Short: Improving reproducibility in bioinformatics workflows with BioFlow-Model

Mouna El Garb
mouna.el-garb@liris.cnrs.fr
Université Claude Bernard Lyon 1, CNRS, INSA Lyon, LIRIS, UMR5205
F-69622 Villeurbanne, France

Emmanuel Coquery
emmanuel.coquery@liris.cnrs.fr
Université Claude Bernard Lyon 1, CNRS, INSA Lyon, LIRIS, UMR5205
F-69622 Villeurbanne, France

Fabien Duchateau
fabien.duchateau@liris.cnrs.fr
Université Claude Bernard Lyon 1, CNRS, INSA Lyon, LIRIS, UMR5205
F-69622 Villeurbanne, France

Nicolas Lumineau
nicolas.lumineau@liris.cnrs.fr
Université Claude Bernard Lyon 1, CNRS, INSA Lyon, LIRIS, UMR5205
F-69622 Villeurbanne, France

Abstract

Scientific workflows are crucial for managing data, but they do not fully comply with FAIR principles yet. To improve the sharing and reuse of workflow, recent models enable the representation of traces from scientific workflow executions. However they still lack of detailed or unambiguous information. In this paper, we present BioFlow-Model, a model for improving reproducibility and querying of scientific workflows. It extends existing models when possible and provide new concepts where needed. We have also proposed mappings with existing models to increase interoperability.

CCS Concepts

• **Information systems → Data provenance**; **Semantic web description languages**; *Ontologies*; • **Applied computing → Bioinformatics**.

Keywords

Reproducibility, Workflows, Modelling, Meta-model, Bioinformatics

ACM Reference Format:
Mouna El Garb, Emmanuel Coquery, Fabien Duchateau, and Nicolas Lumineau. 2025. Short: Improving reproducibility in bioinformatics workflows with BioFlow-Model. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (REP'2025).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The increased adoption of workflows, especially in bioinformatics and medicine [9], has been accompanied by the development of various systems designed to implement and manage them [16]. The main advantages of using workflow management systems (WfMS) include the definition and orchestration of a complete scientific process (usually as a script), the management of dependencies, the execution on different platforms and servers, and the re-entrancy capability (i.e., to resume part of a workflow execution). As workflows produce intermediate outputs and may be resource-consuming, they can be shared and reused if they are sufficiently described and easily accessible (FAIR principles [15]). To facilitate this reuse, workflow developers have turned to collaborative platforms (e.g., GitHub) and promoted initiatives such as nf-core [5] and bio.tools [7] for defining best practices, curating and standardizing metadata. Finally, Common Web Language (CWL) and Workflow Description Language (WDL) have been designed to increase interoperability between workflow languages [3].

Despite these initiatives, searching for an existing workflow is still challenging due to insufficient descriptions [2]. To improve workflow search, a formal description of bioinformatic workflows, including their provenance trace, has been proposed in PROV-O [8], P-Plan [6] and Ro-Crate [10]. However, these representations remain limited, as no model fully addresses all aspects of workflows (e.g., detailed information about scripts used at each step, limited control flow representation).

To tackle these issues, we propose BioFlow-Model, a structured and fine-grained description of a workflow using metadata and provenance in a generic, machine-readable, and workflow-engine-independent format. The main objective of this model is to allow users to build specific queries to find shared workflows. It includes a set of classes and properties to represent elements of the workflows that could be useful to search for, such as the sequence of the steps, the tools used and the control flow specification. It is worth noting that classes and properties are reused from existing ontologies when relevant. To promote interoperability, we also provide a set of mappings with several popular models, including schema.org, PROV, P-Plan and RO-Crate.

The next section deals with related work, and Section 3 describes our novel model BioFlow-Model. We discuss its benefits and limitations and compare it to existing models in Section 4. We conclude and outline perspectives in Section 5.

2 Related work

This section briefly describes workflow management systems, and the existing provenance models used in these systems.

