INFRAEDI-02-2018

www.bioexcel.eu

BioExcel-2 Project Number 823830

D2.6 – Integration of BioExcel software framework with standards and international initiatives

WP2: Convergence of HPC/HPDA and Improved Usability

Document Information

Deliverable Number	D2.6
Deliverable Name	Integration of BioExcel software framework with standards and international initiatives
Due Date	2021-12-31 (PM36)
Deliverable Lead	UNIMAN
Authors	Stian Soiland-Reyes (UNIMAN) Adam Hospital (IRB) Genís Bayarri (IRB) Ania Niewelska (EMBL-EBI) Cibin Sadasivan Baby (EMBL-EBI) Douglas Lowe (UNIMAN) Pau Andrio (BSC) Josep Ll. Gelpi (BSC)
Keywords	Project Management
WP	WP2
Nature	Report
Dissemination Level	Public
Final Version Date	2021-12-21
Reviewed by	Bert de Groot (MPG) Alexandre Bonvin (UU)
MGT Board Approval	2021-12-22

Document History

Partner	Date	Comments	Version
UNIMAN	2021-11-15	First draft and structure	0.1
IRB, EMBL-EBI, UNIMAN, BSC	2021-12-07	Section content	0.2
UNIMAN, IRB	2021-12-09	Editorial changes, figures, references	0.3
UNIMAN	2021-12-17	Expanded on RO-Crate, CWFR	0.4
UNIMAN	2021-12-21	Revised following review	0.5

Executive Summary

This deliverable details the integration of the BioExcel computational framework with external initiatives and standards for the purpose of interoperability and portability, as well as wider FAIR aspects on discoverability and metadata.

An ongoing challenge for users to take advantage of BioExcel tools in workflow management systems and across multiple compute infrastructures is to get the software installed and integrated into their system, without losing the significant optimization performance gains developed by BioExcel.

Our work on forming the BioExcel Building Blocks (BioBB) have helped unify programmatic invocation of biomolecular simulation and modelling tools, making it easier to include them in major workflow systems in life sciences. Technically the building block functionality has been made possibly by building on multiple standards, technologies and initiatives for interoperability, as detailed in this deliverable.

This document expands on how we have addressed Exascale challenges such as supporting multiple hardware-optimized binaries for execution on a variety of cloud compute environments.

This is significant because the compute landscape has changed for BioExcel users towards a *hybrid* model of private clusters and public cloud usage. Workflows, which capture a computational method for reuse, need to be *portable* not just across these compute backends but also between different workflow systems. This is important to reliably apply a workflow through a wide range of scenarios: from training and prototyping, through development and customization towards scalable production runs.

We have identified and generalized another advantage of what we call *canonical workflow building blocks*, that a conceptual workflow using a combination of blocks can be realized the same way in multiple workflow systems and give reproducible results, thanks to their common behaviour and underlying software packaging mechanisms.

We show how our active engagement with ongoing initiatives have influenced WP2 development, publication of workflows and the BioExcel Building Blocks, as well as further evolved the standards themselves.

Table of Contents

1 Making BioExcel tools available

1.1 BioConda and Containers

As presented in deliverable *D2.3 – First release of demonstration workflows* [BioExcel-D D2.3], ease of installation and deployment of our tools is at the core of the BioExcel Building Blocks library [Andrio 2019]. BioBB strongly relies on containerization and packaging. Thus, all the BioBB modules (categories) are available through BioConda packages and Docker/Singularity containers. This containerization and packaging of the individual categories strongly reduces the time and effort needed to install the required dependencies to run biomolecular simulation workflows like the ones developed by the project. Clear examples of this can be found in our collection of Jupyter Notebook demonstration workflows.

Besides the generation of packages and containers for the BioBB modules, BioExcel is also contributing in the sharing and reproducibility of different biomolecular simulation tools that were previously only available from the source code. Examples of these tools are Curves+ (BioConda), a tool to analyse DNA structure flexibility through its helical parameters, CMIP (BioConda) (Classical Molecular Interaction Potentials), a tool to compute molecular interactions, or GOdMD (BioConda), a tool to compute conformational transition trajectories of protein structures. Having all these tools packaged speeds up the process of wrapping and integration in the BioBB library, as the BioConda package can be included as a software dependency.

An extreme case has been the packaging of our key application GROMACS [Abraham 2015]. The GROMACS tool has been packaged up, both into Docker containers and into Conda packages. In both cases optimized GROMACS binaries have been provided for a range of SIMD types (SSE2, AVX_256, and AVX2_256), to facilitate efficient computation. An environment *activation script* is used to select at runtime the appropriate binary for the system that is being used.

Building these optimized binaries has required us to push the build systems to their limit, as the build process must be repeated for each binary. The CircleCI build system used by the *bioconda* channel has limited build times, so to fit within these we had to focus only on these three SIMD types, and one MPI implementation (and one non-MPI build) – which we believe will be the options which prove most useful to the majority of users.

For building the Docker containers we had to move to using GitHub Actions in a staged process, rather than the DockerHub build system, in order to build the necessary multiple binaries in a staged process.

The different build systems have different support for GPU libraries. The bioconda build system does not currently provide CUDA support, so the conda packaged binaries use OpenCL to provide GPU support. We are investigating a move to the CondaForge repository, with a more permissive build system based on Azure (6 hour build limit) this may alleviate this limitation and allow more SIMD types and potentially CUDA support.

The GitHub Actions build system already supports building Docker containerised binaries with CUDA support. The resulting gromacs container has also been tested with Windows 10's recent WSL2 support for CUDA.

We have documented this approach to building hardware optimized containers and packages in a series of blog posts [Long 2020, Long 2021].

1.2 Bio.tools, EDAM

The BioExcel Building Blocks library and its associated tutorials and demonstration workflows are all included in the ELIXIR catalogue of bioinformatic tools bio.tools:

- BioBB main library
- BioBB with CWL
- BioBB in Command-Line
- BioBB through REST-API
- BioBB demonstration workflows:
  - Protein MD Setup (Gromacs)
  - Protein-Ligand complex MD Setup (Gromacs)
  - Automatic Ligand Parameterization
  - Mutation Free Energy Calculations
  - Protein-Ligand Docking
  - Protein MD Setup (Amber)
  - Protein-Ligand complex MD Setup (Amber)
  - Constant pH MD Setup (Amber)
  - ABC (Ascona B-DNA Consortium) DNA MD Setup (Amber)
  - Structural DNA helical parameters from MD trajectory

The main BioBB entry in bio.tools is updated with every new GitHub release of building blocks (latest v3.7, 2021.3). New demonstration workflows are registered once released.

Proper registration of BioExcel tools in bio.tools required an extension of the available EDAM terms, in particular for the structural (and MD) bioinformatic terms. Back in 2019, BioExcel contributed with a set of new terms related to the biomolecular simulation field (see edamontology/edamontology#353). Thanks to these new terms, most of the BioExcel tools and workflows can now be registered,

with adequate input/output data and formats, operations and topics, helping in the finding of similar and interoperable tools.

However, some of the suggested data types and formats were not approved at that time, with the main concern being that they were only used by a single tool (e.g. GROMACS *mdp* files: molecular dynamics parameters configuration file and GROMACS *tpr* files: GROMACS portable binary run input file) (see edamontology/edamontology#384). We had to accept the decision of the EDAM team, although some of these terms were important for our tools.

Now, 2 years after that, we have extended the use of the structural and biomolecular simulation EDAM terms to specify inputs and outputs of the BioBB CWL descriptions. The biobb_adapters module contains the CWL descriptions of all the developed BioBB building blocks.

In some of the cases, those terms that were discarded at that moment are now important for our CWL definitions. See as an example the grompp building block, that takes as an input an mdp (gromacs MD configuration file), and that we have been forced to define using the generic EDAM term for "*Textual format*" or "*Plain text*" (EDAM edam:format_2330) in grompp.cwl. This issue has been directed to the current EDAM board and it was agreed that BioExcel would build and present a new set of biomolecular terms, including all the considered important terms we have found during the development process of our BioBB library.

This example highlights one tension between usability and findability. Detailed markup of EDAM terms helps discovery of BioExcel tools, for instance BioBB shows up in Bio.Tools search for inputs of EDAM data_0883 (*Structure*). However, in workflow systems like Galaxy and CWL, which apply *type checking*, tools cannot be directly combined if they have inconsistent output/input type annotations (even if syntactically the tools may be compatible); so in workflow definitions it may be beneficial to use a less specific EDAM type than in the bio.tools registration.

1.3 Common Workflow Language (CWL)

The Common Workflow Language (CWL) is a community standard for defining interoperable computational workflows, which can be executed on a series of workflow engines and compute platforms [Crusoe 2022]. UNIMAN is a member of the CWL Leadership team.

As mentioned in deliverable on best practice guides [BioExcel-2 D3.5], BioExcel has developed guides for Creating workflows with Common Workflow Language and How to choose which CWL workflow engine to deploy building on WP2 effort and experience.

We have developed a tutorial on using the BioBB building blocks for Molecular Dynamics setup using CWL, based on the similar tutorial using Jupyter Notebook, as described in deliverable D2.3 (section 2.2.1).

