On Assisting Scientific Data Curation in Collection-Based Dataflows Using Labels

Pinar Alper, Carole A. Goble
School of Computer Science
University of Manchester
Oxford Road, Manchester, UK
first_name.last_name@cs.man.ac.uk

Khalid Belhajjame
Université Paris Dauphine
Place du Maréchal de Lattre de Tassigny
Paris, FR
Khalid.Belhajjame@dauphine.fr

ABSTRACT
Thanks to the proliferation of computational techniques and the availability of datasets, data-intensive research has become commonplace in science. Sharing and re-use of datasets is key to scientific progress. A critical requirement for enabling data re-use, is for data to be accompanied by lineage metadata that describes the context in which data is produced, the source datasets from which it was derived and the tooling or settings involved in its generation. By and large, this metadata is provided through a manual curation process, which is tedious, repetitive and time consuming.

In this paper, we explore the problem of curating data artifacts generated from scientific workflows, which have become an established method for organizing computational data analyses. Most workflow systems can be instrumented to gather provenance, i.e. lineage, information about the data artifacts generated as a result of their execution. While this form of raw provenance provides elaborate information on localized lineage traced during a run in the form of data derivation or activity causality relations, it is of little use when one needs to report on lineage in a broader scientific context. And, consequently, datasets resulting from workflow-based analyses also require manual curation prior to their publishing.

We argue that by making the analysis process explicit, workflow-based investigations provide an opportunity for semi-automating data curation. In this paper we introduce a novel approach that semi-automates curation through a special kind of workflow, which we call a Labeling Workflow. Using 1) the description of a scientific workflow, 2) a set of semantic annotations characterizing the data processing in workflows, and, 3) a library of label handling functions, we devise a Labeling Workflow, which can be executed over raw provenance in order to curate the data artifacts it refers to. We semi-formally describe the elements of our solution, and showcase its usefulness using an example from Biodiversity.

1. INTRODUCTION
In this age of “Data-Intensive Science” [15] researchers are ever more relying on computational tools and datasets to gain new scientific insights. The sharing and the re-use of valuable scientific data is of paramount importance, as these datasets are often captured through costly instrumentation, complex experiments or labor intensive methods [24].

Data re-use is facilitated through publishing datasets in community databases or archives. A crucial enabler of sharing is metadata. It is expected that data is accompanied with at least basic metadata describing 1) its Provenance i.e. origins, scientific methodology and 2) its Context i.e. assumptions, relations to other datasets and scope [7]. Such metadata is not only useful for scientists to discover 3rd party datasets but also useful for them to preserve, recall and understand their own past results, or for peers to easily review data submissions [24]. In order to promote metadata creation there is a recent proliferation in the number of community vocabularies targeted for describing, for instance, derivative relations among datasets [1][10], or, representing citations to source datasets [8].

Metadata for data publishing is created, in most cases, through a manual curation process, which is often performed after the completion of the computational analysis and just prior to sharing of the results. At this stage scientists are faced with the task of recollecting significant experimental details such as configuration settings, source datasets, external resources. This information is then expressed as Provenance and Context metadata using community vocabularies. Unfortunately scientists often have little time to spare for such curation, a recent survey [24] has shown that scientists are welcoming any methods or tools to streamline this process.

In various domains of research “scientific workflows” have become an established mechanism for weaving data processing activities into structured computational pipelines [11]. Workflows provide 1) an automation function as they are re-executable analysis pipelines 2) a methodological documentation function as they capture the process followed for the investigation. These benefits encourage scientists to invest significant effort to design their analysis as workflows [12]. Scientific workflows represent computational how-tos or best-practices, consequently they are designed once and
executed several times, with differing inputs and configurations. Executions result in the generation of intermediary and final data artifacts. Most workflow systems provide rich metadata, called workflow execution provenance through instrumented execution of workflows. Workflow provenance describes the trail of activity instantiations, their causal relationships and the derivation/lineage relations among activity input and outputs.