REP'2025, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

2.1 Workflow management systems

Many WfMS have been proposed to structure the development and management of computational workflows (360 on this list[1]). In this work, we focus on the key features of Nextflow and Snakemake systems, with the possibility of extending to other WfMS.

Nextflow enables bioinformaticians to develop structured workflows, driven by either data flow or control flow [4]. It implements a Domain-Specific Language (DSL) built on top of Groovy (Java), with a top-down approach promoting a dynamic ordering of steps based on previous results. Nextflow supports several control flow patterns, such as conditionals, iterative processing, and parallel splitting. A process is the basic processing unit, with input, user script and output requirements. Traces related to the execution of a workflow can be captured using the nf-prov plugin[2].

Snakemake is also a workflow management system designed to create reproducible and scalable data analyses [13]. Workflows are defined using a human-readable, output-oriented, Python-based language, which relies on rules and filename patterns to determine execution order. Snakemake handles some control patterns, including sequence, conditionals and parallel splits (by specifying the number of jobs). Snakemake automatically generates detailed self-contained HTML reports that encompass runtime statistics, provenance information, workflow topology and results.

Although the most popular systems include execution traces, additional models have been proposed to improve provenance.

2.2 Provenance models

The W3C PROV standard defines provenance as "a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing"[3].

Two types of provenance have been identified in the literature: **prospective provenance** deals with the abstract specification of a workflow by providing a structured representation of its key elements while **retrospective provenance** refers to the actual execution of a workflow and its steps, including data used or generated, details about resource consumption and execution time [11].

The specification of interoperable workflows has already garnered significant attention from various standardization bodies and scientific communities, resulting in several established specifications. The P-Plan ontology for provenance and plans [6] mainly extends the W3C PROV-O ontology [8]. RO-Crate describes workflows, their steps, data dependencies, and, to a lesser extent, control dependencies between steps in the WFRUN ontology [10]. The differences among these specifications primarily lie in the level of detail they provide in capturing workflow descriptions.

2.3 Positioning

P-Plan extends the PROV-O ontology to represent the plans that guided the execution of scientific processes, describing how these plans are structured and how they correspond to provenance records of the execution. Yet it has several limitations. First this model introduces novelty by capturing the sequence of steps using the transitive property `p-plan:isPrecededBy`, but it does not explicitly capture significant details such as the scripts, tools, or commands invoked in each step. Additionally, it does not provide detailed annotations for data reuse, nor does it track computational resources, or workflow engine information.

As for RO-Crate, it defines the Workflow Run Crate profile, which describes the execution of a computational workflow that orchestrates the execution of other tools. It also introduces the Provenance Run Crate profile, which extends the Workflow Run Crate by adding specifications to capture internal details of the workflow run, such as step executions and intermediate outputs. However, these two profiles have limitations. They do not explicitly differentiate between the software tools used in a step, whether they are scripts, bioinformatics tools (e.g., Samtools), or command-line executions. Besides, these profiles mainly focus on data flow rather than control flow mechanisms (e.g., loops, conditionals). Although RO-Crate allows for some representation of workflow execution details, it does not systematically provide metadata about the computational environment, such as the software and the hardware configurations.

To conclude, key elements such as sub-workflows, detailed information about scripts used at each step are not explicitly defined and the naming conventions used to represent workflow information can be ambiguous. The next section describes a novel model which overcomes some of these limitations.

3 BioFlow-Model

Our main contribution is a model for representing bioinformatics workflow with prospective and retrospective provenance: BioFlow-Model. Figure 1 (in Appendix A) depicts our model, where white box classes represent prospective provenance and coloured box classes stand for retrospective provenance. Properties are drawn as plain oriented arrows while inheritance links (`rdfs:subclassOf`) use dashed lines. For readability reasons, this visualization only displays the most significant properties (e.g., `rdfs:label` properties are not shown). A full version is available on 10.5281/zenodo.14945692 and it includes a description of its classes and properties, a list of mappings to existing ontologies and a RDF Turtle version.

