Towards Reproducible Bioinformatics: The OpenBio-C Scientific Workflow Environemnt

Alexandros Kanterakis
Institute of Computer Science
Foundation for Research and Technology–Hellas (FORTH)
Heraklion, Crete, Greece
kantale@ics.forth.gr

Galateia Iatraki
Institute of Computer Science
Foundation for Research and Technology–Hellas (FORTH)
Heraklion, Crete, Greece
giatraki@ics.forth.gr

Konstantina Pityanou
Dept. of Electrical and Computer Engineering,
Hellenic Mediterranean University
Heraklion, Crete, Greece
kwna.pitianou@gmail.com

Lefteris Koumakis
Institute of Computer Science
Foundation for Research and Technology–Hellas (FORTH)
Heraklion, Crete, Greece
koumakis@ics.forth.gr

Nikos Kanakaris
Industrial Management and Information Systems Lab, MEAD,
University of Patras,
Patras, Greece
nkanakaris@upnet.gr

Nikos Karacapilidis
Industrial Management and Information Systems Lab, MEAD,
University of Patras
Patras, Greece
karacap@upatras.gr

George Potamias
Institute of Computer Science
Foundation for Research and Technology–Hellas (FORTH)
Heraklion, Crete, Greece
potamias@ics.forth.gr

***Abstract*—The highly competitive environment of post-genomics biomedical research is pushing towards the production of highly qualified results and their utilization in clinical practice. Due to this pressure, an important factor of scientific progress has been underestimated namely, the reproducibility of the results. To this end, it is critical to design and implement computational platforms that enable seamless and unhindered access to distributed bio-data, software and computational infrastructures. Furthermore, the need to support collaboration and form synergistic activities between the engaged researchers is also raised, as the mean to achieve consensus on respective scientific discoveries. The aforementioned needs and requirements underlie the motivations for the development of OpenBio-C environment, and determine its fundamental objectives. The insufficiencies of current workflow editing and execution environments are explored, and the key-directions to advance over an environment that supports reproducibility are stated. The basic components and functionality of the OpenBio-C initial implementation (beta version) are also presented.**

Keywords—*bioinformatics, scientific workflows, collaborative systems*

I. INTRODUCTION

With the global scientific output doubling every nine years, there is a question whether this increase has a clear impact to the "real" and actionable knowledge growth. The majority of publications are characterized as "background noise" and researchers are struggling to "separate the wheat from the chaff" [1]. Moreover, researchers still use generic Google-like searches to locate useful tools and analysis pipelines, and rely on specialized web forums to seek technical advices. It is not an exaggeration to claim that science on that matter has not changed over the last 20 years. This is surprising however, given that the results of most of the millions of publications in experimental sciences have been generated through an *analysis pipeline* of some kind. In fact, these pipelines are rarely publicly available. The temporal decay and eventual loss of these pipelines, in contrast to the almost ever-preserved and well-indexed scientific papers, constitutes a *major knowledge and cultural loss* not only for the scientific community but, for the society as a whole.

Reproducibility or, "*to publish or not to publish*". It is reported that published and highly-ranked biomedical research results are very often not *reproducible* [2]. In a relevant investigation, about 92% (11/12) of interviewed researchers could not reproduce results even if the methods presented in the original papers were exactly replicated [3]. In addition, there are estimates that preclinical research is dominated by more than 50% of irreproducible results at a cost of about 28 billion dollars [4]. We argue that this unnatural separation between scientific reports and analysis pipelines is one of the origins of the *reproducibility crisis* [5]. Taking the above into account, one may even state that "*reproduce or perish*" should be the new motto for contemporary science [6].

Data, data everywhere! The irreproducibility phenomenon is particularly evident in the field of genomics, largely due to the rapid evolution of biotechnology and the delivery of advanced high-throughput platforms. Evidently, the last ten years NCBI's resources like *Genbank* (the annotated collection of publicly available DNA sequences), *WGS* (the whole genome shotgun-based sequences) and *SRA* (the sequence read archive) has grown with unprecedented rates[1], going even beyond the predictions of Moore's law [7]. The daily production of post-genomic data worldwide outpaces even traditional "big data" domains, such as astronomy and Twitter, in terms of acquisition, storage, distribution, and analysis of respective datasets [8]. Even if advanced computing environments, and solutions for analyzing large amounts of data are available (e.g., distributed computing applications, bulk data storage technologies, cloud environments), their application to post-genomics data is still

[1] www.ncbi.nlm.nih.gov/genbank/statistics, www.ncbi.nlm.nih.gov/sra/docs/sragrowth

limited. This is due to difficulties in accessing relevant computational infrastructures, to their adaptation and customization complexity, as well as to the needed computer skills. In addition, factors related to security as well as to ethical and legal issues, constrain biomedical researchers to adopt these technologies [9].

Interpretable analytics. However, the notion of "big data" is expanded to refer not only to the data volume itself, but to the ability to analyze and interpret those data. Especially in the field of post-genomics the rate of data generation outperforms the respective figures for their analysis and effective interpretation, making the production of interpretable results largely unfulfilled [10]. In domains where reliability and transparency are of critical importance, like in *precision medicine* studies, the need is even more intense [11], [12]. As a consequence, the involved researchers become "isolated", being unable to support their findings as well as the primary research questions themselves. In other words, we are experiencing a research landscape, in which the participating researchers often do not know (*how*) to analyze and fail to make sense (*what*) of the scientific findings and discoveries; a situation that constraints and limits the exploitation of relevant scientific results. This makes it harder to *translate* scientific discoveries into credible clinical protocols and recommendations. Evidently, from about 150,000 scientific publications reporting on the discovery of new biomarkers, only 100 of them have found their way to the clinic! [13]. Without exaggeration, one would say that researchers of interdisciplinary fields, such as that of post-genomics biomedical research, experience a "*lost in translation*" situation! [14].

