A Data Model for Analyzing User Collaborations in Workflow-Driven e-Science

Ilkay Altintas[*] and Manish K. Anand[*]
University of California, San Diego, CA, USA

Trung N. Vuong[*]
University of California, San Diego, CA, USA
United States Military Academy (USMA), West Point, New York USA

Shawn Bowers[†]
Gonzaga University, Spokane, WA, USA

Bertram Ludäscher[‡]
University of California at Davis, Davis, CA USA

Peter M. A. Sloot[§]
University of Amsterdam, Amsterdam, The NETHERLANDS
National Research University ITMO, St. Petersburg, RUSSIA
Nanyang Technological University, SINGAPORE

Abstract

Scientific discoveries are often the result of methodical execution of many interrelated scientific workflows, where workflows and datasets published by one set of users can be used by other users to perform subsequent analyses, leading to implicit or explicit collaboration. In this paper, we describe a data model for "collaborative provenance" that extends common workflow provenance models by introducing attributes for characterizing the nature of user collaborations as well as their strength (or weight). In addition, through the implementation of a real-world bioinformatics use case scenario and an associated collaborative provenance database, we demonstrate and evaluate the effectiveness of our model in understanding and analyzing user collaboration in scientific discoveries driven by scientific workflows.

Key Words: Collaborative e-Science, user collaborations, scientific workflow systems, provenance, workflow runs, data publication, querying.

1 Introduction

Today, scientists need to collaborate more than ever. Due to the increasing number and sophistication of data acquisition technologies, the amount of raw data acquired has vastly increased over the last two decades [9]. This explosion of scientific data and knowledge along with the increased number of scientific studies that require access to knowledge from multiple scientific disciplines amplify the complexity of scientific problems, often requiring large teams to work together. To address these challenges, scientists use computational, data, and collaborative technologies that are rapidly evolving.

The requirements for these technologies are based on a common goal to enable collaborative studies serving one or more scientific themes or domains through a computational experimental infrastructure, data sharing, publication and preservation, and a common user interface. Community portals [4] and virtual laboratories [34] are popular technologies and platforms that have emerged as a response to these collaborative requirements of science. These environments establish a common infrastructure where community members can access and contribute data, middleware, and computational tools, and launch and manage computations through their user spaces under generic governance rules.

Scientific workflows [16] often represent repeatable patterns of computational activities, typically designed iteratively and run multiple times by one or more users. Provenance tracking [28] is an important feature of scientific workflow systems as it helps track the origin and processing history of scientific data products, and validate experimental processes that were used to derive these scientific products. Thus, information on data collection, data usage, and, especially, the computational outcome of a scientific workflow provides a rich source for conducting similar future scientific studies [17]. Scientific workflow provenance spans workflow design and execution, and includes essential information for recreating the associations between workflow inputs, workflow outputs,

[*] San Diego Supercomputer Center. E-mail: {altintas, mkanand}@sdsc.edu, trung.vuong@usma.edu.
[†] Department of Computer Science. E-mail: bowers@gonzaga.edu.
[‡] UC Davis Genome Center. E-mail: ludaesch@ucdavis.edu.
[§] Computational Science. E-mail: p.m.a.sloot@uva.nl.

workflow definitions, and intermediate data products.

Collaborative processes often involve the design and execution of multiple scientific workflows [10] where different members of a team conceptualize their contribution via workflows and make them available through a common infrastructure. In this case, a scientific discovery is often the result of methodical execution of multiple scientific workflows (executed over many datasets) invoked at different times by one or more users. Collaborative platforms [26] may also provide end-user interfaces that allow workflows to be executed once or multiple times by scientists. Workflows that are executable within collaborative environments may use data from external data sources, where the scientific outputs are subsequently saved within local or global repositories. Some systems, e.g., [4], also allow intermediate results to be saved within data archives together with corresponding provenance information.

Collaboration is further supported by environments such as myExperiment [19] in which workflows developed using different workflow systems (as well as additional scientific resources) can be published and referenced through the myExperiment web site. These workflows become available to scientists for reuse as well as for creating more complex scientific experiments involving multiple workflows, thereby increasing the potential for collaboration across one or more scientific communities.