When we look at the publishing practices for data artifacts resulting from such workflow-based analyses we see that they are also subject to manual curation [3] to denote the results’ Provenance and Context. Raw workflow execution provenance has limited or almost no contribution to the curation of data results. It is easy to observe that there is a gap between workflow provenance and the metadata needs of scientific data publishing. The former provides a local, implementation-oriented view of the data derivation method and lineage relations among data artifacts encountered by the workflow execution engine, whereas the latter requires information on the scientific methodology, significant experimental settings, the scope of datasets, and any external sources from which data has been obtained. Often the metadata required for publishing is to be found implicitly; either in the data itself (e.g. data values, file headers, file names), or it manifests in the workflow design elements such as names of input/output parameters and activities. During manual curation the burden lies with the scientist to sift through workflow descriptions, numerous result files (from multiple runs) and provenance traces to recollect the experimental context.

In this paper, we propose a new approach for semi-automating the curation of data artifacts generated from workflow-based analyses. We argue that by making explicit the process followed for the data analysis, scientific workflows provide an opportunity, a foundation on which a curation framework can be built. Specifically, 1) We argue that the metadata required for publishing datasets is different than raw workflow provenance captured by workflow execution engines. 2) We adopt labels as a means to describe the scientific lineage and scientific context of the data artifacts. This idea of using annotations to denote origins has been previously put forward, particularly in the areas of “Where-Proofance” of database queries [4] [25]. 3) We obtain labels via curation processes called Labeling Workflows, which are comprised of steps, for label generation and propagation. 4) We propose to generate Labeling Workflows from scientific data analysis workflows, which are decorated with semantic annotations that characterize the data processing nature of their activities [13], and pinpoint associated labeling functions. Based on these annotations, we compile a Labeling Workflow, which in turn could be executed over raw workflow execution provenance so as to curate the data artifacts it refers to.

The paper is organized as follows, we first provide a real-life workflow (from Biodiversity) as a running example (Sec 2.1), this is followed by elaborating on the capabilities of state of the art in workflow provenance (Sec 2.2) and we illustrate metadata requirements of data publishing (with an example again from Biodiversity) (Sec 2.3). The second half of the paper introduces our approach from an architectural point of view and outlines our contributions (Sec 3). This is followed by sections elaborating on each contribution, namely the model of Data Labels (Sec 5), a Process-Based curation model containing four Labeling Operators (Sec 6) and an case based illustration of how Labeling Workflows are generated from annotated scientific workflows (Sec 7). We review related work (Sec 8) and conclude (Sec 9).

2. DATA-INTENSIVE BIODIVERSITY RESEARCH

2.1 Sample Scientific Dataflow

Biodiversity research includes all investigations on understanding biological diversity, its evolution and preservation. Datasets containing species taxonomies, occurrence records and genomic data are pooled in community repositories, which are made accessible on the web. Alongside data, analysis tools are also exposed for the use of the research community through web services. BIOVEL is a project that is pioneering the adoption of scientific workflows for biodiversity by building golden-exemplar workflows that bring together datasets and analysis tools for different scenarios.

Often, prior to performing any particular analysis there occurs a data collection phase. The workflow we give in Figure 1 is designed for such purpose. A set of species names that are of interest in the scope of the analysis is input to this workflow. For each name in the list the workflow retrieves occurrence records using the search services of community repositories. The workflow branch on the left hand side queries the GBIF2 data repository through a REST service (activity named “gbifOccurrence”). As the access service provides some kind of pagination functionality, for datasets that exceed 1000 occurrence records, the retrieval operation is configured to iterate until all pages of information regarding one species is collected. For each retrieval the results are to be returned in a community agreed XML format [26]. The data is stripped of its XML tagging and converted to a CSV format using an XSL transformation step (“Transform_XML”). Each page of occurrence data regarding a species makes up an item in a list of strings that contain CSV formatted data. These items in the list are flattened to a single CSV (“Merge_String_List_to_a_String”). The branch on the right hand side perform a similar retrieval operation from the SLW3 repository (“slwOccurrence”), only difference being the lack of pagination, hence the entire occurrence information from SLW repository is retrieved in a single call per species. The lists originating from the two branches are joined up (“Flatten_List”) and the resulting list of lists (depth 2) is merged into a single list (depth 1) and finally flattened into a single CSV value.