The above pose a major challenge to the contemporary bioinformatics community; namely, to establish an *extrovert research ecosystem that enables open-science activities and ease the seamless exploitation of data and tools* (ec.europa.eu/research/openscience). This is exactly the motivation and the vision underlying the OpenBio-C bioinformatics platform that is presented in this paper. Our aim is to develop and deliver an open solution that (i) offers services to incorporate workflows from different scripting or compiled languages; (ii) it is equipped with graph-based workflow orchestration capabilities; (iii) offers collaborative services to support decision-making in workflow design and sharing; (iv) enables semantic annotation, flexible management and reuse of resources based on the utilization and appropriate adaptation of the state-of-the-art *Research Objects* technology (www.research object.org), as the means for the encapsulation of research data and associated analytical methodologies; and (v) supports seamless workflow execution and monitoring in high-performing computational installations.

II. Bioinformatics Environments and Collaboration Support Tools: Current State-Of-the-Art

A. Bioinformatics Workflow Environments

Extensive reviews of bioinformatics *scientific workflow management systems* (SWfMS) already exist in the literature [15]–[18]. In this section, we briefly report on the most well-known and widely utilized ones, aiming to identify gaps and problematic issues towards the development of an integrated, open and flexible workflow environment.

Galaxy. It is probably the most successful SWfMS environment [19]. Galaxy is supported by a strong bioinformatics community with a fairly large number of distributed server installations in many research institutions worldwide. In spite of its success, Galaxy is still lacking in several capabilities such as the ability of users to collaborate during the composition and sharing of workflows, as well as the ability to evaluate and score the delivered workflows by exchanging ideas, suggestions and alternative approaches.

Taverna. It is the second most utilized SWfMS by the bioinformatics community [20]. Taverna has, at least in relation to Galaxy, limited use in post-genomics research. The reason is that it is based on an offline and relatively more complex workflow synthesis framework.

Molgenis. It offers services for embedding tools and simpler workflows in a more flexible ways than Galaxy and Taverna [21]. Molgenis has incorporated many well-known and well-established tools for the retrieval of genomic resources and data, focusing on phenotype-to-genotype association analysis and next generation sequencing data analysis, while offering workflow execution in distributed computing environments [22].

Programmatic packages. In addition to integrated *scientific workflow* (SWf) synthesis and execution environments we should also refer to solutions that are based on programming languages and provide the ability to synthesize workflows through relevant programmatic scripts. Such solutions are provided by relatively Python (such as *bcbio-nextgen* [23]) and Java packages (such as *bpipe* [24]). Other quite flexible solutions include *Snakemake* [25], *Nextflow* [26] and *BigDataScript* [27], which provide new domain-specific programming languages (DSL) to synthesize workflows and bioinformatics pipelines. Computational languages for the description and sharing of workflows, including *CWL* (common workflow language, software.broad institute.org/wdl) and WDL (open workflow description language, www.commonwl.org) are also standard solutions; followed by environments dedicated to specific tasks like, *Omics Pipe* [28], *EDGE* [29] that focus on next generation sequencing analysis, and *Chipster* [30] dedicated to microarray data analysis.

B. Collaboration Support Tools

Collaborative support systems (CSS) relate to software designed to ease a group of people to elaborate on their common tasks and achieve their goals [31]. Advances over the Web 2.0 technology introduced a multitude of CSS tools that ensure user interaction on a massive scale. These tools address a wide range of needs including, sharing of knowledge, mental mapping, and argumentation. The range of CSS could be grouped into two main categories namely, *mind mapping* and *argumentation support* tools.

Mind Mapping. These tools allow the creation and processing of the so-called "*mind maps*". A mental map is a diagram formed to represent ideas, or other elements connected and arranged around a central topic to explore. Mental maps are mainly used to visualize and appropriately organize ideas offering problem-solving functionality to support decision-making. A mental map may be considered as an illustration of accumulated knowledge in the field. Around the central concept, usually in a circular layout and with lines connecting them to the central idea, other ideas and concepts can be added. Mind mapping tools are primarily designed to support brainstorming, mainly in environments with increased data volumes and multiple users. They provide appropriate notifications in order for a user to be aware of the changes

made by other users to mental maps the creation of which the user has contributed. They also support detailed follow-up of the history of each mental map thus enabling the retrieval of any version of the mental map from its creation to its last version. Representative mind mapping tools include: *MindMeister*. (www.mindmeister.com), *Mindomo* (www.mindomo.com), *Bubbl.us* (www.bubbl.us), and *Xmind* (www.xmind.net).

Argumentation support. Argumentation is a verbal activity, which is usually conducted in plain language. A person involved in argumentation uses words and phrases to declare, ask or deny something, respond to someone else's statements, questions or denials, and so on [32]. Argumentation is also a social activity which, in principle, is addressed to other people and is directly related to the conclusion through some reasoning mechanism. It always refers to a specific view of a particular subject and the need for arguments arises when there is a divergence of views on this issue. Technologies supporting argumentative collaboration usually provide the means to structure and visualize discussions, exchange documents, and manage users. In addition, they aim to use the argument as a means to create a common basis among the engaged stakeholders to comprehend specific positions and arguments on a topic, to establish assumptions and criteria, and to build a collective consensus on it. Representative argumentation support tools include: *Araucaria (*araucaria.computing.dundee.ac.uk/doku.php*)*, *DebateGraph* (debategraph.org), *Compendium (*compendium.open.ac.uk*)*, *CoPe_it!* (copeit.cti.gr), *Cohere (*cohere.open.ac.uk*)*, and *Dicode* (dicode-project.cti.gr)

C. *Insufficiencies of current workflow technology*

As we have mentioned already, Taverna and Galaxy are the most known bioinformatics SWf environments, while *myExperiment* is the broadly used repository for the Taverna workflows (www.myexperiment.org) [33]. Even if these environments are widely utilized by the worldwide bioinformatics community, the need for a universal environment that promotes open and reproducible science is still prominent. Moreover, their popularity seems to decline, and other more flexible and open solutions, like Bioconductor (www.bioconductor.org) and Python-based workflows, start to take over. As a prove for this, we queried Google Scholar for articles related to the aforementioned environments, for a period of eleven years (2008 to 2018). Each query included the terms "bioinformatics" and "workflow" and "<environment>", with <environment> taking the values, "taverna", "galaxy", "myexperiment", "bioconductor" and "python", in each respective query. The hits of an environment for each year was normalized relatively to the sum of hits across all environments, getting a measure of *popularity trend* for each environment. The results are illustrated in Fig. 1. From Fig. 1, one could easily observe that the relative popularity of both Taverna and myExperiment constantly drops; for Taverna already from 2008 (denoted with the shaded square mark) to reach a relative popularity figure of 4% in 2018; and for myExperiment from 2009-2010 (where it enjoyed its maximum popularity) to reach a popularity figure of just 2% in 2018. Even Galaxy, the most recent and widely supported bioinformatics workflow environment, gained its maximum popularity around 2015 (~19%), and it then started to decline, reaching a relative popularity figure of 13% in 2018. Opposite trends are observed for Bioiconductor that started with a relative medium popularity in 2008 (~16%) and progressively reached a figure of 30%.