We have also replicated in CWL a port of a scalable PyCOMPSs HPC workflow, using the BioBB building blocks tool descriptors. For this we have created one base, linear, workflow, which can then be wrapped by workflows using the *scatter* function to replicate the looping through workflows as done in the original PYCOMPSs scripts.

One benefit of porting PyCOMPSs workflows to CWL is to enable their use across a wider range of workflow engines and compute platforms, particularly moving from HPC to cloud. For BioExcel users this interoperability means that our *conceptual workflows* can be adapted without necessarily moving to a different workflow management system.

However, while PyCOMPSs has been shown to give significant performance for such workflows on HPC [Andrio 2021], similar benchmarks have not yet been measured for the many combinations of CWL engine and cloud backends; a start of this is planned for the BioExcel-2 extension period based on this port.

The writing of these workflows has required the use of the latest (v1.2) CWL standard [Amstutz 2020], to support conditional branching for generating protein mutations[1]. UNIMAN contributed to this highly requested update to the CWL standard as part of task T3.4 and to facilitate WP2 needs.

At the time of writing, the 1.2 release of the CWL standard is only supported by the major implementations (cwltool, Arvados, and Toil; see guide to CWL engines). However, we expect this limitation to ease as other engines progressively update their implementation to the latest standard.

We have identified CWL to be an important aspect of describing workflow structures even for workflow systems otherwise not yet implementing the standard. So-called abstract workflows, where placeholder *Operation*s are used instead of configured command line tools (also contributed to CWL 1.2 by BioExcel), were prototyped together with the WorkflowHub and the Galaxy workflow system (galaxy2cwl, since incorporated in Galaxy's tool gxformat2). These abstract CWL workflows are used together with the native workflow format

[1] We use the *'pickValue'* conditional when collecting outputs to return at the end of the workflow, selecting all non-*null* files to be returned to the user. Within the *Mutate* workflow we make use of the *'when'* logical operator to control the calling of the molecule mutation step. Combining this with the use of the *'pickValue'* conditional, this time selecting the first non-*null* file, within the list of inputs to pass to the main workflow, we can determine if the molecule being analysed requires mutating or not.

when registering a workflow in WorkflowHub, allowing a common rendering of inputs/outputs/steps/tools, improving from the current ad-hoc use of WorkflowHub collections.

Further work on linking abstract workflows with *canonical workflow building blocks* [Soiland-Reyes 2021] will allow common workflow registrations and FAIR metadata for BioBB-based workflows that are realized in multiple "flavours" for different workflow languages, but which shares a common conceptual structure. We have identified this use of *canonical workflows* as important for training purposes - e.g. a user learns to understand a BioBB workflow using Jupyter Notebook, then extends an equivalent HPC variant of the same workflow using CWL.

We are in the process of testing the portability and scalability of this workflow with different CWL engine configuration on multiple HPC systems. We have focused on the Toil workflow engine in our testing of CWL workflows on HPC. We have tested Toil with the SLURM job manager on two HPC systems (ARCHER2 and Mare Nostrum), and with the SGE job manager on one HPC system (CSF3, at University of Manchester). We will also continue testing using a recent PRACE allocation for BioExcel-2 on TGCC Joliet Curie.

By performing these tests we have discovered, and fixed, bugs in Toil, and also solved problems caused by site-specific configurations for job engines (showing that we have to be careful what functionality we assume that HPC administrators have enabled for their job engine).

We submitted a bug-fix for using SGE in Toil (DataBiosphere/toil#3410); as well as identifying a conflict between Toil and Slurm on ARCHER2 (and prototyping a solution), where setting the memory allocation had been disabled by the administrators (DataBiosphere/toil#3349).

On Mare Nostrum we found that the system administrators had removed access to the *'sinfo'* tool for SLURM, which Toil uses to gather information about the batch system. This was solved by the Mare Nostrum system administrators providing us with a shell script which spoofs the output from the *sinfo* tool that Toil requires. This has allowed *us* to run our workflows but, unlike the other fixes, this has not yet been made into a global fix for all users of Toil on this HPC system.

The reality is that different HPC systems always have potential kinks that can hinder portability and interoperability. For instance, on one HPC system we were able to submit a grid job of the workflow engine run, that then itself could submit further step jobs from the job node. This meant we were able to use Conda-installed version of a recent version of the workflow engine Toil, rather than

running the engine on the control node. But this is example of a feature that would not work universally, as system administrators of HPC systems add different firewall rules and restrictions, or may have deliberate or accidental inconsistencies between control nodes and job nodes.

We plan to add some of these recent experiences and corresponding recommendations to our existing BioExcel best practice guide on CWL engines to better highlight each engine's requirements, as well as detail the HPC challenges in a whitepaper currently being drafted (see section *CWL on HPC whitepaper*).

While such challenges have not greatly hindered our use of CWL and Toil on these HPC systems, they have highlighted that it would overall still be better for most users to have their HPC administrators install and verify their workflow engine. With CWL, this will still allow the user to keep control of installation of tool binaries such as with the BioBB building blocks, and combined with BioExcel effort users can still benefit from hardware-optimized Conda packages and containers.

Autogenerating CWL

Aiming to achieve BioBB—CWL full compatibility, the creation of the CWL tool wrappers had been automated. The auto-generation of the wrappers ensures that regular releases of BioBB (once every quarter) will have their corresponding updated CWL wrappers released in the biobb_adapters package.

The CWL-generator is part of the BioBB release protocols. This generator obtains all the information from the JSON description files of each tool. The JSON description files are created using a combination of code documentation scrapping and code introspection. Finally, the CWL wrappers are created using the JSON information and Python conditional templates.

The auto-generated CWL wrappers are then used in CWL workflows that are being tested in HPC and Cloud infrastructures (see sections below) and are also used to automatically convert BioBB Python pre-configured workflows into CWL workflows in the *BioBB-Workflows* web site (see section 2.4).

CWL execution on cloud and Kubernetes clusters

We have investigated the practicalities of running our CWL workflows on cloud infrastructure. As noted above, our choice of workflow engine has been limited by the requirement of support for the CWL v1.2 standard.

Cloud support by CWL engines

The major CWL implementations include a wide support for executing on cloud infrastructures, including AWS, Azure, Google Cloud Platform, OpenStack; in addition to traditional job management systems like Grid Engine, Slurm, PBS/Torque, LSF, Mesos and HTCondor. In this way of operating, a CWL engine like Toil will manage the cloud instances either to dynamically start a virtual cluster (Toil uses Apache Mesos), or use a pre-configured compute cluster (Arvados uses Slurm).

These two major modes of executing CWL workflows on the cloud cater well for ad-hoc workflow execution (nodes shutdown at end of each workflow) or continuous production runs of the same workflow (nodes remain available for a relatively fixed load)

However, these modes can be a challenge for instance for biomolecular simulation training events: on a dynamic cluster a sudden burst of multiple short, but compute intensive workflows could cause a costly large number of nodes to start and almost immediately shut down. A fixed cluster can be suboptimal with multiple concurrent users, as node-saturating compute-intensive workflows typically are executed one a time by engines. In a production setting this risk is typically managed by careful configuration of limits on cluster/job managers and restricting concurrent jobs within the workflow definition. Some workflow engines can also use long-living pilot jobs [Turilli 2019] as a way to reuse (or eventually time out) deployed nodes, but so far this is not commonly implemented by CWL engines.

For the extension period of BioExcel-2 we will explore deeper the matrix of CWL engines and their combined cloud support, using the BioBB workflows for benchmarking.

Kubernetes

From our conversations with industry users we have identified a large interest in Kubernetes (K8s) as an intermediate cluster layer that manages cloud instances and their executions. In one way Kubernetes offers orchestration similar to scientific workflow management systems, but primarily aimed at service deployment with horizontal scaling (e.g. dynamically add more MySQL instances) between network-linked containers. It is worth noting that traditional pipelines (command line jobs that exchange files) will in Kubernetes require additional long-running services to handle data transfer between steps, e.g. a data store, and pre/post data transfers.

Given the current industry interest in Kubernetes it is still of value to explore further how BioExcel workflows can utilise this as a compute backend.

Due to the limited access to public cloud resources and the interest also within the EMBL-EBI group of providing a shared Kubernetes infrastructure, we limited the investigation of CWL cloud execution to the engines offering Kubernetes support.

Future work may investigate rewriting BioExcel workflows to use the building blocks from a Kubernetes-native workflow system like Argo. As a first step we here approached CWL, for which we already have several BioExcel workflows defined, in order to investigate if its interoperability features also extended into Kubernetes.

CWL 1.2 on Kubernetes

The investigation concluded with *not* finding a single engine that would support both Kubernetes and CWL 1.2 standard and had no requirement on public cloud infrastructure.

ReaNa & cwl-tes offered a potentially good Kubernetes support, but lacked the support for CWL 1.2. This is likely to change for ReaNa in the near future and is worth reinvestigating later in the project.

Toil supports both K8s backend and CWL 1.2, but currently has a dependency on AWS API for storage, which cannot be replaced with self-hosted S3-like storage, even if K8s cluster is installed on a private cloud. The investigation of Toil ended when we exceeded the free tier AWS use.