In a typical scenario, this workflow, which is designed as a golden exemplar, is to be used as a utility by several biodiversity scientists to retrieve occurrence records of different species for different investigations. The results are input to follow-on workflows that run population modeling simulations. In the remainder of this paper we refer to this workflow as the Data Retrieval (DR) Workflow.

2.2 Workflow Provenance

Figure 2 illustrates a sample execution of the DR workflow, which is run with a collection of two species names, “Cercopagis Pangoi” and “Branciura Sowerby”, as input. We

1http://www.biovel.eu/
2http://www.gbif.org/
3http://www.slu.se
Figure 1: An example workflow from Biodiversity domain. This workflow is part of a larger set of workflows developed for species population modeling.

We can see that some of the activities, including the data retrieval operations are repeatedly invoked (i.e. iterated). This is either because there are explicit iteration configurations on the activity (e.g. pagination) or because there is a cardinality mismatch between the input expected by the activity by-design and the input encountered at run-time. Certain workflow systems, in particular Taverna [21], overcomes cardinality mismatches in two ways.

- In cases where activities are designed to operate over single items and encounter collections, then Taverna repeatedly invokes the target activity with each item in the incoming collection. (This behavior is denoted with a split icon in Figure 2)
- In cases where activities expect collections rather than singletons, Taverna performs cardinality adjustment by converting all accumulated input artifacts into a list. (This behavior is denoted with square-brackets icon in Figure 2)

There is an extensive body of research on provenance in general [23] [22], and workflow provenance in particular [11]. There exist several models for representing workflow provenance [18] [3], and dedicated query languages, query APIs and browsers. Workflow provenance allows us to represent and infer activity instantiations, their causality relations among each other and the lineage relations among intermediary and final data artifacts consumed and produced by activities. This viewpoint on lineage is particularly useful 1) for workflow debugging, by allowing scientists to perform traversals over derivation relations (e.g. do these two runs with different inputs produce the same values along an execution path) or 2) for smart re-runs, by allowing the re-use of results from previous runs (e.g. re-run the SLW branch of the DR workflow with an updated SLW service end-point).

That said, when judged against the metadata needs of data publishing we observe that, while workflow provenance captures some form of “origin” information it is an implementation oriented viewpoint of data lineage that is local to the workflow execution environment (i.e. the trail of all data artifacts that the workflow engine has encountered on the computation path of activities leading to a particular result).
2.3 Metadata Required for Data Publishing in Biodiversity

When reporting their findings scientists are expected to make available the datasets used and produced during their analysis and provide information on the context and origin of data. If, for instance, the datasets are derivatives of existing ones, this relation needs to be specified with a data citation[8]. Note that citations are a very specific kind of metadata for data publishing, it is not our intention here to provide an exhaustive review of metadata requirements in data publishing. Nevertheless, in order to illustrate a special case of metadata and to discuss its coverage with provenance, we provide a sample data citation represented in one of several styles given by the popular Biodiversity data repository GBIF[8]. In order to build up this citation the scientist needs to recollect the endpoint, from which the data is retrieved, the query string used, the identifiers of datasets that contribute to the retrieved records, and the identifier that she has assigned to the derivatives.

The way scientists recollect experimental details is by scanning through several result files. Some parts of the information (e.g. IDs, query string and record counts) required to build a citation is implicitly available as information embedded in data artifacts generated during the execution of the workflow. Other parts of required information (e.g. the fact that a particular input value is a query string for data retrieval) may serendipitously be available in the workflow description as activity or port names. Against this setting a desired capability would be to have annotations over workflow results that makes explicit the above illustrated information. Note that this information would need to be sourced from the data itself, the workflow description and its execution provenance.