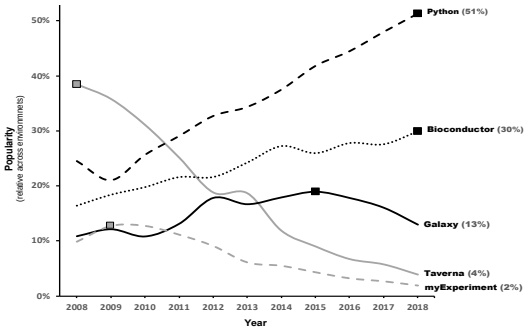

Fig. 1. Popularity trends of established and widely utilized Bioinformatics workflow environments based on Google Scholar hits..

The most noticeable observation concerns the popularity trend of Python-based workflows; it is constantly and aggressively increasing, reaching a relative popularity figure of 51% in nowadays. This phenomenon could be attributed to two main reasons: (i) the *closed nature* of environments, like Taverna and Galaxy, and the subsequent "inflexibility" to embed other scientific workflows (or other native computational objects); this is also the case for Bioconductor which, even providing an open environment with a big number of community contributed analytical packages, the revision of these packages in order to meet specific data analysis needs is difficult (if not impossible), and this justifies its relatively constant but "conservative" usage across the years; and (ii) the *limited collaboration and community-based co-creation* services offered by most of these workflow environments. Both the above reasons constrain the acceptance and widespread exploitation of current bioinformatics SWf environments. Of course, when these systems were developed (before 2010) the reproducibility crisis and big data challenges had not been emerged as eminent problems. Going deeper we could identify a number of intrinsic insufficiencies to the existing workflow environments: (a) they demand and enforce users to learn new and even domain-specific description languages, and this constitutes a burden for newcomers since it requires large amounts of time to acquire the needed expertise to build even simple solutions; (b) they are not *inclusive*, suffering from the '*lock-in*' syndrome and the inability to include and integrate diverse and heterogeneous workflows; (c) they demonstrate external dependencies, and do not support versioning and packaging of third party applications, components and services, thus hindering end-users to reproduce the computational environment necessary to achieve the same experimental results; and (d) they offer limited support for the execution of workflows in High Performance Environments (HPC) –whenever this is supported, it requires extensive technical knowledge and involves a lot of manual configuration and 'tweaking' of the environment. So, researchers tend to explore alternative (more open) solutions for their analytical needs, like Python, and even move to lower-level alternatives (e.g., scripting languages) in order to design their computational scientific workflows and pipelines from scratch. As a consequence, new collaborative communities are formed, e.g., *biopython* (biopython.org) [34], a tendency that even if it shows the vivid nature of collaborative research, it leaves unaffected the current fragmented research landscape. Trapped in such a problematic research landscape, and guided by an important statement that "*one-size-fits-all is unlikely to work"*, the challenge for a new conceptualization of a workflow environment is raised. Such a conceptualization should be originated on four foundational specifications:

- a workflow is not to be considered as stand-alone digital object but as *community building infrastructure* that work best when interlinked with other objects and activities around it,
- a workflow, and its constituents, should comply to the so-called FAIR (Findable, Accessible, Interoperable and Reusable) principles [35],
- a workflow should fully expose its provenance and context-of-use elements, i.e., "*for what and where this comes from*", and
- reproducibility, provenance and context information should be *intrinsic* attributes of the workflow (e.g., analysis code and its versioning, platform to run it on, verifiable versions of the data and methods, usage ranks) and not just standardized meta-data descriptions.

III. Facing the Challenge

A. The Need for Open-Science Methodologies

Publishing the source code and making accessible the data of a research study allows the community not only to verify the reliability of the followed methodology but also, to use and adapt it to new research projects. Every new invention or tool evolves exactly through this process, i.e., *create* => *test* => *use* the tool, and it is strange why Bioinformatics did not embrace it, at-least until the ΄90s when initiatives such as the *Open Bioinformatics Foundation* (OBF, http://www.open-bio.org/wiki/BOSC) was formed, and the *BOSC* annual conferences began [36]. One of the OBF suggestions states that any scientific publication should incorporate and make available the whole research methodology [37]. This raises the need for the development of tools and services that enable the reproduction and verification of published results, a pretty complicated task, just because the big number of available analytical tools. For example, Galaxy includes 3356 different tools; myExperiment repository contains about 2700 workflows; *Bioinformatics.ca* (bioinformatics.ca) integrates more than 2100 tools and databases; *OMICtools* (omictools.com) registers about 5000, and *Nucleic Acids Research* registry includes about 1700 databases [38].

B. Necessary Ingredients and Functionality

Even if the concept of SWfs constitutes a relatively simple and old idea in the bioinformatics community, a number of critical but ignored factors constrain their wide adoption.

Embed and integrate. Just because of the *lock syndrome*, the devise of a SWf requires a significant amount of time, code libraries and configuration files for its reconstruction. So, a SWf synthesis environment should provide mechanisms for the seamless integration of its constituents', without any "*selfish*" assumption that it is the sole and unique solution for the problem. With this assumption any "*sovereignty*" behavior is avoided giving its place to "*contribution*" habits. Based on this, SWf environments should allow researchers to extract their full analysis in forms that can be easily assimilated by other contributions. *BASH* command scripts and scenarios (www.gnu.org/software/bash) with metadata described in known and easily manageable formats (e.g., *XML*, *JSON*), are typical examples that support such actions and research behaviors. Another option is writing scripts in serialized objects such as *PICKLE* (docs.python.org/3.4/library/pickle.html) or *CAMEL* (camel.readthedocs.io), which can be easily integrated with other tools.