Finally, Arvados promised both CWL1.2 and non-production support for Kubernetes, which we decided to test. Unfortunately, we failed to even install the platform in this mode - trying both private cloud and GCP. All errors were reported back to Arvados support, with some quickly solved and some still pending.

Of particular promise to further investigate is the newcomer StreamFlow, which promises support for CWL 1.2 with hybrid HPC/Cloud execution and Kubernetes backend [Colonnelli 2021].

CWL execution on HPC

As noted above we have investigated the practicalities of running our CWL workflows on a range of HPC infrastructure, using the Toil workflow engine.

Once Toil was set up for the particular HPC system, it did facilitate the running of workflows well – making moving a workflow between HPC systems very straightforward. Toil/CWL can make use of Singularity for running containerised

tools on HPC systems (even converting Docker containers to Singularity containers). This made the process of moving between HPC systems even easier.

The one limitation we found with using the BioBB tool containers directly from Toil/CWL was that we would be locked into using the GROMACS binary which is bundled with the tools in that container. To get around this restriction we can instead natively load the BioBB tools from the Conda packages (by using CWL *overrides*, or by stopping Toil from using any containerised tools). This enables us to use the configuration options provided by the BioBB tools for specifying the path to the desired GROMACS *gmx* binary (including specifying a Docker or Singularity container for that tool).

Another limitation we found in using Toil for workflow management on HPC systems was the (relatively) large fraction of the overall simulation time which was made up of tasks waiting in the job engine queues. Each CWL task will, as default, be submitted as a single job to the system job engine. If wait times for jobs are long compared to the compute time for that job then this causes a significant increase in simulation time.

The Toil workflow engine does carry out some analysis of the CWL task tree, in order to try to determine which tasks can be chained within a single system job. However the logic used for this analysis is very basic and conservative, resulting in very few job chains actually being formed. The Toil developers are intending to improve the logic for this analysis, and we have provided them with some examples of CWL workflows where job chaining should be possible, to use as tests for their development work. But, for now, simulation times are significantly longer for comparable workflows when running using Toil/CWL instead of PyCOMPSs.

CWL on HPC whitepaper

Combined with our experiences from BioExcel Binder (section 4.3), we are now in the process of writing up our findings and recommendations for executing scalable CWL workflows on HPC, Kubernetes and cloud systems as a BioExcel whitepaper.

We will publish this as part of the BioExcel Best Practice guides [BioExcel-2 D3.5] and further evolve the document with the wider CWL community. Here we will also suggest future work directions for the different engines.

1.4 Bioschemas and Schema.org

Bioschemas is a community effort to encourage semantic markup of life science resources on the Web, to improve their findability and indexing, as well as federated cataloguing and metadata integration across repositories. The markup

is embedded within the HTML of the resource pages (Figure 1), from where it can be programmatically extracted and integrated into a knowledge graph.

Building on the widely used and extensive Schema.org vocabulary as supported by search engines, Bioschemas profiles like ChemicalSubstance and

```
<script type="application/ld+json">
{
    "@context": "http://schema.org",
    "@type": "SoftwareApplication",
   "description": "biobb_analysis is the Biobb module collection to perform analysis
of molecular dynamics simulations. Biobb (BioExcel building blocks) packages are Python
building blocks that create new layer of compatibility and interoperability over popular
bioinformatics tools.",
    "name": "BioBB Analysis",
    "url": "https://github.com/bioexcel/biobb_analysis",
    "additionalType": "Library",
    "applicationCategory": "Computational Biology tool",
    "applicationSubCategory": "http://www.edamontology.org/topic_3892",
    "citation": "https://www.nature.com/articles/s41597-019-0177-4",
    "license": "https://www.apache.org/licenses/LICENSE-2.0",
    "softwareVersion": "3.7.0",
    "applicationSuite": "BioBB BioExcel Building Blocks",
    "codeRepository": "https://github.com/bioexcel/biobb_analysis",
    "isAccessibleForFree": "True",
    "image":
"http://mmb.irbbarcelona.org/biobb/public/assets/layouts/layout3/img/logo.png",
    "operatingSystem": ["Linux", "MacOS"],
    "offers": {
      "@type": "Offer",
      "price": "0",
      "priceCurrency":"EUR"
    }
  }
</script>
```

***Figure 1**: Bioschemas JSON-LD semantic markup for biobb_analysis, embedded in HTML. Keys are defined at https://schema.org/SoftwareApplication and selected based on the ComputationalTool profile.*

MolecularEntity specify relevant types and properties recommended for consistent markup. Despite its "bio" name, several profiles are general purpose across sciences, such as Dataset, ComputationalTool and TrainingMaterial.

The FAIR Guiding Principles have an aim to improve the Findability, Accessibility, Interoperability and Reusability of digital research objects for both humans and machines. While FAIR was conceived for research data, it has recently been argued that also Research Software artefacts themselves should be treated as FAIR objects [Katz 2021]. This has influenced the way we have designed and published BioExcel Building Blocks.

For the BioExcel Building Blocks to be a **Findable** software library, it must be easy to discover from many different sources, thanks to its presence in software registries, a rich metadata description, and in the central landing website and every library module using Bioschemas.

Bioschemas in the BioExcel Building Blocks libraries

Each BioExcel Building Blocks package has its own Bioschema markup, as shown in Figure 1 from biobb_analysis.

These Bioschemas are automatically gathered from the GitHub repositories and combined in the generated Modules section of the BioExcel Building Blocks landing website, joined with the global Bioschema description of the overall BioBB family and release. Figure 2 shows the search engine's view of the combined markup.

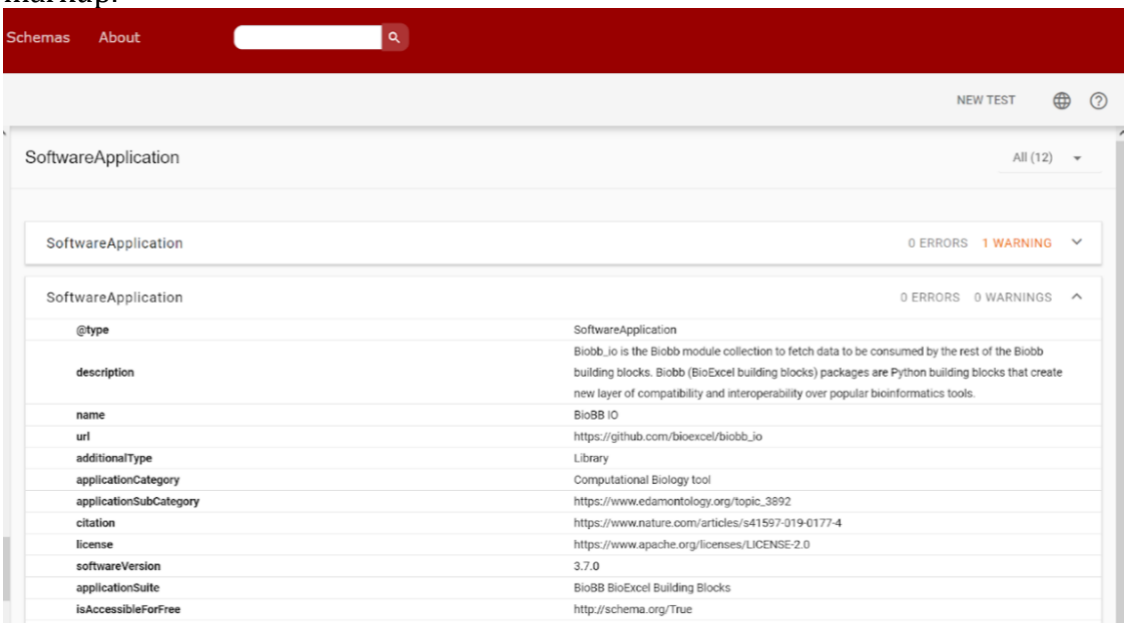


Figure 2: *schema.org validator showing structure of Bioschema markup of Biobb modules following the ComputationalTool profile. Note that the warning is due to the extra property of* `codeRepository`*, recommended as extension by the Bioschemas profile.*

To assist search engines in finding the corresponding documentation, all these Bioschema snippets have additionally been inserted into the corresponding BioExcel Building Blocks official documentation on Read The Docs.

1.5 Research Objects using RO-Crate

RO-Crate is a community-developed specification for packaging and describing Research Objects with FAIR metadata [Soiland-Reyes 2022]. UNIMAN is co-leading the RO-Crate community and have had a key role in the its establishment and maturing the specifications. Similar to Bioschemas (section 1.4), RO-Crate is based on Schema.org, but can be used independent of Web pages to become part of any data deposits. A comprehensive list of RO-Crate tools and libraries have evolved.

In BioExcel we have mainly used RO-Crate as part of WorkflowHub registration (section 1.6), but also for exposing BioBB canonical workflow building blocks as FAIR Digital Objects (section 4.4).

Further development work is underway, in collaboration with EOSC-Life, Synthesys+ and GA4GH, to expand the workflow engine toil for CWL (section 1.1) and Galaxy (section 2.2) to export workflow run provenance as RO-Crate according to an emerging Workflow Run profile, building on and generalizing our earlier work on CWLProv [Khan 2019].