3. APPROACH OVERVIEW

We outline our approach through the architecture given in Figure 4. There are two processes illustrated in this figure; the design and the execution of scientific workflows (Steps prefixed by A), and the curation of resulting data artifacts with labels (Steps prefixed by B). Our approach does not interfere with the standard lifecycle of workflows (process...
A), where workflow designers create workflows (Step A.1), which are later executed (Step A.2), resulting in the creation of several data artifacts and workflow provenance traces that are stored in respective repositories (Step A.3).

The curation of data artifacts generated in process A is achieved with Labeling Workflows, which are generated and executed in a separate process (process B). To obtain Labeling Workflows, existing scientific workflow descriptions need to be manually annotated to denote the Motifs of their activities and the Label Model that is to be supported by the curation process (Step B.1). Briefly;

**Motifs** denote the data processing characteristics of each activity from a domain independent perspective (such as Data Retrieval, Merging, Filtering). Each motif outlines the data processing behavior, but also defines the labeling behavior, and associated labeling function expected of an activity. Depending on their motif, each activity is associated with either a label generation or label propagation function. Functions will be selected from a library of generic ones (e.g. copy), or could be domain specific (e.g. a labeling function for GBIF endpoints).

**Label Model** is intended as a schema for labels which will carry explicit metadata to be utilized while publishing workflow results. A label model is comprised of Label definitions and Label Vector definitions. These are intended as a basic metadata schema, a basic set of attributes to be tracked for the data that can be generated during the execution of a workflow. Note that label and label vector definitions can be shared among several workflows, in fact we anticipate that label model definitions are to be made at the investigation level, which spans multiple workflows.

Given the supported label model for a workflow description and motif annotations on its activities, we then generate a Labeling Workflow (step B.2). This labeling workflow can be applied to the provenance trace of a selected workflow, which will result in generation of labels for data artifacts (step B.3).

Following from the outlined approach, in this paper we make the following contributions that we elaborate in the rest of the paper:

- A model of Labels for carrying metadata regarding data artifacts generated in workflow based scientific data analysis.
- A process model for automating curation of intermediary and final data results of workflows.
- A case study illustrating the generation of Labeling Workflows containing operators from the process model.

4. **MOTIFS IN SCIENTIFIC WORKFLOWS**

When to generate new labels and when to relay them is mainly dependent on when data is generated and relayed during the execution of a workflow. In previous work [13] we performed an empirical analysis of 200+ scientific workflows from various workflow systems and diverse domains. The analysis was intended to obtain a categorization of data-processing activities in workflows from a domain independent perspective. The categorization resulted in a catalog of Motifs for Scientific Workflows. The analysis has shown that a certain and minority group of activities in workflows perform the scientific heavy lifting in a workflow. These steps are responsible for creating new data either from an analysis/visualization or by retrieving data from external sources. For example, the “gbifOccurrence” and “slwOccurrence” data retrieval activities in our biodiversity workflow are such steps. The remainder majority activities can be broadly categorized as Data Preparation steps. These helper steps mainly act as adapters that glue together significant activities, or they are dedicated for the local organization of data. (Some of these motifs can be likened to relational query operators, such as Join or Select. The difference is: motifs are high level classifications that would give a rough

\[\text{Represented in a light-weight ontology}\]

http://purl.org/net/wf-motifs)
idea of the data processing rather than an explicit traceable behavior). In Figure 5 we denote the motifs of each activity in call-outs.

A common characteristic of all Data Preparation steps is that the nature of their data processing is largely comprised of value-copying from the inputs of an activity to its outputs. Note that this copying could be inexact as well. Following from Figure 5 the “Transform/XML” steps transforms the XML formatted occurrence data to a CSV format, the “Merge/String/List/To/a/String” and “Flatten/List” steps consume a list of strings and produce a single string by concatenating all input strings with a separator.