Integrate standard and validated bioinformatics tools. The inclusion of available, validated and widely used bio-data analysis tools in SWf environments provides a valuable development framework that can quickly and easily attract users to participate and contribute with their implementations. These tools can be classified into three categories: (i) Tools for genotype-phenotype association analysis, such as *plink* (zzz.bwh.harvard.edu/plink) and *GATK* (software.broadinstitute.org/gatk); (ii) Tools for genome annotation tools, e.g., Avia (avia.abcc.ncifcrf.gov), *ANNOVAR* (annovar.openbioinformatics.org/en/latest), *GEMINI* (gemini.readthedocs.io/en/latest), SIFT (sift.bii.a-star.edu.sg), Polyphen-2 (genetics.bwh.harvard.edu/pph2); and (iii) Tools and pipelines for next generation sequencing data analysis, e.g., *PRADA* (sourceforge.net/projects/prada) suitable for RNA-sequencing, and *Molgenis-inpute* (github.com/molgenis/molgenis-imputation) for imputation of missing genotypes [39].

Semantic annotation of resources. Most SWf environments ignore the semantic content of workflows and focus mainly on the analytical part of it. *Semantic enrichment* of SWfs brings a series of key-benefits to the engaged researchers including: search ordering and filtering; autocompletion (during query formulation); content grouping (according to users' interests and focus) and personalized recommendations [40]. Formation of the appropriate *meta-data* descriptions for the research objects included in a workflow, and their *ontology-based annotation* is the medium towards SWf semantic enrichment. For example, ontologies for gene function classification, like *Gene Ontology* (GO, gene ontology.org) [41]; for genetic diversity, like *VarioML* (github.com/VarioML/VarioML) [42], as well for assessing and analyzing the evidence of target phenotypes, like *Human Phenotype Ontology* (HPO, hpo.jax.org/app) [43], are extremely useful. In any case, ontologies should not considered a panacea as they have their own issues, especially their functional, efficient and homogenized use [44].

Virtualization. The computational execution environment of a SWf is often hidden to the user that compose and tries to run a workflow, and carries its own dependencies e.g., operating system, needed code libraries and packages. These requirements make it difficult to set-up a workflow execution environment, even for skilled IT personnel and experienced bioinformaticians. In addition, the lack of documentation makes the inexperienced users to mis-configured environments, and to the mismanagement of available resources. *Virtualization*, a "package in a package" solution for all the required components (software, operating system, libraries, and packages) seems as a promising approach [45]. The virtualized "*image*" or "*container*" may run on different operating systems, making virtualization a very convenient technique towards integrating workflows developed in different and heterogeneous environments. *Docker* (www.docker.com), perhaps the most widespread virtualization infrastructure, offers services to set-up and execute virtualized containers, and its value in contemporary data analytics has been already recognized [46], [47].

Support access to widespread post-genomics resources. Major life-sciences research initiatives, consortia and projects have already created and delivered well-organized repositories with huge volumes of qualified clinico-genomic data. So far, although the identification and collection of these data are relatively simple, their volume and scatter descriptions make it difficult to integrate. Contemporary SWf

environments should offer automated methods of data collection from well-known, reliable and widely-used bio-data sources, such as *ExAC* (exac.broadinstitute.org) with ~60,000 fully qualified exoms; *1000 Genomes Project* (www.internationalgenome.org) with ~2,500 full genomes; *ENCODE* (www.encodeproject.org), the *European Genome-Phenome Archive* (EGA, ega-archive.org) with about 1900 post-genomic studies, and *TCCA multi-omics* cancer data repository (www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga) [48] with about 11.000 cancer cases.

Support visualization. In post-genomic research some visualization methods have become standard and easily interpreted by the communities involved. For example, in *genome wide association studies* (GWAS), *Manhattan* charts for detecting the significance of specific results, *QQ* charts for inflation data detection, and *PCA* charts for population stratification detection, are key ingredients of bioinformatics SWf, and should be integrated within any contemporary SWfMS.

IV. The OpenBio-C Platform

A. Basic Design Principles

One of the main challenges of the OpenBio-C platform is to enable all types of users to create their own *editions* of *research objects* that are imported (or altered) by other users. OpenBio-C guarantees that a saved research object will remain unchanged forever. To implement this, we have adopted the logic of *forks*, which is a very popular concept in software management. Each research object is accompanied by a name and a unique number for that (the *edit*). Any object that has been saved and contains some specific version cannot be changed (considered immutable). However, any user, and at the touch of a button, can fork any object, creating a copy of that object, which has exactly the same data as its version number. So, for all research objects, a *tree structure* is formed with all versions made by users (managed by the *jsTree* library, www.jstree.com). In addition, each object has its own usage statistics, its own annotation, and its own quality attributes attributed by the system.

B. The power of BASH

Nowadays, there are over a hundred tools for managing workflows. It is very common for each tool to define its own job description language (also known as "Domain Specific Languages", DSL). Every new language, as simple as it may be, requires from users to devote time and effort to learn it. In addition, each language requires software that checks the syntactic and semantic correctness of the workflows described therein. The above adds unnecessary complexity to workflow systems and makes their spreading difficult. As an example, we quote that for 2018, the number of published jobs that used Galaxy was about 1,000 (https://galaxyproject.org/galaxy-project/statistics/). If we assume that Galaxy is the most widespread bioinformatics workflow environment, and that more than 400,000 scientific works are published in the areas of biology, genetics and bioinformatics, we may conclude that the overwhelming majority of analyzes do not use a workflow management environment! However, any task that requires the combination of more than one tool, or requires a tool that is only available in a Linux environment, then it utilizes the *BASH* environment. BASH is the most common command-line interface in Linux and OSX (there are also versions for Windows). Although BASH is not a workflow management environment, it is basically an environment where the individual steps of a workflow can be described. Also, most existing workflow management environments encode or describe the individual steps of a workflow in BASH format [50]. The idea is to realize all workflow functions in BASH, while at the upper level the user handles the BASH scripts in a user-friendly environment. The basic design criterion for these tools is to "*conceal*" the user from the complexity and difficulty of BASH. The innovation of OpenBio-C is that it brings BASH at the "*surface*". The user does not need to learn any new programming language or specialized description language.