RO-Crate will be used by the newly launched BY-COVID project for FAIR deposits in the COVID-19 Data Platform as well as for rich workflow provenance as detailed above.

1.6 WorkflowHub and EOSC-Life

As part of the EOSC Life project, the WorkflowHub registry has been developed to enable the describing, sharing and publishing of scientific computational workflows.

The BioExcel programme has been registered on this service, and a BioBB Building Blocks project space, associated with this programme, has been created. This has enabled us to gather all the BioBB related workflows into one registry area, as well as create collections of related workflows within the project, such as all tutorials for BioExcel Building Blocks, or to gather together the instances in different workflow languages of a single *canonical workflow* (in this case, the Protein MD Setup tutorial workflow, for jupyter notebook, CWL, Galaxy, PyCOMPSs, and KNIME.

For each registered workflow, a Research Object Crate (RO-Crate) is created, and a citable DOI can be minted for the workflow too, using DataCite. This allows for long term preservation of the registered workflows and their metadata. The WorkflowHub has an end-of-line preservation plan where their RO-Crates will be archived in Zenodo, and the DOIs redirected. The WorkflowHub also supports

versioning of workflows, so the records can be updated as the workflow is further developed.

Currently workflows have to be uploaded by directly providing the workflow file, or by providing an RO-Crate containing the workflow and associated metadata. Work within EOSC-Life is on-going to provide direct *git* support, to make registration of workflows from source code repositories such as at https://github.com/bioexcel/ more integral part of the workflow development (and, in particular, version control) process. In particular, BioExcel have been championing for such support in WorkflowHub to better handle multiple "flavours" of the same workflow, as we found necessary in the pmxlaunchCV19 workflow for COVID-19 research (see deliverable [BioExcel D6.3]).

2 Graphical pipeline interfaces for BioExcel workflows

2.1 BioBB in KNIME

A prototype plugin was developed to provide BioBB bindings for the workflow system KNIME. An example of using this plugin can be found in the workflow https://doi.org/10.48546/workflowhub.workflow.201.1 published in WorkflowHub, however the plugin itself was not matured beyond prototype status and is not publicly available.

Due to the nature of KNIME Nodes being developed as Java plugins, only a small subset of building blocks (sufficient for the above workflow) was added. Extending this work for the whole of BioBB (currently 155 nodes) would require significant manual coding or a careful Java code generation.

In addition we had challenges with KNIME's restrictive tabular data format, not always seeming compatible with biomolecular simulation step data, necessitating workarounds like a table cell containing a ZIP file of GROMACS files. This can become unnatural to KNIME users, who would normally expect to be able to inspect intermediate data using KNIME's data visualizers. Adding visualizers for molecular simulation to KNIME would alleviate some of these impedance mismatches in data structures between KNIME and BioBB, but again this would be a significant development job that BioExcel alone is not sufficiently resourced to prioritize.

We found it more natural to focus our effort on integration with workflow systems that are more file-oriented, like CWL, Galaxy and Jupyter Notebook.

2.2 BioBB in Galaxy

Galaxy is a data-intensive computational platform and workflow management system, used extensively in bioinformatics and other life sciences, popular partly due to its graphical interface and large collection of available tools. Galaxy can be installed on a local server or on the cloud, with multiple scalable compute backends. Several public Galaxy instances like ELIXIR-supported usegalaxy.eu provides free and easy access for new and light users before electing to set up their own instance.

As BioExcel BioBB tools have been described using CWL, and Galaxy has partial support for executing CWL tools, we have attempted to utilize this combination.

However issues have arisen, such as being advised by Galaxy developers to install the engine from a feature branch in git, rather than following the regular, more mature Galaxy install mechanisms. This adds to the fact that CWL support in Galaxy is still largely undocumented (but see the BioExcel Best Practice Guide on CWL engines).

We therefore concluded that convincing large public Galaxy servers like https://usegalaxy.eu/ to adapt BioBB through the CWL route would not at the moment be a sustainable approach.

Concurrently we tried a manual curation of a small subset of BioBB building blocks to add to Galaxy using its regular tool definition mechanism, which was successfully tested on the BSC's test instance https://dev.usegalaxy.es/

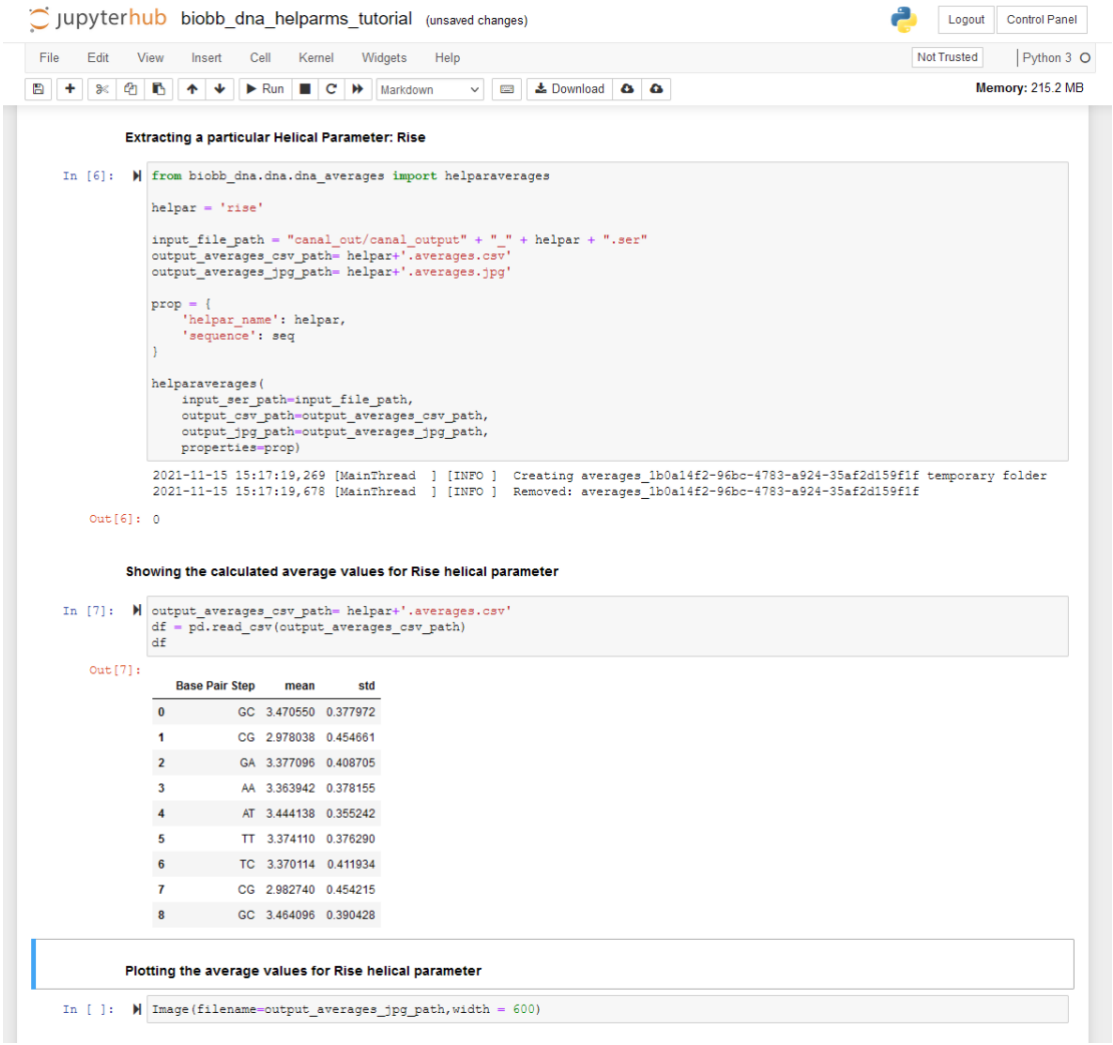
Subsequently, we were able to auto-generate the Galaxy tool definitions from BioBB's source code, which can make the whole of the BioBB family available, relying on their BioConda/Biocontainer images for distributing the blocks and their dependencies.

Next steps are that, following sufficient testing, we will be publishing these BioBB definitions to the official Galaxy Toolshed so that the BioExcel building blocks become available to users of large public Galaxy instances, as well as making BioBB easier to integrate in private Galaxy instances.

2.3 Jupyter Notebook and myBinder

A growing collection of BioBB workflows have been developed and made available for multiple languages, including Jupyter Notebook.

Jupyter is a Web-based platform for exploring data and executing code in multiple languages, including Python. One small challenge of getting started with a BioBB-



based Jupyter Notebook is that the user will first need to install Jupyter Notebook and the corresponding dependencies, including the BioBB wrappers and the individual tools they invoke, such as GROMACS.

myBinder allows instantiating a Notebook from Git-based repository, where its dependencies are automatically installed into a temporary Kubernetes container. The public instance of myBinder is able to use the BioBB workflows, which allows potential BioExcel and BioBB users to experiment with the workflows step by step.

However, for training purposes we found some challenges in scalability and concurrent users in using the public myBinder instance (particularly for simulation-heavy parts of the notebook), and therefore, as the corresponding BinderHub software is open source, we set up our own instances currently hosted by the EMBL-EBI Embassy cloud.