BioFlow-Model builds on existing ontologies by integrating well defined classes that already describe accurately workflow aspects while introducing new concepts to address elements that were misrepresented or entirely missing. Table 1 provides statistics about classes and predicates in BioFlow-Model. We have introduced 17 classes and 15 properties (e.g., `sf:Branch`, `sf:triggers`), for instance to represent the control flow and provenance trace elements, which are useful for workflows driven by control patterns. We have reused 22 concepts from SCHEMA, 4 concepts from PROV, 3 from P-PLAN, 3 from EDAM[4] (used in other ontologies to model scientific data management) and 8 from WFRUN. We observe that many existing properties (66%) were relevant (especially from SCHEMA), but less classes could be directly reused (39%).

Prospective concepts. As shown in Figure 1, BioFlow-Model captures prospective provenance, which defines the abstract specification of a workflow by providing a structured representation

[1]List of existing workflow systems, https://s.apache.org/existing-workflow-systems (accessed 06/2025)
[2]nf-prov, https://github.com/nextflow-io/nf-prov (accessed 06/2025)
[3]Provenance notation, https://www.w3.org/TR/prov-n/ (accessed 06/2025)
[4]EDAM ontology, https://edamontology.org/ (accessed 06/2025)

Table 1: Statistics about classes and properties from our BioFlow-Model model.

Concept	All	New	SCH	PROV	P-Plan	EDAM	WFR
`Class`	28	17	3	3	0	2	3
`Property`	44	15	19	1	3	1	5

Table 2: Statistics about mappings between BioFlow-Model and existing ontologies.

	SCH	PROV	P-Plan	EDAM
Same class	3	3	0	2
Inheritance class	14	2	6	4
Same property	15	1	3	1
Inheritance property	0	2	2	0

of its key elements. The workflow itself is represented by the class `sf:Workflow`. Its key characteristics and elements are defined using specific classes: the topic with `sf:Topic`, the steps with `sf:Step`, the sub-workflows with `sf:Subworkflow` (linked to other classes through P-PLAN predicates), and the responsible agents with the `prov:Agent` class. By adapting SWCF, a generic provenance model to capture the behaviour of control dependencies [1], our model defines a new class, `sf:Startable`, which unifies all executable steps (steps and control operators are sub-classes of `sf:Startable`). Workflow control operators are represented by the class `sf:ControlOperator`, with the property `sf:hasBranch` specifying the branch associated with a control operator. A branch can trigger either a workflow step or another control operator. For instance, an *if* condition may check the type of input data and activate a corresponding branch, which in turn triggers the appropriate step for handling that specific data type. The different software (tools, commands, or scripts) have their own classes and are linked to a step using a predicate `sf:use`. Input and output variables for both steps and workflows are represented by the properties `sf:inputVariable` and `sf:outputVariable`, which reference the class `sf:Variable`.

Retrospective concepts. BioFlow-Model uses new classes to represent step and workflow executions, namely `sf:StepExecution` and `sf:WorkflowExecution`. They are linked to the classes representing their specifications (i.e., `sf:Workflow` and `sf:Step`) using the `sf:correspondsToWorkflow` and `p-plan:correspondsToStep` properties. Data used during execution is another aspect to address in retrospective provenance. In BioFlow-Model, it is represented with the `sf:Entity` class, which can be linked to its variable and to the agent (using predicates `sf:correspondsToVariable` and `prov:wasAttributedTo`). New properties `sf:WorkflowExecution` and `sf:StepExecution` may use or generate `sf:Entity`, as illustrated by the predicates used to link them. Additional information about the execution (start and end times, consumed resourced) can also be captured and stored.

Mappings. Defining mappings between BioFlow-Model's concepts and those from existing ontologies is crucial for interoperability. As one of the objectives of this model is to build a query language that can retrieve workflows, mappings would thus expand the scope of workflow research and provide better results for users. Table 2 summarizes the number of mappings either by using inheritance properties (`rdfs:subClassOf` and `rdfs:subPropertyOf`), or by directly using the same class or property of the ontology.