C. Importing tools and composing workflows

Tool dependencies. In the Bioinformatics scientific community, there are tools considered as "basic" and tools that are more "experimental" (which are based on "basics" but rarely used). There are also common programming libraries (e.g., statistical, linear algebra, numerical analysis etc) that are used by a variety of other tools. This creates an *interdependence* between tools and libraries. If a research object that models and ultimately represents a research tool does not contain installation instructions for the libraries on which it depends, then it is practically useless. On the other hand, if a small workflow constituent contains information about its potential dozens of dependencies, its maintenance and overall management becomes complex and impractical. An innovation of OpenBio-C is that the user can declare a tool as dependent on one or more other tools. These tools are independent research objects, exist in different records in the database, and have their own usage statistics and annotations. When a tool is fork(ed), it adopts the dependencies of the original tool (the user of course can change them). So, each tool is accompanied by its *dependency* tree (also managed by the jsTree library). This tree is visible in the graph-based workflow synthesis environment of OpenBio-C.

Installing tools. To install a tool, the user simply inserts the BASH commands that install it. Similarly, when describing a *step* of a workflow, it simply enters the BASH commands that perform this step.

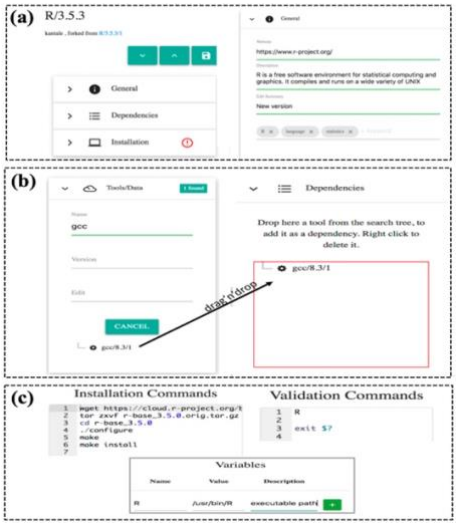

Fig. 2. Importing a tool in OpenBio-C.

OpenBio-C (i) provides a user-friendly online environment for processing these commands; (ii) stores the commands in the database accompanied by the annotation of the research objects used to model the engaged tools and workflows, and of course (iii) execute them in a user-defined

environment. OpenBio-C utilizes JavaScript library with special *visual aids* to facilitated insertion and editing of BASH commands, including *autocompletion* functionality. When entering (importing an existing or creating a new) tool, the user may: fill-in fields about the tool's general information (site, description, keywords, see Fig 2(a)); import other tools in the case that the tool depends (dependencies) by other tools by dragging-and-dropping from the respective dependency tree (Fig. 2(b)); write down the BASH commands that install the tool or download the respective data (Fig. 2(c), up-left part), and the BASH commands that confirm the correct installation of the tool (Fig. 2(c), up-right part). The variables whose values are accessible by any other tool will depend on the tool being imported; information about these variables is visible (Fig. 2(c), bottom part). Similar operations take place in order to import and compose a workflow, aided by a *graph-based workflow synthesis environment*.

D. The Execution Environment

When a user imports a tool, a dataset, or a workflow, OpenBio-C tries to confirm their correctness. To do this, the BASH commands for all tools engaged with the current workflow are collected, and sent to an independent subsystem that has its own architecture (it runs in its own computing environment). This subsystem receives *requests* from OpenBio-C's back-end to run a BASH program. Upon receiving such a request, it activates a virtual computing environment in which it executes the commands and when they terminate, it returns the execution results to the back-end. The virtual computing environment is configured through Docker. It is important to emphasize that the whole process is based on *asynchronous execution*. The back-end can send thousands of requests in a minute, which come in a queue of priorities. Subsequent processes then "take" requests from the priority queue, execute them and return the result. This enables performing all "requests" regardless of the existing computing infrastructure. Following this mode of operation, the system remains "modular" and provides scaling by adding additional computing resources. Currently, its implementation is based on the Python 3 *asyncio* library (docs.python.org/3/library/asyncio.html). In the future, however, implementation will be based on the *Kubernetes* virtual desktop computing environment (kubernetes.io).

E. The Collaboration Component

In its current version, the OpenBio-C collaboration component supports the following key features: (a) *Import* - *Edit* - *Delete* nodes in the discourse / argumentation graph; (b) *Edit* the text of a discourse block node; and (c) Provide brief information on the discussion topic through a specific *tooltip*. Every discussion is represented as an *argumentation graph* (Fig. 3). The *argumentation model* used is based on the *IBIS* (Issue-Based Information System) framework [51], following the design principles presented in [52]. Each node in the argumentation graph may belong to one of the following five categories / types: *issue* – a question / problem that a user has or a topic he/she posts for discussion; *solution* –a suggested solution to the issue under consideration; *position in favor* – an argument defending a proposed solution for the problem; *position aga*inst – a counter-argument to a proposed solution; *note* –a comment that does not affect the formal assessment of the discussion. Data from a conversation are stored in JSON files for further retrieval (i.e., recall a previous discussion).

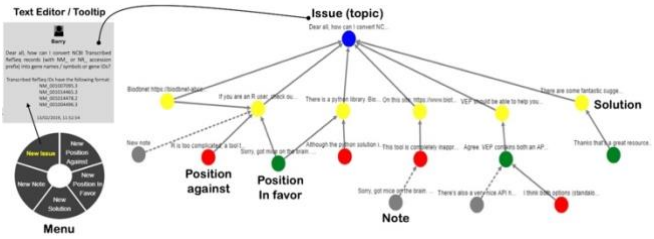

Fig. 3. The OpenBio-C collaboratrion component founded on an argumentation graph set-up.