We postulate that in cases of such value-copying, the labels of the input data can be transferred to the output data. Our analysis has shown that a very large percentage of operations in workflows (90+) is categorizable with a motif, i.e. as a data minting or relaying step. This also serves as a justification for introducing a metadata management model based on label generation and propagation.

In scientific workflows each activity has a technical grounding with which it is implemented. For the majority of activities, i.e. above-mentioned Data Preparation steps, in which value-copying occurs, we have access to the computing instructions in the form of scripts (e.g. Phyton, R, Bean-shell). By using techniques such as static code analysis or programme slicing one could learn these value-copying relations. Similarly using information sources like activity and port names it could be possible to semi-automate the classification of each workflow activity with its motif. This area of investigation is currently left out of the scope of our work.

4.1 Specifying Motifs through Workflow Activity Annotations

We specify the motif of an activity and the corresponding labeling behavior in the curation process by semantic annotations over the workflow. Annotations refer to the Motif Ontology. In Figure 6 we provide a fragment that partially depicts the markup of two sample activities in the DR Workflow. The fragment tells us that the “gbifOccurrence” activity in the DR workflow has the DataRetrieval motif, whereas the “Merge/String/List/To/a/String” activity has the Merging motif. The labeling behavior that the curation process should exhibit over the input/output data artifacts of these activities is inferred based on the motif of the activity. To exemplify, the motif ontology states that the Merging motif by-default corresponds to label propagation behavior. (This is realized with restriction on the isLabelingSpecFor property stating that any spec associated with a motif of type Merging is also classified as a label copying function denoted with the Copy class, which is defined to be a subclass of the Propagate class). The operational behavior implied by the Propagate operator is applying the designated function (in this case a copy) to the labels of the data artifact of the source port of an activity and associating the resulting labels to the data artifacts at the sink port of the activity. As seen in Figure 6 the source and the sinks of propagation are also specified during motif annotations. (We shall elaborate label operators in the next section).

In scientific workflows each activity has a technical grounding with which it is implemented. For the majority of activities, i.e. above-mentioned Data Preparation steps, in which value-copying occurs, we have access to the computing instructions in the form of scripts (e.g. Phyton, R, Bean-shell). By using techniques such as static code analysis or programme slicing one could learn these value-copying relations. Similarly using information sources like activity and port names it could be possible to semi-automate the classification of each workflow activity with its motif. This area of investigation is currently left out of the scope of our work.

5. DATA LABELS IN SCIENTIFIC WORKFLOWS

We provide a semi-formal representation for data labels. A label definition is a tuple of the form:

\[ L_{\text{DEF}} = (n, t) \]

\( n \) denotes the name of the label, \( t \) denotes its type. Labels are of sets of primitive types such as “xsd:string” or “xsd:int”. (From an technical grounding perspective a label definition could, in an RDF based implementation [17], correspond to an RDF property definition, label values would be represented with RDF literals which the property points at).

A label vector is a set of distinct label definitions:

\[ L_{\text{DEF}} = \{ L_{\text{DEF}} \} \]

In an analogy to metadata kept regarding files in a file system (e.g. file name, file size, date of creation etc.), when workflows are associated with a given label vector definition, it means that the data artifacts generated from the execution will be auto-curatoried with these labels. Let us assume that the species population modeling investigation, which, amongst others, incorporates the DR workflow, is associated with a label vector of the following form:

\[ L_{\text{DR-WF}} = [L_{\text{origin}}, L_{\text{scope}}, L_{\text{model}}] \]

where \( L_{\text{origin}}, L_{\text{scope}} \) and \( L_{\text{model}} \) are labels defined as follows:

\[ L_{\text{origin}} = \{ \text{"origin"}, \{ \text{"xsd : string"} \} \} \]

\[ L_{\text{scope}} = \{ \text{"scope"}, \{ \text{"xsd : string"} \} \} \]

\[ L_{\text{model}} = \{ \text{"model"}, \{ \text{"xsd : string"} \} \} \]