4 Discussion

BioFlow-Model provides both prospective and retrospective provenance, it is capable of representing bioinformatics workflows such as those designed with Snakemake or Nextflow, with possible extensions to other systems. To assess the coverage of our model, table 3 provides a comparative overview of the features proposed by BioFlow-Model compared to CWL, PROV + P-Plan and the WRROC profiles of RO-Crate, using the categories from Leo *et al.* [10] (and detailed on this Github page). Scientific context category refers to design choices and parameter values. Data category includes input and intermediate outputs. Software category covers tools orchestrated by the workflow, and their dependencies. Workflow category focuses on the workflow and tool descriptions, but not the software they control. Computational environment category captures metadata about hardware and software on which the workflow is executed. Execution details category include additional information about the workflow run. In the table, the symbols (✔), (~) and (–) respectively stand for a feature completely covered, a feature partially covered and a missing feature. For instance, the first subcategory (i.e, workflow design) indicates that the design of the workflow and its steps is partially modelled by PROV+P-Plan, and fully represented in CWL, WRROC and BioFlow-Model. The last column in the table specifies the concepts provided in our model to support the subcategory. We observe that CWL is dedicated to the description of software and workflow, and partially deal with data. PROV + P-Plan mainly focus on describing (partially) the scientific context and data. WRROC supports more features, especially at the software and workflow levels. However, it mainly relies on generic concepts from schema.org, which have broad definitions that are not always specific to workflows. Among the models compared in the table, BioFlow-Model is the only one to support key workflow patterns, including sequence, conditional, and parallel split, which are already proposed by WfMS. However, we note that BioFlow-Model partially covers data file characteristics subcategory. This is because it uses the class `sf:Entity` to represent an actual data, extending `prov:Entity`, which does not provide specific properties to fully describe file data. Overall, BioFlow-Model covers the same categories as WRROC. But it defines its own classes by inheriting from existing ontologies, mostly schema.org. This inheritance refines the meaning of generic concepts to better fit scientific workflows. It also preserves interoperability with popular vocabularies. To validate the expressiveness of BioFlow-Model, we applied it to real-world workflows from three widely-used systems: MethylSeq (Nextflow), Kraken2 (Snakemake) and CESM (Galaxy) are briefly described in Appendix B and available on the repository 10.5281/zenodo.14945692. They are concrete examples of the capabilities of BioFlow-Model. The last example shows that our model can be used to represent workflows designed with other systems (Galaxy in this case). Given its domain (Earth simulation), it also

Table 3: Feature comparison between BioFlow-Model and existing models (CWL, P-PLAN and RO-Crate).

Category	Subcategory	CWL	P-Plan	WRROC	BioFlow-Model	How supported in BioFlow-Model
Scientific context	Workflow design	✔	~	✔	✔	`sf:Workflow, sf:Step, sf:Subworkflow, sf:Variable, sf:ControlOperator`
	Entity annotations	~	~	✔	✔	`URI, schema:name, schema:description, schema:encodingFormat, schema:additionalType`
	Workflow execution ann.	-	~	✔	✔	`sf:WorkflowSystem, sf:WorkflowExecution, sf:StepExecution, schema:actionStatus, schema:startTime, schema:endTime`
Data	Data identification	~	~	✔	✔	`URI, prov:wasAttributedTo, edam:has_format, sf:source`
	File characteristics	~	-	✔	~	`schema:name, edam:has_format`
	Data access	~	-	~	✔	`sf:source, prov:wasAttributedTo`
	Parameter mapping	✔	~	✔	✔	`sf:correspondsToVariable, sf:inputVariable, sf:outputVariable`
Software	Software identification	✔	-	✔	✔	`URI, schema:name, schema:description, sf:Script, schema:version, sf:Tool, sf:Command, sf:Library`
	Software documentation	✔	-	✔	✔	`URI, schema:name, schema:description, schema:version`
	Software access	✔	-	✔	✔	`URI, schema:license`
Workflow	Workflow software	✔	-	✔	✔	`sf:Tool, sf:Command, sf:Script, sf:Library`
	Workflow parameters	✔	~	✔	✔	`sf:inputVariable, sf:outputVariable, schema:encodingFormat, schema:additionalType`
	Workflow requirements	✔	-	~	-	
Computational environment	Software environment	-	-	-	-	
	Hardware environment	-	-	-	-	
	Container image	~	-	~	~	`wfrun:ContainerImage, wfrun:SIFImage, wfrun:tag, wfrun:DockerImage, wfrun:registry`
Execution detail	Execution timestamps	-	~	✔	✔	`schema:startTime, schema:endTime`
	Consumed resources	-	-	✔	✔	`wfrun:resourceUsage`
	Workflow engine	-	-	✔	✔	`sf:WorkflowSystem`
	Human agent	-	✔	✔	✔	`prov:Agent, prov:Person, prov:Organization`
Control flow		-	-	-	✔	`sf:Startable, sf:ControlOperator, sf:Branch`