Figure 3: *BioBB workflow [Bayarri 2021] instantiated in the BioExcel Binder dedicated instance*

We integrated BinderHub installation with the BioExcel Cloud Portal, so that we can give dedicated access to training participants, maximise computational resources per user and provide persistent storage.

The BioBB entries in the WorkflowHub registry now include "*Launch in Binder*" links that can instantiate the notebook for almost immediate execution in our BinderHub instance available publicly as http://bioexcel-binder.tsi.ebi.ac.uk/, as shown in Figure 3.

Our experience with operating BinderHub is overall positive, however it exposed the fragility of both the Conda build system used by Binder and Kubernetes ecosystem.

We expected to obtain reproducible build results and times for all the notebooks we successfully tested with our installation, however, without making any changes to the Binder config and Biobb notebooks, we started to experience Conda dependency resolution performance problems (resolution running for hours or never finishing), reported widely by other Conda users and acknowledged by Conda authors.

This motivated the Binder team to replace Conda with Mamba and forced us to perform what turned out to be a chain of upgrades, complicated by planned deprecation of Docker runtime by Kubernetes and a Kubernetes bug for our cloud provider. Some issues with dependency resolution persist, as this time 2 different versions of Binder – our and myBinder – give different build results, so the investigation and potentially further upgrades will continue.

BioBB REST API

Another component of the BioBB library that is extending and easing its accessibility for our final users is the BioBB REST API. The BioBB REST API offers the possibility to remotely and programmatically access the broad collection of wrappers on top of common biomolecular simulation tools included in the BioBB library. This remote access facilitates the use of the available wrappers (building blocks), avoiding any type of installation process, convenient in the cases where local installation is not feasible.

Extended documentation complements the BioBB REST API: All the information related to the BioBB REST API endpoints includes:

- OpenAPI (Swagger) documentation with generic sample endpoints
- Specific examples of tools endpoints
- Tools endpoints for testing
- A tutorial details how to use the BioBB REST API endpoints

- Read the Docs documentation.

The BioBB REST API endpoints can be also used to run complete workflows. For example, see the tutorial to run the BioBB protein MD setup demonstration workflow using the BioBB REST API.

2.4 BioBB Workflows

Responding to the high interest shown by participants in the various training events organized by BioExcel, and following the work started by the IRB MDWeb server, a web-based GUI tool to help entry level users in biomolecular simulations has been developed: BioBB Workflows. The web server, completely based on our BioBB library, offers a collection of transversal, pre-configured BioBB workflows in a graphical, intuitive and interactive way.

The development started replicating the workflows included in the old MDWeb server (e.g. MD setup, trajectory quality control analyses). But thanks to the power of the BioBB library, more workflows were progressively added to the server, currently offering 12 different biomolecular simulation workflows.

The set of available workflows are based on the Jupyter Notebook demonstration workflows, covering different aspects of the biomolecular simulations field such as Molecular Dynamics, protein-ligand docking or DNA flexibility analysis.

The tool is designed to be easily extended with new BioBB workflows. Workflows are internally defined as pure Python scripts, with templates that are automatically modified with users input parameters. These input parameters are filled in by the user, helped not only by web-based forms but also by graphical tools such as 3D visualizers.

An example is a selection of the protein pocket where we are interested in performing a docking process with a small ligand. Here the website will automatically discover the different protein pockets (using fpocket BioBB building block) and will represent them using NGL, easing the selection of the corresponding pocket by the user (Figure 4).

Once the inputs are defined and the template is automatically modified, the workflow can be then either run in the server computational resources, or downloaded and run in the users' local premises.

For the first approach, the web server offers computational resources to run all the pipelines using an on-demand OpenNebula service, which automatically deploys new VMs when a new workflow is launched. The main advantage of this type of infrastructure is its scalability: new VMs can be deployed if the number of

concurrent users increases. This particularity is very convenient in this case, as we are planning to use the server in our BioExcel training events.

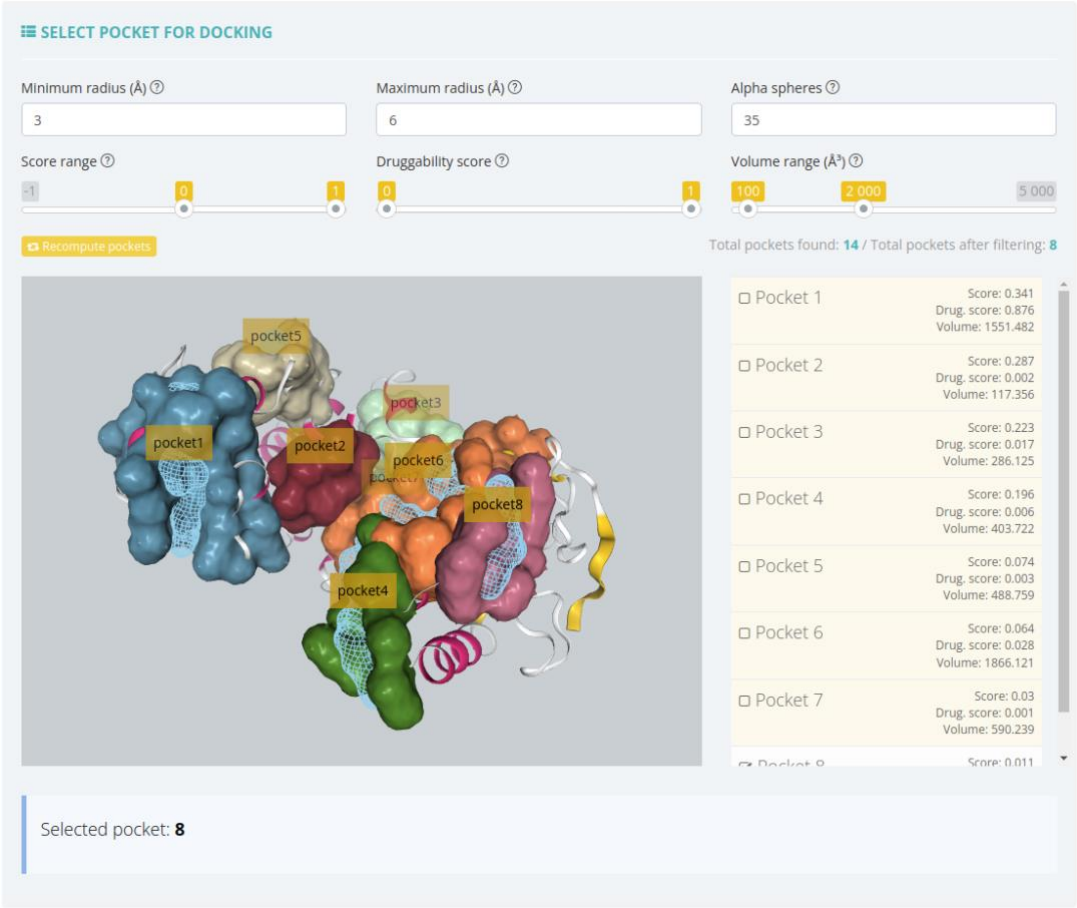


Figure 4: *BioBB_Workflows example helping the user in the selection of the particular protein pocket for a protein-ligand docking procedure*

For the second approach, the workflow is available to download in two different flavors: BioBB Python script and CWL. The Python script is based on BioConda packages, whereas the CWL workflow is based on Docker containers. In both cases, a README file with instructions on how to run the workflow is included.

Finally, BioBB_Workflows integrates the biobb_remote module, a package that allows remote execution of our BioBB building blocks through ssh connection. This module allows direct executions of building blocks in remote machines, and in particular, supercomputers. With this integrated feature, BioBB_Workflows allows to run, for example, a Molecular Dynamics setup in the server machines, and then launch the production MD in a supercomputer, all from the GUI, a functionality that was requested several times in our BioExcel-1 visits to pharmaceutical companies.

3 Community standards and activities

3.1 ELIXIR

ELIXIR is a European-wide research network that provides access to bioinformatics data resources and their computational processing. BioExcel has maintained an official collaboration with ELIXIR since 2017.

ELIXIR is actively involved in several community activities for interoperability, training and helping researchers to establish academic collaborations across organizations and infrastructures.

3.1.1 ELIXIR RDMkit

BioExcel contributed to the development of the **ELIXIR Research Data Management kit** (RMDkit). ELIXIR RDMkit is an online guide containing good data management practices applicable to research projects from the beginning to the end. RDMkit contains guidelines, information, and pointers to help in the research data life cycle. An important concept of RDMkit is its support on FAIR data from the first steps of data management planning to the final steps of depositing data in public archives.

RDMkit divides the research data management plans into different stages of the data management life cycle, and also gives clues depending on what is your role in the research data management process, or what is your **research domain**. In collaboration with the ELIXIR-CONVERGE project and helped by the ELIXIR Czech Republic node, a new Biomolecular Simulation data domain section was developed, reviewed and integrated into the RDMkit.

Information included in this new RDMkit domain is a brief introduction on **Biomolecular Simulations**, useful guidance on how to store and share data generated in this particular field, typical problems that one can encounter in this process, and relevant tools and resources associated with these tasks.