Label definitions act as a schema for label instances. Actual labels are generated in conformance to the definitions by using the actual data artifacts found in the execution trace of workflows. In Figure 2 we have illustrated label values for data resulting from “gbifOccurrence” and “slwOccurrence” activities.
currence” activities according to the above outlined schema. These label will be generated using the activity definitions and configurations and actual data outputs of the aforementioned data retrieval activities. According to this schema the sample label vectors in Figure 2 contains information on 1) origin that is the service endpoint from which the data is retrieved, 2) scope information, which is the species name input parameter to the retrieval operation, 3) model information, which describe the retrieved data using the Darwin Core vocabulary (a standard vocabulary in Biodiversity) [26]. In our model there is a one to one correspondence between a data artifact and its label vector. So a data artifact can have zero or one label vector, and a label vector can belong to only one data artifact. In scientific data flows, as with the case of Taverna, data is carried around in very basic data structures; namely, collections and single items. We adopt this collection-oriented data structuring approach of data flows. Consequently our framework caters for labels of singletons and of collections.

In the previous section we described Motifs, these annotations are used to specify the labeling behavior expected of each activity in a workflow. As part of the automated cura-tion process that we introduce, there are cases where new labels need to be inferred from existing ones, outside the scope of specific activities. These are the cases of datalinks where data that is outputted by one activity becomes an input to another. The need to infer labels arise in the case of cardinality mismatches between the two ends of a datalink. Consequently in our model, each label \( L \) definition is associated with a tuple of functions \((f_{gen}^L, f_{dis}^L)\), which correspond to generalization and distribution respectively. To illustrate the semantics of these two functions, consider our previous example. Each of the three labels in our example would be associated with function tuples of the form:

\[ \langle \text{“motifs : functions : union”}, \text{“motifs : functions : copy”} \rangle \]

These functions are identified by function URIs. Specifically:

- The generalization function allows us to infer a label for the collection when multiple items are aggregated into a collection. So for the each of the three labels in our example, a union function will be used to infer a label for a collection from the individual labels of items forming the collection.
- The distribution allows the transfer of a collection’s labels to each individual item in the collection. Consequently the association of the above tuple with the three labels in our model would specify that each label of the collection will be copied over to individual items in the collection (should need arise due to cardinality mismatches among data links).

Association of functions with label definitions was a design decision we took to cater for domain specific label propagation and inference capabilities. Instead of functions, we could have opted for fixed operators. As in the case of annotation propagation in databases [4], fixed algebraic operators (such as Union) can identify how to infer labels in cases of generalization. We are aware that this approach would allow for a less complicated label model, but label inference behavior would be fixed in nature. An example of domain specific label inference can be given from Biodiversity. Biodiversity datasets contain usage licenses, this information is often provided within data records themselves. When data is aggregated from multiple sources and providers, the license of the aggregate dataset corresponds to the most restrictive license of the data in the set. Considering license information is represented with a label, such an inference capability can only be achieved with having functions coupled with labels.

We foresee that for a majority cases of metadata attributes a common generic set of functions for inferring new labels will be sufficient. We capture this with support for default functions, which we exemplified in Section 4.1.

6. DATA CURATION PROCESS MODEL
As introduced in our approach, we automate curation of data artifacts through Labeling Workflows. These workflows are underpinned by a Process Model that contain operators for label creation and propagation. Labeling Workflows are not authored directly by users. Instead these workflows are generated/compiled automatically by exploiting 1) the Motif annotations on the scientific workflow description and 2) the Label Model designed to be adopted for the curation. When generating a Labeling Workflow from these two inputs, the generator includes a labeling operator for each annotated data processing activity in the workflow definition, the choice of the operator is informed by the motif annotation of that activity. The generator also includes a labeling operator for the datalinks in the workflow definition, for those which appear to have cardinality mismatches between data structures expected at their two ends. The choice of operator for datalinks is informed by the difference of depths in the data structures expected at the two ends of a workflow data link.