supports the idea that BioFlow-Model is not strictly dedicated to bioinformatics, but to broader scientific contexts.

5 Conclusion

The representation of scientific workflows has been largely studied during the last decades. Although standards are emerging for workflow languages, a fine-grained representation of their execution is still expected to query and retrieve relevant pieces of workflows (FAIR principles). We proposed BioFlow-Model, which builds on existing models such as P-Plan and RO-Crate. It reduces ambiguity for several concepts and it improves the definition of sub-workflows, software and control flow. Subclasses and mappings facilitate its use and interoperability with existing models.

The first perspective deals with feedback from the bioinformatics community, for instance users or contributors of similar models (e.g., P-Plan, RO-Crate). Integrating missing concepts such as computational environment (e.g., about container images), underlying hardware environment, mechanisms for identifying and tracking data or additional control flow patterns [14] is an ongoing work. The next essential challenge concerns the adoption of BioFlow-Model. We are currently working with authors of Bioflow-Insight to extract workflow metadata according to our model [12], and we plan to design a benchmark to evaluate the quality of the workflows' representation. As our model serves as the foundation of an upcoming workflow query language, typical use cases for retrieving relevant (parts of) workflows need to be identified and illustrated. From these examples, the main operators of the query language should be defined to enable bioinformaticians to query workflows based on specific attributes derived from the model, such as the tools used, the types of data processed, or the dependencies between steps.

Acknowledgments

This work was funded by a government grant managed by the Agence Nationale de la Recherche under the France 2030 program, reference ANR-22-PESN-0007 (ShareFair project).

References

[1] Anila Sahar Butt and Peter Fitch. 2021. A provenance model for control-flow driven scientific workflows. *Data & Knowledge Engineering* 131-132 (2021), 101877. https://doi.org/10.1016/j.datak.2021.101877

[2] Sarah Cohen-Boulakia, Khalid Belhajjame, Olivier Collin, Jérôme Chopard, Christine Froidevaux, Alban Gaignard, Konrad Hinsen, Pierre Larmande, Yvan Le Bras, Frédéric Lemoine, Fabien Mareuil, Hervé Ménager, Christophe Pradal, and Christophe Blanchet. 2017. Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities. *Future Generation Computer Systems* 75 (2017), 284–298. https://doi.org/10.1016/j.future.2017.01.012

[3] Michael R. Crusoe, Sanne Abeln, Alexandru Iosup, Peter Amstutz, John Chilton, Nebojša Tijanić, Hervé Ménager, Stian Soiland-Reyes, Bogdan Gavrilović, Carole Goble, and The CWL Community. 2022. Methods included: standardizing computational reuse and portability with the Common Workflow Language. *Commun. ACM* 65, 6 (May 2022), 54–63. https://doi.org/10.1145/3486897

[4] Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo, and Cedric Notredame. 2017. Nextflow enables reproducible computational workflows. *Nature biotechnology* 35, 4 (2017), 316–319.