Fig. 3 depicts the basic functionality of the collaboration component. It refers to a discussion carried out through *Reddit* (www.reddit.com), a widely used social news aggregation, web content rating, and discussion website. The specific post concerns a bioinformatics-related question posted by a user namely, "*How to convert NCBI RefSeq IDs to Gene names / symbols*"[2]. There were six solutions (answers) to the posted question by other users ("solutions", yellow nodes in Fig. 3). As expected, there are conflicts between solutions and different views. Specifically, there are arguments in favor of (green nodes) and against (red nodes) the five posted solutions. There is also the possibility to post some comments (gray nodes) linked with either the solutions or the arguments. Users of the collaboration component may create a new node in the argumentation graph by using the *menu* (left-down in Fig. 3), shown by right-clicking in the collaboration workspace. The options available concern the creation of a new issue node, the creation of a new issue position in favor or against, the creation of a new solution node, and the creation of a new note node. When inserting a node, the creator and date of creation are annotated and appropriately registered in the OpenBio-C database. Each node in the argumentation graph can be edited and even deleted by the users. The node creator has the ability to process / delete / add a description for the node through an embedded *text editor* (Fig. 3, top-left part). In the current implementation, various shortcuts (e.g., copy-paste a node) are integrated to allow users receive brief information about a node (such as its creator, description, and date of creation) via a pop-up tooltip, which is called when the user moves the mouse to a specific node. Additional functionality under development includes saving the argumentation graph in a graph database (*Neo4J*, neo4j.com), enriched search functionality, ability to expand/collapse and show/hide nodes, and assignment of like/dislike functionality for the argumentation nodes. Fig. 4 (next page) illustrates the basic components of the OpenBio-C platform, and depicts the basic data flows between them. The implementation of the system is in its beta version, which can be launched via OpenBio-C's project website, (www.openbio.eu/platform).

F. Working with OpenBio-C

As a showcase we present a simple yet indicative example of how to add tools, compose workflows, execute them and monitor their execution through the OpenBio-C platform (www.openbio.eu/platform). We assume a user with an entry level of knowledge about bioinformatics, and interested in "*creating a PCA (Principal Component Analysis) plot of the world-wide genetic heterogeneity based on the HAPMAP3 data*".

Initially the user posts a simple question on the Q&A section of OpenBio-C (see Fig. 5a).

[1] www.reddit.com/r/bioinformatics/comments/8jcvql/how_to_convert_ncbi_refseq_ids_to_gene_names

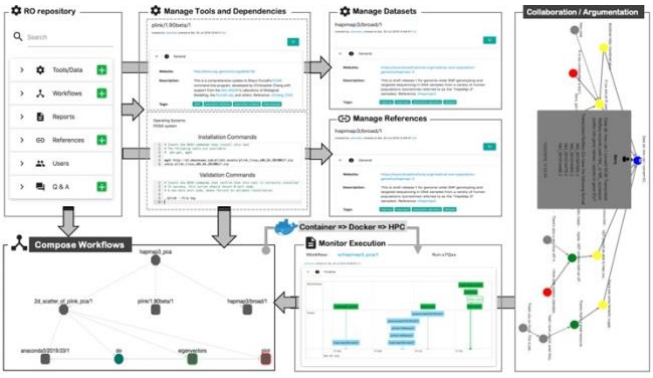

Fig. 4. The OpenBio-C SWf workflow environemnt: components and data flow.

Another user suggests that this task can be done with the *plink* tool (zzz.bwh.harvard.edu/plink), and also pinpoints how to download the *HAPMAP3* dataset. A third user with more advanced programming knowledge attempts to create the workflow in four simple steps: (i) *add* the tool by writing down the needed BASH *installation commands*, '*wget*' and '*unzip*', and *validate* the installation by inserting the BASH *installation commands* or, the commands that *confirm* successful installation (in our case one command that invokes the tool "/.*plink*", Fig. 5b). When trying to install a tool the system first executes the validation commands and if these run successfully then it considers that the tool is installed correctly. If the validation fails then the system runs the installation commands and then the validation commands again. If the validation fails again, an error message appears to the user; and finally; (ii) *set the engaged variables* by defining the engaged variables (Fig. 5c), the values of which are accessible by any other tool of the workflow that dependents from the one being added. One common use of these variables is to define the path to the executable of the tool. It is important to note that these steps come naturally as the steps that any user would perform to include a tool to a pipeline system, and do not require the knowledge of any DSL or wrapping mechanism;

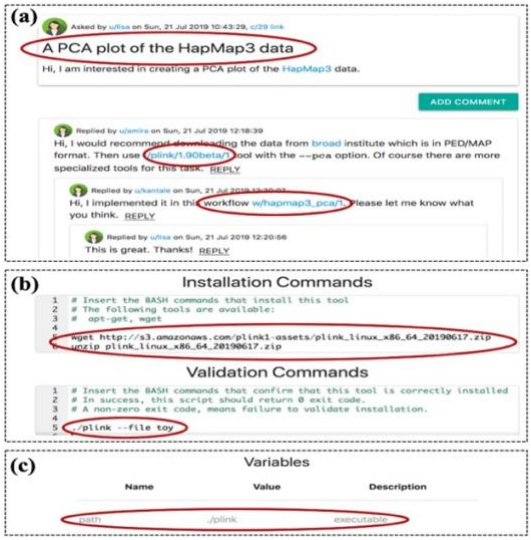

Fig. 5. Querying and adding a tool in OpenBio-C for the task "*PCA plot of HAPMAP data"*.

(iii) Then, the user saves the tool. The universal name of the tool is set to "plink/1.90beta/1", with number "1" assigned by the system to ensure uniqueness of the universal name. This means that multiple entries of the same tool/version can exist, each with different universal names. Now that the tool is imported in the system, the user can create a workflow that performs the "PCA analysis". To do so, the user navigates in the "Workflows" section of OpenBio-C and creates a new workflow with the name "hapmap3_pca/1", by simply dragging and dropping the "plink/1.90beta/1" tool in the OpenBio-C graph-based workflow editing window, and add the needed execution steps as well as other tools, e.g., PCA "plot" (Fig. 6a);a nd (v) Finally, the whole workflow is *virtualized* using the Docker infrastructure (Fig 6a); the specific image was run on a local cloud infrastructure. OpenBio-C offers graph-based monitoring of the workflow execution (Fig. 6b).