Figure 7 illustrates operators included in the Labeling Workflow in response to activities in the Workflow, and Figure 8 illustrates operators that are included in the Labeling Workflow in response to datalinks in the Scientific Workflow. The vertically drawn inputs to operators are those that are obtained from the annotated scientific workflow description. The horizontally drawn inputs are those that are obtained from the provenance repository, which is also the destination where generated labels are forwarded.

The *Mint* operator underpins the label generation, whereas, *Propagate*, *Generalize* and *Distribute* underpin label propagation. The major difference between generation and propagation is that the former relies solely on the existence of data artifacts within the execution log of a workflow, whereas the latter require the existence of labels to be previously associated with certain data artifacts. Let us visit each operator:

- **The mint operator** generates labels that conform to the label model associated with the scientific workflow. These are domain/activity specific functions that extract metadata from input and output data artifacts (e.g. species name, occurrence details, usage restrictions) of the activity and activity execution configurations (e.g. endpoint, activity type etc). The labeling behavior corresponding to the data retrieval activities in the DR workflow would be represented with the *mint* operator.
- **The propagate operator** corresponds to propagation of the labels of the data artifacts that appear at the designated input port of an activity to the designated output of the activity. The labeling behavior of the activities, “Transform_XML_String”, “=Merge_String_List_To_a_String”, “=Flatten_List”, “Transform_XML”, and “Add_CSV_Headers_To_Occ_Records” correspond to the *propagate* operator.
- **The distribute operator** corresponds to generating labels for individual items from the label of a collection.
- **The generalize operator** corresponds to obtaining a label for the entire collection from the label set of the individual items.

7. LABEL GENERATION WORKFLOWS

In order to illustrate the compilation/generation process of Labeling Workflows in Figure 9 we provide a fragment of the DR Workflow on the left hand side, and we provide the corresponding fragment of the Labeling workflow on the right hand side. The inputs values to each label operator in the labeling workflow are static configuration parameter. The variable part of the inputs (not depicted for brevity) are the actual data artifacts, data labels and the provenance information that are looked up from the relevant repositories. This parameterized lookup capability allows us to execute the Labeling workflow over different execution traces of the same workflow and be able to generate labels for each.

As the communication of data is achieved through a repository lookup process and as the label propagation operators depend on labels of source ports to be able to infer labels of sink ports, there are simple control flow dependencies (run-after type of dependencies) exist between them. Please note
that mint operators do not depend on labels, therefore the compilation/generation could result in multiple small Labeling Workflowlets. The algorithm for this generation process is mainly based on a traversal of the Scientific Workflow definition and generating the labeling workflow by picking up the corresponding operators depending on the motifs of activities or the cardinality statuses of datalinks.

8. RELATED WORK

Semantic annotation of workflows and propagation of these annotations to actual data artifacts generated during workflow runs has been proposed by authors in [20]. Here, annotation occurs at the design time and consequently annotations do not exploit information in the values of data artifacts (i.e. they are static such as a set values selected from a domain vocabulary). Therefore annotations in these approaches tend to describe the general nature/characteristics of data (e.g. it is a query string or occurrence record). Annotated data traces allow for querying lineage paths by using domain ontology terms. These works adopt a black-box view of activities and they do not propagate annotations among data artifacts.

Another closely related field is the study of provenance of database queries. Research in this area is categorized [9] as “Why”, “How” and “Where” provenance focusing (respectively) on 1) tracking which source tuple(s) cause a particular record to appear in a query result, 2) through which operations are the source tuples combined and 3) from which source cells are the data values copied to the result. Where-Provenance is particularly relevant to our work as it tracks value-copying. Where provenance, has been applied to annotation propagation in relational data integration systems. DBNotes [4] is one such system that propagates annotations on source cells to results of Select-Project-Join-Union queries. Where-provenance is sensitive to query re-writes, consequently, DBNotes provides the option to do annotation propagation by computing all equivalent formulations of a query and propagating annotations from cells addressed by all equivalent queries. DBNotes performs a simple accumulation of all annotations of source cell values and does not provide an algebra for annotations themselves, but it does provide a storage scheme for annotations and means to query them alongside data. Polygen [25] is another system for querying multiple databases and aims to track a very particular kind of annotation, which is the designator of the Source Database that a value comes from. Polygen outlines a set of operational rules for propagating annotations through relational operators.