The information page is divided into questions and answers, including relevant questions in the field such as: **What** type of data do you have and what should you store? **Where** should you store this data? or **how** do you want your data to be shared? Possible solutions, including many existing databases and their references, were added, divided in different fields (e.g. Molecular Dynamics, Docking, Virtual Screening, etc.).

BioExcel collaboration with the RDMkit is still ongoing, and a review and update of the Biomolecular simulation data domain is planned for the next year.

In addition, the ELIXIR 3D-BioInfo structural community has also shown interest in combining effort, for the obvious overlapping with the structural domain, and the new releases will be put together also in collaboration with these ELIXIR community members.

3.1.2 ELIXIR 3D-BioInfo community

As introduced in the previous section, BioExcel is also in close collaboration with the new ELIXIR 3D-Bioinfo structural community. The collaboration has already solidified in the form of an ELIXIR Project Plan for Community-led Implementation Study called *Building on PDBe-KB to chart and characterize the conformational landscape of native proteins*.

The project, starting in June this year (2021) and with a period of 24 months, will explore the conformational diversity of the experimentally available proteins in the PDB database, augmented with results of state-of-the-art computational tools.

The set of benchmarking tools and workflows integrating some of these tools will be ELIXIR compliant, that means, will need to follow best practices on software findability, availability, and reproducibility/reusability. This is the area where BioExcel will help, through our expertise in FAIR workflows, following our work with the BioBB library.

3.2 ELIXIR Cloud and GA4GH standards

Global Alliance for Genomics and Health (GA4GH) is a collaborative effort to make standards for genomics data and processing [Rehm 2021]. In particular, the GA4GH Cloud workstream has developed API standards that BioExcel considered relevant to evaluate and potentially contribute to, namely Workflow Execution Service (WES), Task Execution Service API (TES), and Tool Registry Service (TRS), along with their corresponding open source reference implementations.

Our collaborative work with GA4GH is done largely together with the ELIXIR Cloud and AAI project.

Task Execution Service (TES)

TES provides a standard mechanism of defining computational tasks that can be executed across different compute environments such as HPC or cloud.

TESK was the earliest implementation of GA4GH Cloud WS API done as part of an Elixir implementation study that later morphed into Elixir Cloud and AAI GA4GH Driver Project. The origin of TESK lies in EMBL-EBI team's interest in the fairly new Kubernetes platform and an idea to use K8s Jobs as a basis of a Batch System

for private clouds such as OpenStack. Making the system compatible with GA4GH standards promised easy integration with existing workflow managers.

We quickly realised that the Kubernetes Batch API on its own is not enough to build a production ready system and in the light of new mature Kubernetes-based batch projects emerging, such as kube-batch, the project's ambitions were scaled down to be a playground for testing the TES standard development.

EMBL-EBI has co-lead the development of TES and guided it to the approval of the 1.0 version by the GA4GH board. This led to renewed interest in TES by both cloud providers and workflow manager developers.

Workflow Execution Service (WES)

cwl-WES is an implementation of a WES server for CWL workflows and a TES client and as such supports CWL execution over a standard API on potentially different computational backends. However, its current design as a thin wrapper around non-production cwl-tes causes scalability issues and its use is currently limited to demonstrations.

Tool Registry Service (TRS)

The TRS API allows programmatic discovery of computational tools to access their software packaging in containers, along with workflows using such tools for CWL, WDL, Nextflow, Galaxy and Snakemake.

UNIMAN have added support for TRS to the workflow registry WorkflowHub, which has enabled programmatic execution of workflows from multiple platforms, including the ELIXIR service UseGalaxy.eu and BSC's workflow execution service WfExS.

In the ELIXIR Biohackathon 2021, the WES implementation Sapporo demonstrated using TRS API to load and execute CWL and Nextflow workflows from WorkflowHub. We continue to contribute feedback to the TRS standard and its implementation following these practical experiences.

4 Workflow communities

4.1 MolSSI collaboration

The *Molecular Sciences Software Institute* (MolSSI) is US-based community-building activity spanning across computational molecular scientists and an official BioExcel partner.

Following the successful joint MolSSI and BioExel workshop [Naden 2019] arranged at BSC in 2018 [BioExcel-2 D2.1], planning and arrangements started for a follow-up workshop to be held in Arlington, Virginia, US in 2020, inviting tool developers and domain experts to pair up in a collaborative session to reduce the knowledge gap between biomolecular simulation scientists and workflow system developers. This meeting was unfortunately subsequently postponed and then cancelled, due to the COVID-19 pandemic and its effects on international travel, however several contacts were established with leading experts.

The focus of the collaboration with MolSSI changed focus to establish the COVID-19 Molecular Structure and Therapeutics Hub [BioExcel-2 D6.3] and UNIMAN provided letter of support for MolSSI's renewed NSF funding (which was announced August 2021); the aim there is to continue the collaboration with the aspects of FAIR Sharing of biomolecular simulation data and software.

The COVID-19 Molecular Structure and Therapeutics Hub is a community-driven data repository and curation service for molecular structures, models, therapeutics, and simulations related to COVID-19 computational research. The repository was designed from scratch to share data from the scientific community, making science and data completely open to better tackle the COVID-19 global emergency.

The Hub has become a reference repository sharing useful information such as MD simulations. Renowned groups in the field (e.g. D.E. Shaw, Riken, Folding@home)

have contributed to the repository, summing up milliseconds of data. The Hub is

Figure 5: *Example of trajectory visualization MCV1900274 [Mori 2021] of BioExcel-CV19 integrated in the COVID-19 Molecular Structure and Therapeutics Hub.*

today an essential repository in the biomolecular simulations field.

The BioExcel-CV19 database and associated web server expands the power of the Hub, including interactive graphical representations of the trajectories and analyses performed on them, with a main objective of being a tool for scientists interested in the COVID-19 research to interactively and graphically check key structural and flexibility features stemming from MDs. Both web servers were extensively presented in the deliverable D2.4 – Development of a framework for the combination of HPC and HPDA operations [BioExcel-2 D2.4].

The collaboration between BioExcel and MolSSI is currently active on this topic, and we are now working on the integration of the BioExcel-CV19 trajectory visualization section into the COVID-19 Molecular Structure and Therapeutics Hub (**Figure 5**).

4.2 Workflows-RI and Workflows Community Initiative

BioExcel participated and led sessions in two Workflow Community Summits hosted by the US-led WorkflowsRI and ExaWorks projects: *Bringing the Scientific Workflows Community Together* and *Advancing the State-of-the-art of Scientific Workflows Management Systems Research and Development*. These meetings brought together workflow engine developers, HPC infrastructure providers, and expert workflow users in order to build a roadmap for future Scientific Workflow research and development.

The future directions identified are *FAIR computational workflows [Goble 2020]; AI workflows; exascale challenges; APIs, interoperability, reuse, and standards; training and education*; and *building a workflows community* [Ferreira de Silva 2021]. These aims align well with the goals of BioExcel and EOSC for workflows and tools.

This recently formed community has agreed to continue collaboration beyond the WorkflowsRI/ExaWorks projects, with a newly initiated *Workflows Community Initiative (WCI). WCI* includes UNIMAN & BSC as founding partners together with several universities, HPC centers and NVIDIA. WCI is pulling together key players from the world of workflow engines and HPC, positioning itself to be strategically important for the future of FAIR and Exascale workflows.Carole Goble (UNIMAN) has agreed to join WCI as Director of FAIR Computational Workflows and Rosa M. Badia (BSC) as Director of Systems and Applications Performance. We are investigating forming a strategic partnership with BioExcel CoE after WCI gets established.

4.3 EOSC-Life community

As mentioned in section 2.7.1, BioExcel has been collaborating with the EOSC-Life project on using and improving their WorkflowHub repository. Regular meetings and events by the *WorkflowHub Club* community, EOSC-Life and ELIXIR, along with subsequent networking, have additionally shown to be beneficial for BioExcel's workflow development and use of standards as detailed in this deliverable.

For instance, when BioExcel was developing Bioconda packages as mentioned in section 2.1, or integrating BioBB into Galaxy (section 3.2), these established relations to the European life science workflow communities meant we could readily bring in expert assistance from the maintainers of the underlying infrastructures.

4.4 FAIR Digital Objects and Canonical Workflow Frameworks for Research

FAIR Digital Objects (FDO) [De Smedt 2020] have been proposed as a conceptual framework for making digital resources available in a Digital Objects (DO) architecture which encourages active use of the objects and their metadata. In particular, an FDO has five parts:

1. The FDO *content*, bit sequences stored in an accessible repository
2. a *Persistent Identifier* (PID) such as a DOI that identifies the FDO and can resolve these same parts
3. Associated rich *metadata*, as separate FDOs
4. Type definitions, also separate FDOs. A Digital Object typed as a *Collection* aggregates other DOs by reference.
5. Associated *operations* for the given types.

The concept Canonical Workflow Frameworks for Research (CWFR) has been defined by the FAIR Digital Objects forum, later establishing a CWFR working group where UNIMAN is an active member.