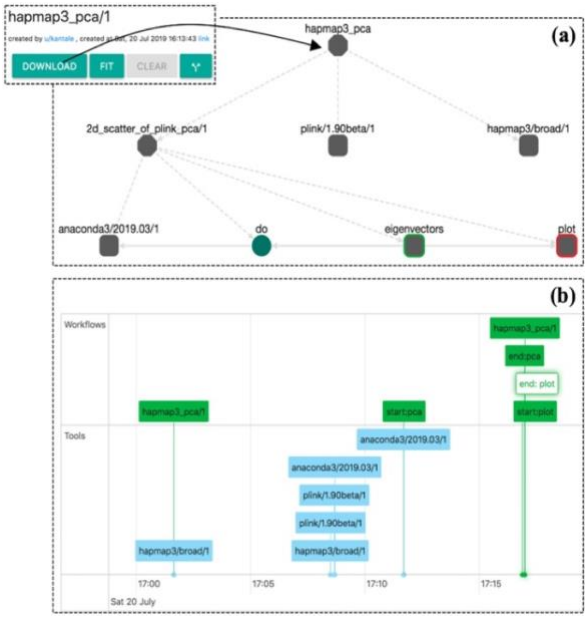

Fig. 6. Graph-based workflow editing and monitoring in OpneBio-C.

V. CONCLUSIONS

OpenBio-C provides a novel scientific workflow environment that is flexible enough to easily and meaningfully accommodate heterogeneous tasks, which are typically handled by separate systems and require specific competences. This alleviates expenses related to the large-scale data loading into multiple systems and eases the management of big data volumes as well the manipulation of various tools and services, which constitute basic concerns and top priorities of scientific research communities. Our overall approach provides researchers with the required capacity to easily assemble their own working environment using their preferred tools, as well as to exploit the wealth of existing resources and competences. Due to its flexible development approach, which takes into account the way science is organized and run, the proposed solution enables a *digital science* environment for multidisciplinary work by supporting and sustaining dynamic research communities with their associated tools, networks and practices.

ACKNOWLEDGMENT

The work presented in this paper is supported by the OpenBio-C project (www.openbio.eu), which is co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project id: T1EDK- 05275).

REFERENCES

[1] J. A. Hill, "How to Review a Manuscript," *J. Electrocardiol.*, vol. 49, no. 2, pp. 109–111, 2016.

[2] C. G. Begley and L. M. Ellis, "Raise standards for preclinical cancer research," *Nature*, vol. 483, p. 531, Mar. 2012.

[3] F. Prinz, T. Schlange, and K. Asadullah, "Believe it or not: how much can we rely on published data on potential drug targets?," *Nat. Rev. Drug Discov.*, vol. 10, p. 712, Aug. 2011.

[4] L. P. Freedman, I. M. Cockburn, and T. S. Simcoe, "The economics of reproducibility in preclinical research," *PLOS Biol.*, vol. 13, no. 6, pp. 1–9, 2015.

[5] M. R. Munafò *et al.*, "A manifesto for reproducible science," *Nat. Hum. Behav.*, vol. 1, p. 21, Jan. 2017.

[6] D. R. Grimes, C. T. Bauch, and J. P. A. Ioannidis, "Modelling science trustworthiness under publish or perish pressure," *R. Soc. Open Sci.*, vol. 5, no. 1, p. 171511, Jul. 2019.

[7] J. November, "More than Moore's Mores: Computers, Genomics, and the Embrace of Innovation," *J Hist Biol*, vol. 51, no. 4, pp. 807–840, 2018.

[8] Z. D. Stephens *et al.*, "Big data: Astronomical or genomical?," *PLoS Biol.*, vol. 13, no. 7, 2015.

[9] K. Charlebois, N. Palmour, and B. M. Knoppers, "The Adoption of Cloud Computing in the Field of Genomics Research: The Influence of Ethical and Legal Issues," *PLoS One*, vol. 11, no. 10, p. e0164347, Oct. 2016.

[10] V. Marx, "The big challenges of big data," *Nature*, vol. 498, p. 255, Jun. 2013.

[11] [T. Hulsen *et al.*, "From Big Data to Precision Medicine," *Front. Med.*, vol. 6, p. 34, Mar. 2019.

[12] K. Y. He, D. Ge, and M. M. He, "Big Data Analytics for Genomic Medicine," *Int. J. Mol. Sci.*, vol. 18, no. 2, p. 412, Feb. 2017.

[13] G. Poste, "Bring on the biomarkers," *Nature*, vol. 469, no. 7329, pp. 156–157, 2011.

[14] L. A. Levin and H. V. Danesh-Meyer, "Lost in translation: Bumps in the road between bench and bedside," *JAMA - Journal of the American Medical Association*, vol. 303, no. 15. pp. 1533–1534, 2010.

[15] J. Leipzig, "A review of bioinformatic pipeline frameworks," *Brief. Bioinform.*, vol. 18, no. 3, pp. 530–536, May 2017.

[16] M. R. Karim, A. Michel, A. Zappa, P. Baranov, R. Sahay, and D. Rebholz-Schuhmann, "Improving data workflow systems with cloud services and use of open data for bioinformatics research," *Brief. Bioinform.*, vol. 19, no. 5, pp. 1035–1050, Sep. 2018.

[17] O. Spjuth *et al.*, "Experiences with workflows for automating data-intensive bioinformatics," *Biology Direct*, vol. 10, no. 1. 2015.

[18] N. Kulkarni *et al.*, "Reproducible bioinformatics project: a community for reproducible bioinformatics analysis pipelines," *BMC Bioinformatics*, vol. 19, no. 10, p. 349, Oct. 2018.

[19] B. Giardine *et al.*, "Galaxy: A platform for interactive large-scale genome analysis," *Genome Res.* , vol. 15, no. 10, pp. 1451–1455, Oct. 2005.

[20] K. Wolstencroft *et al.*, "The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud," *Nucleic Acids Res.*, vol. 41, no. W1, pp. W557–W561, Jul. 2013.

[21] M. A. Swertz *et al.*, "The MOLGENIS toolkit: rapid prototyping of biosoftware at the push of a button," *BMC Bioinformatics*, vol. 11, no. 12, p. S12, 2010.

[22] H. Byelas, A. Kanterakis, and M. Swertz, "Towards a MOLGENIS Based Computational Framework," in *2011 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing*, 2011, pp. 331–338.

[23] R. Guimera, "bcbio-nextgen: Automated, distributed next-gen sequencing pipeline," *EMBnet.journal*, vol. 17, no. B, p. 30, 2012.

[24] S. P. Sadedin, B. Pope, and A. Oshlack, "Bpipe: a tool for running and managing bioinformatics pipelines," *Bioinformatics*, vol. 28, no. 11, pp. 1525–1526, Apr. 2012.