Why and How provenance approaches of databases have recently been applied to dataflows with white-box activities corresponding to query operators [16] or PigLatin programs [2]. Such fine-grained tracking of provenance finds particular usage in workflow debugging, or change impact analysis. In the context of where-provenance for dataflows in [5] authors adopt a logic-based approach to propagation of schema-level semantic annotations through relational query based activities. Rules for propagation of annotations through each relational operator is represented as a logic constraint. A query is represented as a tree of operators, consequently propagation is cast as an application of logical inference with a forward or backward read of the operator tree. Authors speculate that such an approach can find applicability in semi-automated annotation of workflows. Another work from the same authors [6], propose the use of declarative rules for inferring certain classes of dependencies among data artifacts in the execution trace of a workflow. Among the class of dependencies are value and id dependency, which correspond to value or identifier copying from the input of an activity to the output. Similar to our approach they expect rules to be specified on top of workflow descriptions, and later fired over the execution traces to generate dependencies among actual data artifacts. Unlike all other reviewed works, authors have not identified how the resulting dependencies will be utilized.

The work of [19] shall be mentioned here as it has been influential in our work in terms of methodology, specifically for the choice of having a process model for automating the curation. In this work authors describe a scientific workflow re-writing approach, where the workflow is re-written to embed it into a data quality view, which computes quality annotations and filters data based on those annotations.MetaData propagation is also explored in digital library research. Based on concerns that are similar to ours outlined in the introduction, in [14] authors attempt to ease curation of shared research work products through propagation of basic metadata, such as authorship, subject, or publication date. Propagation is from the research articles to their supplementary material (such as data artifacts, visualizations, charts). The authors acknowledge that propagation of metadata may result in incorrect annotations (e.g. not all charts of a paper may have been authored by the same person), which can be edited during a manual curation step.

9. CONCLUSION

The sharing and re-use of scientific datasets has various benefits to data-intensive science, e.g., acceleration of investigations, improved transparency and reproducibility. These benefits come at a cost. In particular, the curation effort expected from scientists prior to publishing data resulting from their analysis. Results of scientific analysis implemented as workflows also require curation. Even though workflow engines collect extensive provenance metadata during the execution this information is not fit for the metadata needs of data publishing. However, by adopting a systematic and structured approach to the analysis process, we observe that
workflow based analysis have a big advantage over ad-hoc analyses and bring-about a substrate on which a metadata generation and propagation framework can be weaved.

In this paper we proposed an architecture for assisting in the curation of data artifacts generated through workflow-based analyses. We proposed a model of Labels for carrying metadata, and a process model based on label generation and propagation operators. The process model formally underpins Labeling Workflows which are generated from scientific workflow definitions with markup denoting the Motif of each activity. Motifs characterize data processing, which allows for inferring associated label/metadata processing operators in the Labeling workflow.

The development of the proposed architecture is on-going at the moment. Upon completion of the algorithm for generating the Labeling workflow, we intend to investigate how much these Labeling Workflows lend themselves to concurrent execution. Evaluations with both a synthetic dataset, and an empirical dataset will be performed. Synthetic tests will be used to demonstrate the practicality of the execution of Labeling Workflows. Real-life workflows will be used to understand how much coverage does the proposed model have in real life workflows.

Acknowledgment

The authors would like to thank Francisco Quevedo from the BioVEL project for his support on the DR workflow (BioVEL: EU FP7 283359 BioVel BioDiversity eLaboratory). This work was supported by the myGrid Platform Grant (EPSRC EP/G026238/1, “myGrid: A platform for e-Biology Renewal”).

10. REFERENCES

[8] V. Chavan. Recommended practices for citation of data published through the GBIF network. (May), 2012.