CWFR is envisioned as layered on top of existing workflow technologies that captures the scientific practices encoded by a computational method in workflow, distinct from its practical workflow implementation. CWFR identifies three elements [Hardisty 2020]:

1. Pattern of recurring canonical steps in research activities
2. Libraries of canonical steps
3. Packages per canonical step

UNIMAN (and through EOSC-LIFE BSC_, have presented WorkflowHub and RO-Crate as an FDO implementation to the CWFR forum. This approach will be detailed in a manuscript in preparation about FDO demonstrators [Wittenburg 2021].

In [Soiland-Reyes 2021], accepted for a *Data Intelligence* special issue for CWFR, BioExcel argue that the BioBB is an example of canonical workflow building blocks, as a library of canonical steps that can be used consistently in multiple workflow systems, assisted by common packaging in Conda and as containers (section 1.1). As mentioned in section 1.3, the use of canonical building blocks also generalize training across workflow systems.

In the WorkflowHub registration of BioBB HPC tutorial with PyCOMPSs we have extended the Workflow RO-Crate to detail the BioBB metadata as a FAIR Digital Object, shown in **Figure 6** and **Figure 7** below.

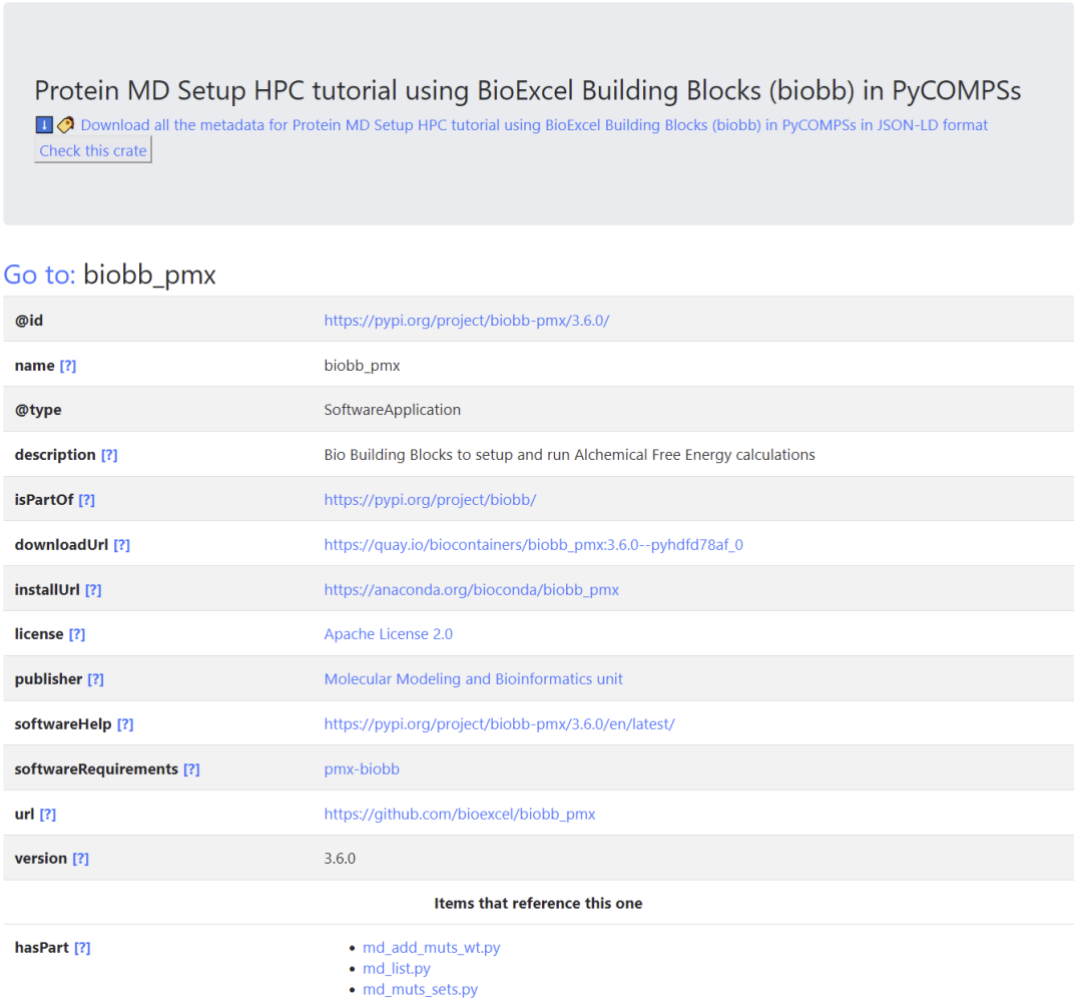
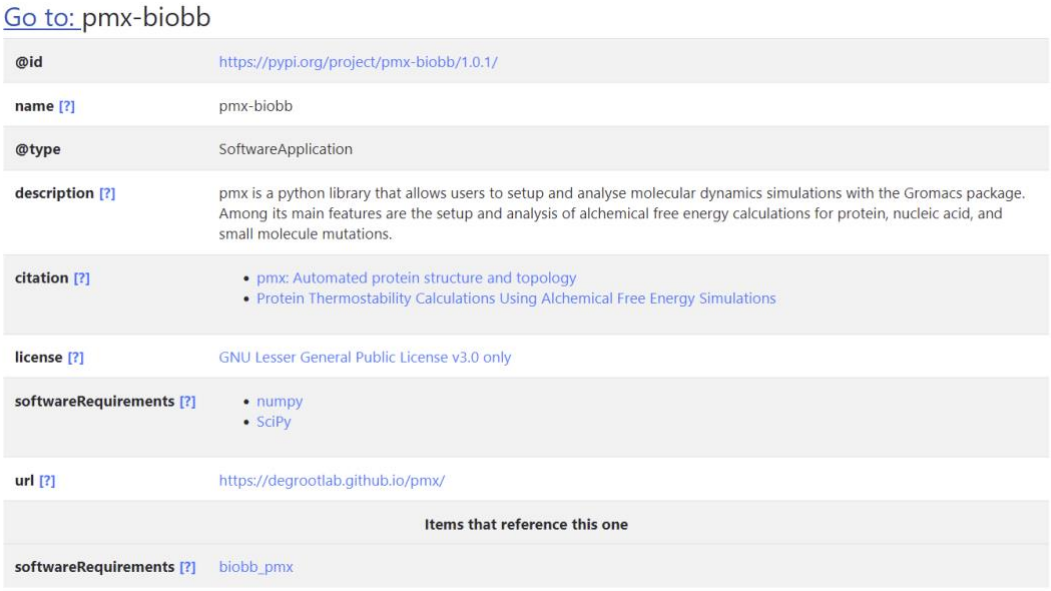


Figure 6: *From Workflow RO-Crate describing the BioBB building block biobb_pmx [Soiland-Reyes 2021b], includes license for the building block, links to BioContainer and Conda, documentation and a software requirement on the wrapped pmx-biobb.*



Go to: pmx-biobb

@id	https://pypi.org/project/pmx-biobb/1.0.1/
name [?]	pmx-biobb
@type	SoftwareApplication
description [?]	pmx is a python library that allows users to setup and analyse molecular dynamics simulations with the Gromacs package. Among its main features are the setup and analysis of alchemical free energy calculations for protein, nucleic acid, and small molecule mutations.
citation [?]	• pmx: Automated protein structure and topology • Protein Thermostability Calculations Using Alchemical Free Energy Simulations
license [?]	GNU Lesser General Public License v3.0 only
softwareRequirements [?]	• numpy • SciPy
url [?]	https://degrootlab.github.io/pmx/
Items that reference this one	
softwareRequirements [?]	biobb_pmx

Figure 7: *Contextual entity within RO-Crate [Soiland-Reyes 2021b], describing the pmx softwareRequirement from biobb_pmx. This highlights that the underlying tool wrapped by the building block has separate citations, license, and its own software requirements, here capture as FAIR metadata within the Workflow Crate.*

Further work for BioExcel with the canonical workflow concept would include making FAIR Digital Objects for individual building blocks and generating RO-Crate (section 1.5) from their Bioschemas annotations (section 1.4) so that the above CWFR descriptions can be created programmatically from any workflow run using BioBB.