[25] J. Köster and S. Rahmann, "Snakemake—a scalable bioinformatics workflow engine," *Bioinformatics*, vol. 28, no. 19, pp. 2520–2522, Aug. 2012.

[26] P. Di Tommaso, M. Chatzou, E. W. Floden, P. P. Barja, E. Palumbo, and C. Notredame, "Nextflow enables reproducible computational workflows," *Nat. Biotechnol.*, vol. 35, no. 4, pp. 316–319, 2017.

[27] P. Cingolani, R. Sladek, and M. Blanchette, "BigDataScript: a scripting language for data pipelines," *Bioinformatics*, vol. 31, no. 1, pp. 10–16, Jan. 2015.

[28] K. M. Fisch *et al.*, "Omics Pipe: a community-based framework for reproducible multi-omics data analysis," *Bioinformatics*, vol. 31, no. 11, pp. 1724–1728, Jun. 2015.

[29] P.-E. Li *et al.*, "Enabling the democratization of the genomics revolution with a fully integrated web-based bioinformatics platform," *Nucleic Acids Res.*, vol. 45, no. 1, pp. 67–80, Jan. 2017.

[30] M. A. Kallio *et al.*, "Chipster: user-friendly analysis software for microarray and other high-throughput data," *BMC Genomics*, vol. 12, no. 1, p. 507, 2011.

[31] N. Karacapilidis, *Mastering Data-Intensive Collaboration and Decision Making: Research and Practical Applications in the Dicode Project*. Springer Publishing Company, Incorporated, 2014.

[32] F. H. Eemeren, R. Grootendorst, R. H. Johnson, C. Plantin, and C. Willard, *Fundamentals of argumentation theory: A handbook of historical backgrounds and contemporary developments*. Hillsdale, NJ, US: Lawrence Erlbaum Associates Inc, 1996.

[33] C. A. Goble *et al.*, "myExperiment: a repository and social network for the sharing of bioinformatics workflows," *Nucleic Acids Res.*, vol. 38, no. Web Server issue, pp. W677–W682, Jul. 2010.

[34] P. J. A. Cock *et al.*, "Biopython: freely available Python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, Jun. 2009.

[35] M. D. Wilkinson *et al.*, "The FAIR Guiding Principles for scientific data management and stewardship," *Sci. Data*, vol. 3, p. 160018, 2016.

[36] N. L. Harris *et al.*, "The 2015 Bioinformatics Open Source Conference (BOSC 2015)," *PLOS Comput. Biol.*, vol. 12, no. 2, p. e1004691, Feb. 2016.

[37] K. Hettne *et al.*, "Workflow Forever: Semantic Web Semantic Models and Tools for Preserving and Digitally Publishing Computational Experiments," in *Proceedings of the 4th International Workshop on Semantic Web Applications and Tools for the Life Sciences*, 2012, pp. 36–37.

[38] D. J. Rigden and X. M. Fernández, "The 26th annual Nucleic Acids Research database issue and Molecular Biology Database Collection," *Nucleic Acids Res.*, vol. 47, no. D1, pp. D1–D7, Dec. 2018.

[39] A. Kanterakis, P. Deelen, F. van Dijk, H. Byelas, M. Dijkstra, and M. A. Swertz, "Molgenis-impute: imputation pipeline in a box," *BMC Res. Notes*, vol. 8, no. 1, p. 359, Aug. 2015.

[40] J. Zarnegar, "Semantic enrichment of life sciences content: how it works and key benefits for researchers," *Emerg. Top. Life Sci.*, vol. 2, no. 6, pp. 769 LP – 773, Dec. 2018.

[41] G. O. Consortium, "Gene Ontology Consortium: going forward," *Nucleic Acids Res.*, vol. 43, no. Database issue, pp. D1049–D1056, Jan. 2015.

[42] M. Byrne *et al.*, "VarioML framework for comprehensive variation data representation and exchange," *BMC Bioinformatics*, vol. 13, no. 1, p. 254, Oct. 2012.

[43] P. N. Robinson, S. Köhler, S. Bauer, D. Seelow, D. Horn, and S. Mundlos, "The Human Phenotype Ontology: a tool for annotating and analyzing human hereditary disease," *Am. J. Hum. Genet.*, vol. 83, no. 5, pp. 610–615, Nov. 2008.

[44] J. Malone, R. Stevens, S. Jupp, T. Hancocks, H. Parkinson, and C. Brooksbank, "Ten Simple Rules for Selecting a Bio-ontology," *PLoS Comput. Biol.*, vol. 12, no. 2, pp. e1004743–e1004743, Feb. 2016.

[45] G. Juve and E. Deelman, "Scientific Workflows in the Cloud - Grids, Clouds and Virtualization," M. Cafaro and G. Aloisio, Eds. London: Springer London, 2011, pp. 71–91.

[46] C. Boettiger, "An introduction to Docker for reproducible research, with examples from the {R} environment," *CoRR*, vol. abs/1410.0, 2014.

[47] P. Di Tommaso, E. Palumbo, M. Chatzou, P. Prieto, M. L. Heuer, and C. Notredame, "The impact of Docker containers on the performance of genomic pipelines," *PeerJ*, vol. 3, pp. e1273–e1273, Sep. 2015.

[48] K. Tomczak, P. Czerwińska, and M. Wiznerowicz, "The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge," *Contemp. Oncol.*, vol. 19, no. 1A, pp. A68–A77, Jan. 2015.

[49] J. C. Mu *et al.*, "VarSim: a high-fidelity simulation and validation framework for high-throughput genome sequencing with cancer applications," *Bioinformatics*, vol. 31, no. 9, pp. 1469–1471, May 2015.

[50] F. B. Menegidio *et al.*, "Bioportainer Workbench: a versatile and user-friendly system that integrates implementation, management, and use of bioinformatics resources in Docker environments," *Gigascience*, vol. 8, no. 4, Apr. 2019.

[51] W. Kunz and H. W. J. Rittel, *Issues as Elements of Information Systems*, no. 131. Institute of Urban and Regional Development, University of California, 1970.

[52] O. Scheuer, F. Loll, N. Pinkwart, and B. M. McLaren, "Computer-supported argumentation: A review of the state of the art," *Int. J. Comput. Collab. Learn.*, vol. 5, no. 1, pp. 43–102, 2010.