[5] Philip A Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm, Maxime Ulysse Garcia, Paolo Di Tommaso, and Sven Nahnsen. 2020. The nf-core framework for community-curated bioinformatics pipelines. *Nature biotechnology* 38, 3 (2020), 276–278. https://nf-co.re/

[6] Daniel Garijo Verdejo and Yolanda Gil. 2012. Augmenting prov with plans in p-plan: scientific processes as linked data. In *Proceedings of the Second International Workshop on Linked Science*. CEUR Workshop Proceedings, CEUR, Germany, 1–4. http://purl.org/net/p-plan

[7] Jon Ison, Hervé Ménager, Bryan Brancotte, Erik Jaaniso, Ahto Salumets, Tomáš Raček, Anna-Lena Lamprecht, Magnus Palmblad, Matúš Kalaš, Piotr Chmura, et al. 2020. Community curation of bioinformatics software and data resources. *Briefings in bioinformatics* 21, 5 (2020), 1697–1705.

[8] Timothy Lebo, Satya Sahoo, Deborah McGuinness, Khalid Belhajjame, James Cheney, David Corsar, Daniel Garijo, Stian Soiland-Reyes, Stephan Zednik, and Jun Zhao. 2013. Prov-o: The prov ontology. https://www.w3.org/TR/prov-o/

[9] Jeremy Leipzig. 2016. A review of bioinformatic pipeline frameworks. *Briefings in Bioinformatics* 18, 3 (03 2016), 530–536. https://doi.org/10.1093/bib/bbw020 arXiv:https://academic.oup.com/bib/article-pdf/18/3/530/25408774/bbw020.pdf

[10] Simone Leo, Michael R Crusoe, Laura Rodríguez-Navas, Raül Sirvent, Alexander Kanitz, Paul De Geest, Rudolf Wittner, Luca Pireddu, Daniel Garijo, José M Fernández, et al. 2024. Recording provenance of workflow runs with RO-Crate. *PLoS one* 19, 9 (2024), e0309210. https://doi.org/10.1371/journal.pone.0309210

[11] Chunhyeok Lim, Shiyong Lu, Artem Chebotko, and Farshad Fotouhi. 2010. Prospective and Retrospective Provenance Collection in Scientific Workflow Environments. In *2010 IEEE International Conference on Services Computing*. IEEE, Miami, US, 449–456. https://doi.org/10.1109/SCC.2010.18

[12] George Marchment, Bryan Brancotte, Marie Schmit, Frédéric Lemoine, and Sarah Cohen-Boulakia. 2024. BioFlow-Insight: facilitating reuse of Nextflow workflows with structure reconstruction and visualization. *NAR Genomics and Bioinformatics* 6, 3 (08 2024), lqae092. https://doi.org/10.1093/nargab/lqae092 arXiv:https://academic.oup.com/nargab/article-pdf/6/3/lqae092/58744836/lqae092.pdf

[13] F Mölder, KP Jablonski, B Letcher, MB Hall, CH Tomkins-Tinch, V Sochat, J Forster, S Lee, SO Twardziok, A Kanitz, A Wilm, M Holtgrewe, S Rahmann, S Nahnsen, and J Köster. 2021. Sustainable data analysis with Snakemake. *F1000Research* 10, 33 (2021), 1–30. https://doi.org/10.12688/f1000research.29032.2

[14] NC Russell, Arthur HM Ter Hofstede, Wil MP Van Der Aalst, and NA Mulyar. 2006. *Workflow control-flow patterns: A revised view*. Technical Report BPM-06-22. BPMcenter. 135 pages.

[15] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data* 3, 1 (2016), 1–9.

[16] Laura Wratten, Andreas Wilm, and Jonathan Göke. 2021. Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers. *Nature methods* 18, 10 (2021), 1161–1168.

A Visualization of BioFlow-Model

Figure 1 depicts BioFlow-Model. White box classes represent prospective provenance and coloured box classes stand for retrospective provenance. Properties are drawn as plain oriented arrows while inheritance links (`rdfs:subclassOf`) use dashed lines. For readability reasons, note that this visualization only displays the most significant properties (e.g., `rdfs:label` properties are not shown).

