé, A.; Meded, V.; Coehoorn, R.; Bobbert, P.; Wenzel, W. Charge Transport by Superexchange in Molecular Host-Guest Systems. *Physical Review Letters* **2016**, *117* (27). https://doi.org/10.1103/physrevlett.117.276803.

(19) Friederich, P.; Gómez, V.; Sprau, C.; Meded, V.; Strunk, T.; Jenne, M.; Magri, A.; Symalla, F.; Colsmann, A.; Ruben, M.; Wenzel, W. Rational In Silico Design of an Organic Semiconductor with Improved Electron Mobility. *Advanced Materials* **2017**, *29* (43), 1703505. https://doi.org/10.1002/adma.201703505.

(20) Neumann, T.; Danilov, D.; Lennartz, C.; Wenzel, W. Modeling Disordered Morphologies in Organic Semiconductors. *Journal of Computational Chemistry* **2013**, *34* (31), 2716–2725. https://doi.org/10.1002/jcc.23445.

(21) Friederich, P.; Symalla, F.; Meded, V.; Neumann, T.; Wenzel, W. Ab Initio Treatment of Disorder Effects in Amorphous Organic Materials: Toward Parameter Free Materials Simulation. *Journal of Chemical Theory and Computation* **2014**, *10* (9), 3720–3725. https://doi.org/10.1021/ct500418f.

(22) Rodin, V.; Symalla, F.; Meded, V.; Friederich, P.; Danilov, D.; Poschlad, A.; Nelles, G.; Wrochem, F. von; Wenzel, W. Generalized Effective-Medium Model for the Carrier Mobility in Amorphous Organic Semiconductors. *Physical Review B* **2015**, *91* (15). https://doi.org/10.1103/physrevb.91.155203.

(23) Kresse, G.; Hafner, J. Ab Initio Molecular Dynamics for Open-Shell Transition Metals. *Phys. Rev. B* **1993**, *48* (17), 13115–13118. https://doi.org/10.1103/PhysRevB.48.13115.

(24) Kresse, G.; Furthmüller, J. Efficient Iterative Schemes for Ab Initio Total-Energy Calculations Using a Plane-Wave Basis Set. *Phys. Rev. B* **1996**, *54* (16), 11169–11186. https://doi.org/10.1103/PhysRevB.54.11169.