5 References

[Abraham 2020] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Pálla, Jeremy C. Smith, Berk Hess, Erik Lindahl (2016): **GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers.** *SoftwareX* **1-2** https://doi.org/10.1016/j.softx.2015.06.001

[Amstutz 2020] Peter Amstutz, Michael R. Crusoe, Kaushik Ghose, John Chilton, Michael Franklin, Bogdan Gavrilovic, Stian Soiland-Reyes (2020): **Common Workflow Language (CWL) Workflow Description, v1.2.**
https://w3id.org/cwl/v1.2/

[Andrio 2019] Pau Andrio, Adam Hospital, Javier Conejero, Luis Jordá, Marc Del Pino, Laia Codo, Stian Soiland-Reyes, Carole Goble, Daniele Lezzi, Rosa M. Badia, Modesto Orozco, Josep Ll. Gelpi (2019): **BioExcel Building Blocks, a software library for interoperable biomolecular simulation workflows**. *Scientific Data* **6**:169
https://doi.org/10.1038/s41597-019-0177-4

[Andrio 2021] Pau Andrio, Adam Hospital, Cristian Ramon-Cortes, Javier Conejero, Daniele Lezzi, Jorge Ejarque, Josep LL. Gelpi, Rosa M. Badia (2021): **Enabling the execution of large scale workflows for molecular dynamics simulations**. *bioRxiv* https://doi.org/10.1101/2021.04.14.439795

[Bayarri 2021] Genís Bayarri, Adam Hospital (2021): **Structural DNA helical parameters from MD trajectory tutorial using BioExcel Building Blocks (biobb).** *WorkflowHub* https://doi.org/10.48546/workflowhub.workflow.195.1

[BioExcel-2 D2.1] Adam Hospital, Stian Soiland-Reyes, Josep Lluís Gelpí, Pau Andrio, Daniele Lezzi, Sarah Butcher, Ania Niewielska, Yvonne Westermaier (2019): **BioExcel-2 Deliverable 2.1 – State of the Art and Initial Roadmap**. *Zenodo* https://doi.org/10.5281/zenodo.4604607

[BioExcel-D D2.3] Adam Hospital, Genis Bayarri, Stian Soiland-Reyes, Jose Lluis Gelpi, Pau Andrio, Daniele Lezzi, Sarah Butcher, Ania Niewielska, Yvonne Westermaier, Rosa Maria Badia, Rodrigo Vargas, Alexandre Bonvin (2020): **BioExcel-2 Deliverable 2.3 – First release of demonstration workflows.** *Zenodo* https://doi.org/10.5281/zenodo.4540432

[BioExcel-2 D2.4] Adam Hospital, Genis Bayarri, Josep Lluis Gelpi, Pau Andrio, Daniele Lezzi, Rosa M Badia, Jorge Ejarque, Javier Alvarez, Salvi Sola, Sandro Barissi, Federica Battistini, Diego Gallego, Alba Sala, Yvonne Westermaier (2020): **BioExcel-2 Deliverable 2.4 – Development of a framework for the combination of HPC and HPDA operations**. *Zenodo* https://doi.org/10.5281/zenodo.4916029

[BioExcel-2 D3.5] Stian Soiland-Reyes, Arno Proeme, Alexandre Bonvin, Vytautas Gapsys (2020): **BioExcel-2 Deliverable D3.5 – Best Practice Guides**. *Zenodo* https://doi.org/10.5281/zenodo.5511704

[BioExcel-2 D6.3] Rossen Apostolov, Alexandre Bonvin, Dmitry Morozov, Stian Soiland-Reyes, et al (2021): **BioExcel-2 Deliverable 6.3 – High Performance Computing in support of COVID-19 Research.** (planned).

[Colonnelli 2021] Iacopo Colonnelli, Barbara Cantalupo, Ivan Merelli, and Marco Aldinucci (2021): **StreamFlow: Cross-Breeding Cloud With HPC.** *IEEE Transactions on Emerging Topics in Computing* **9**(4). https://doi.org/10.1109/tetc.2020.3019202

[Crusoe 2022] Michael R. Crusoe, Sanne Abeln, Alexandru Iosup, Peter Amstutz, John Chilton, Nebojša Tijanić, Hervé Ménager, Stian Soiland-Reyes, Bogdan Gavrilović, Carole Goble, The CWL Community (2021): **Methods Included: Standardizing Computational Reuse and Portability with the Common Workflow Language.** *Communication of the ACM* (accepted). *arXiv* **2105.07028** [cs.DC]

[De Smedt 2020] Koenraad De Smedt, Dimitris Koureas, Peter Wittenburg (2020): **FAIR digital objects for science: from data pieces to actionable knowledge units.** *Publications* **8**(2), 21. https://doi.org/10.3390/publications8020021

[Ferreira de Silva 2021] Rafael Ferreira da Silva, Henri Casanova, Kyle Chard, Ilkay Altintas, Rosa M Badia, Bartosz Balis, Tainã Coleman, Frederik Coppens, Frank Di Natale, Bjoern

Enders, Thomas Fahringer, Rosa Filgueira, Grigori Fursin, Daniel Garijo, Carole Goble, Dorran Howell, Shantenu Jha, Daniel S. Katz, Daniel Laney, Ulf Leser, Maciej Malawski, Kshitij Mehta, Loïc Pottier, Jonathan Ozik, J. Luc Peterson, Lavanya Ramakrishnan, Stian Soiland-Reyes, Douglas Thain, Matthew Wolf (2021): **A Community Roadmap for Scientific Workflows Research and Development.** *arXiv [cs.DC]* https://arxiv.org/abs/2110.02168

[Goble 2020] Carole Goble, Sarah Cohen-Boulakia, Stian Soiland-Reyes, Daniel Garijo, Yolanda Gil, Michael R. Crusoe, Kristian Peters, Daniel Schober (2020): **FAIR Computational Workflows**. *Data Intelligence* **2**(1):108–121 https://doi.org/10.1162/dint_a_00033

[Hardisty 2020] CWFR Group, Alex Hardisty (ed), Peter Wittenburg (ed) (2020): **Canonical Workflow Framework for Research CWFR**. Position Paper. *osf.io* https://osf.io/3rekv/

[Katz 2021] Daniel S Katz, Morane Gruenpeter, Tom Honeyman, Lorraine Hwang, Mark D Wilkinson, Vanessa Sochat, Hartwig Anzt, Carole Goble (2021):
A Fresh Look at FAIR for Research Software. *arXiv [cs.SE]*
https://arxiv.org/abs/2101.10883

[Khan 2019] Farah Zaib Khan, Stian Soiland-Reyes, Richard O. Sinnott, Andrew Lonie, Carole Goble, Michael R. Crusoe (2019):
Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv. *GigaScience* **8**(11):giz095
https://doi.org/10.1093/gigascience/giz095

[Long 2020] Robin Long (2020):
Creating hardware optimised Conda Packages.
https://longr.github.io/posts/gromacs-conda/

[Long 2021] Robin Long, Douglas Lowe, Stian Soiland-Reyes (2021):
Distributing Hardware-optimized Simulation Software with Conda. *Research IT*, The University of Manchester https://research-it.manchester.ac.uk/news/2021/05/25/distributing-hardware-optimized-simulation-software-conda/

[Mori 2021] Takaharu Mori, Jaewoon Jung, Chigusa Kobayashi, Hisham M Dokainish, Suyong Re, Yuji Sugita (2021): **Elucidation of interactions regulating conformational stability and dynamics of SARS-CoV-2 S-protein**. *Biophysical Journal* **120**(6)
https://doi.org/10.1016/j.bpj.2021.01.012

[Naden 2019] Levi N. Naden, Sam Ellis, Shantenu Jha (2019):
MolSSI and BioExcel Workflow Workshop 2018 Report. *arXiv* 1905.11863.
https://arxiv.org/abs/1905.11863

[Rehm 2021] Heidi L. Rehm, Angela J.H. Page, Lindsay Smith, et al. (2021):
GA4GH: International policies and standards for data sharing across genomic research and healthcare. *Cell Genomics* **1**(2)
https://doi.org/10.1016/j.xgen.2021.100029

[Soiland-Reyes 2021] Stian Soiland-Reyes, Genís Bayarri, Pau Andrio, Robin Long, Douglas Lowe, Ania Niewielska, Adam Hospital, Paul Groth (2021):

Making Canonical Workflow Building Blocks interoperable across workflow languages. *Data Intelligence* (accepted)
https://doi.org/10.5281/zenodo.5727730

[Soiland-Reyes 2021b] Soiland-Reyes, Stian, Pau Andrio, and Adam Hospital (2021): **Protein MD Setup HPC Tutorial Using BioExcel Building Blocks (Biobb) in PyCOMPSs.** *WorkflowHub* https://doi.org/10.48546/workflowhub.workflow.200.1

[Soiland-Reyes 2022] Stian Soiland-Reyes, Peter Sefton, Mercè Crosas, Leyla Jael Castro, Frederik Coppens, José M. Fernández, Daniel Garijo, Björn Grüning, Marco La Rosa, Simone Leo, Eoghan Ó Carragáin, Marc Portier, Ana Trisovic, RO-Crate Community, Paul Groth, Carole Goble (2022):
Packaging research artefacts with RO-Crate. *Data Science* (accepted)
https://arxiv.org/abs/2108.06503

[Turilli 2019] Matteo Turilli, Mark Santcroos, Shantenu Jha (2019): **A Comprehensive Perspective on Pilot-Job Systems**. *ACM Computing Surveys* 51(2):43
https://doi.org/10.1145/3177851

[Wittenburg 2021] Peter Wittenburg, Ivonne Anders, Christophe Blanchi, Merret Buurmann, Carole Goble, Jonas Grieb, Alex Hardisty, Sharif Islam, Thomas Jejkal, Tibor Kalman, Christine Kirkpatrick, Larry Lannom, Thomas Lauer, Giridhar Manepalli, Karsten Peters-von Gehlen, Andreas Pfeil, Robert Quick, Mark van de Sanden, Ulrich Schwardmann, Stian Soiland-Reyes, Rainer Stotzka, Zachary Troutt, Dieter van Uytvanck, Claus Weiland, Philipp Wieder (2021): **FAIR Digital Object Demonstrators 2021.** *In preparation.* Draft 2.8.1