B Examples of workflow

Three examples are presented. The RDF code was manually written from the workflow specifications (prospective provenance) and the pictures generated with a RDF-to-SVG tool. Their main objective is not to evaluate the relevance of the transformed data, but rather to show the capabilities of BioFlow-Model for representing workflows. The RDF code and full pictures of each example are available on 10.5281/zenodo.14945692.

B.1 Methylseq (Nextflow)

The first workflow deals with a pipeline for methylation sequence data (description page). It is designed with Nextflow and it consists of 2 subworkflows with 12 steps each. Figure 2 depicts a graph-based excerpt of the representation of this workflow. It highlights one of its subworkflows (bwa-meth, top-left node) linked to its parent using `p-plan:isSubPlanOf`, with its constituent steps defined by `sf:hasPart` and its sequence of steps modelled through the `sf:isFollowedBy` predicate. Inputs and outputs are captured using `sf:inputVariable` and `sf:outputVariable`, and expected data formats are described using `schema:encodingFormat`.

B.2 Kraken2 (Snakemake)

This second workflow deals with metagenomic classification and is implemented using Snakemake (description page). Figure 3 is an extract of the representation of this workflow, with a focus on metadata. The top-left node is an instance standing for the whole workflow, and it is linked to several nodes for different agents (using predicates `schema:creator` and `schema:contributor`), as well as to the programming language via `schema:programmingLanguage`.

B.3 CESM (Galaxy)

The last example, designed with Galaxy, is a workflow for running the Community Earth System Model (CESM) to simulate the state of our planet in past, present and future times (description page). It can be used to study climate change, oceanography or atmospheric chemistry and physics. It includes 4 steps. Figure 4 illustrates part of the CESM workflow representation, with a focus on the tools used (modelled as `schema:SoftwareApplication`). It is interesting for several reasons: first it is based on the Galaxy system, thus showing that our model is not limited to Nextflow and Snakemake. Besides, the design of BioFlow-Model is domain-agnostic. Indeed, the only element specific to biosciences is the use of the EDAM ontology. By ignoring or substituting EDAM with another domain-specific ontology, the model can be used in different scientific contexts such as climate change.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009

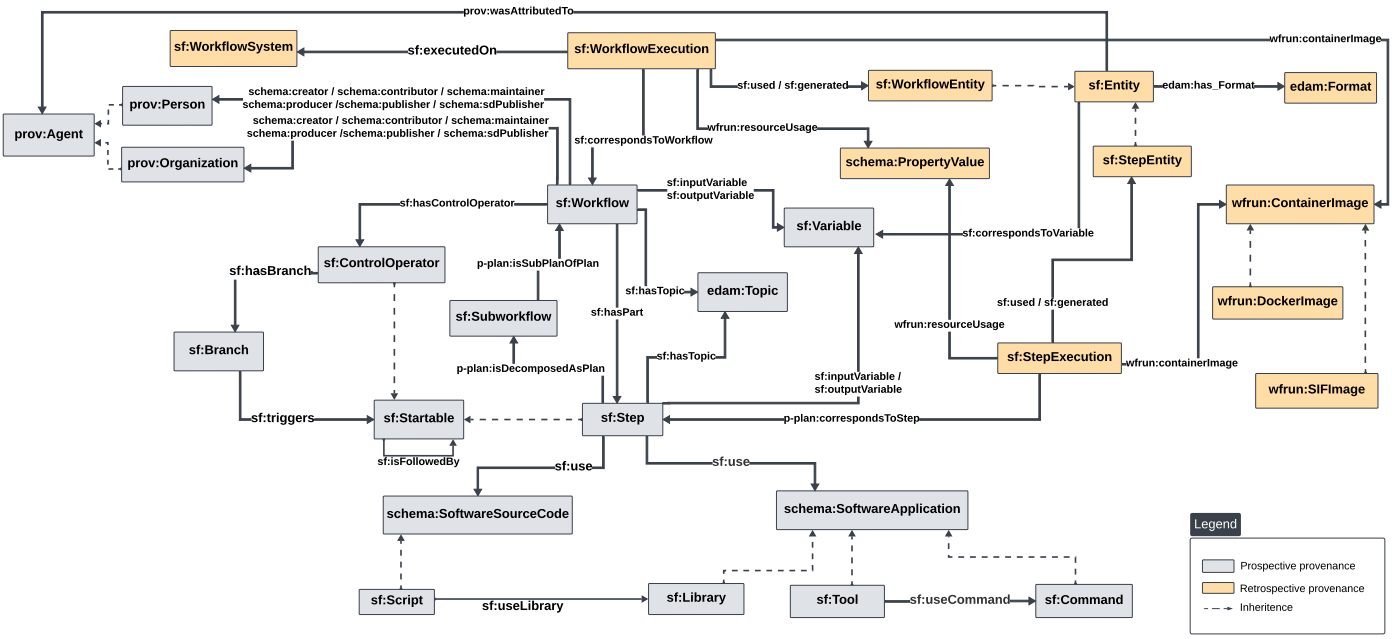

Figure 1: Overview of the BioFlow-Model.

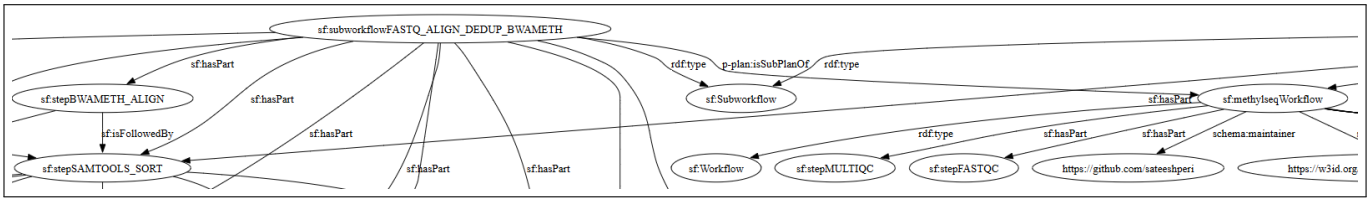

Figure 2: Extract of the Nextflow Methylseq workflow.

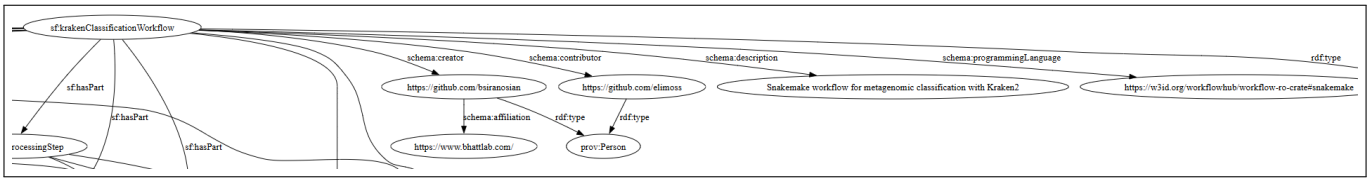

Figure 3: Extract of the Snakemake Kraken2 workflow.

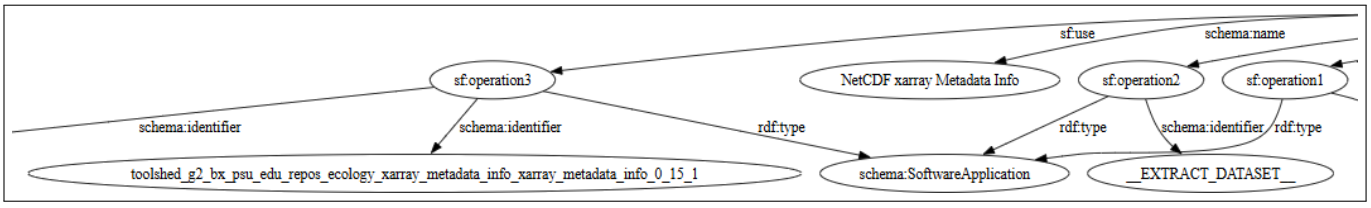

Figure 4: Extract of the Galaxy CESM workflow.