Collaborative eScience platforms targeting specific scientific communities are increasingly becoming popular, e.g., within disciplines such as geoinformatics [32] and metagenomics [4]. A typical set of components for an eScience platform is illustrated in Figure 1. In this paper, we focus on extending these platforms with collaborative provenance, i.e., views over underlying provenance information that highlights user collaboration and that is driven by publication and execution of scientific workflows and their use of shared data objects. As shown in Figure 1, we assume an environment that provides system components to establish user spaces, project spaces, access to multiple workflow engines, shared data and provenance stores, and shared workflow repositories. These collaborative platforms also encapsulate user actions, e.g., workflow sharing, workflow execution, data publishing, and workflow run provenance sharing. As a result of the user actions and interactions with the system, knowledge concerning user collaborations accumulates. This knowledge on users' actions can be analyzed to infer implicit user collaborations based on system observables for data publishing, workflow publishing, and workflow runs. Our recent paper on understanding collaborative studies through interoperable workflow provenance described a simple collaboration model [1]. [2] provided an extended definition for the notion of "collaborative provenance", which establishes attributes for characterizing the nature of user collaborations, the strength of collaborations, and for determining "self" collaboration.

Contributions. This paper describes a data model and its implementation to capture and query collaborative provenance. This model extends our earlier work [1] by supporting attributes for determining the *nature* (or type) and *strength* (or weight) of collaboration between multiple users and analysis of a researcher's independent work (i.e., their "*self*" collaborations). We show how our data model is effective for answering both standard provenance queries as well as queries over the collaborative provenance attributes for determining the nature of collaborations, their strength, and for finding "*self*" relationships. Furthermore, through the implementation of a real-world bioinformatics usecase scenario and its associated collaborative provenance database, we demonstrate and evaluate how the application of the proposed model will lead to development of systems that will enable us to effectively understand and analyze users collaborations in scientific discoveries driven by scientific workflows.

The rest of this paper is organized as follows. In Section 2, we describe a bioinformatics usecase for metagenomics, illustrate the collaborative scenario, and give example queries that require analysis of user collaborations. In Section 3, we present a model for collaborative provenance and associated provenance views. In Section 4, we describe a schema for storing collaborative provenance information and show how the views of Section 3 and queries of Section 2 can be expressed using this schema. In Section 5, we explain the implementation of our data model schema in PostgreSQL

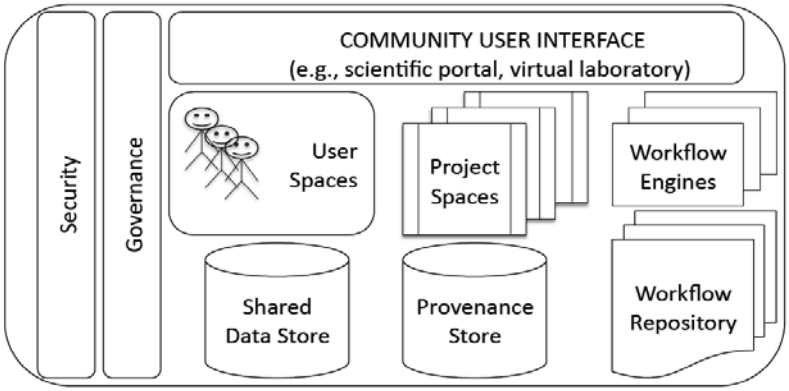

Figure 1: Components in a collaborative eScience project (not shown are the external data, service and computational infrastructure.)

based on an existing workflow execution provenance dataset, and provide an initial evaluation of the performance of collaborative queries. Section 6 describes the relationships between the collaborative model and OPM [25]. We discuss other related work in Section 7 and conclude in Section 8.

2 Collaborative Metagenomics in Camera

CAMERA[1] (Community Cyberinfrastructure for Advance Marine Microbial Ecology Research and Analysis) [30] is a collaborative eScience platform wherein scientific workflows [4] enable the use of various community tools that are shared by a metagenomics [13] research community. CAMERA enables the microbial ecology community to manage the challenges of metagenomics analysis, and has more than 3000 users in over 75 countries worldwide.

2.1 Using Scientific Workflows in CAMERA

CAMERA provides a component-based infrastructure that includes the Kepler scientific workflow system [21]. The Kepler scientific workflows used within CAMERA support the interaction of automated computational tools and human inspection and interaction. Kepler is also used to record the provenance of data products stored within the central CAMERA data repository that were produced through CAMERA workflows. CAMERA also enables users to create, share, and execute workflows specific to their own experiments. Currently, the core CAMERA workflows make the following metagenomic analyses available to researchers: data quality control (specifically, QC Filter and 454 Duplicate Clustering), read assembly (454 Read Assembly), functional annotation and clustering (Metagenomic Data Annotation and Clustering), taxonomy binning (Taxonomy Binning), BLAST, and additional downstream analysis methods. The scientific goals and technical details for these workflows are explained in [4].

Figure 2 shows an example scenario with different observables of shared data, workflows, and workflow runs in CAMERA, where all or part of the output of workflow runs can be used as input to subsequent runs. Figure 2a shows that datasets d_1, d_2, d_5 are published by users u_2, u_3, and u_5, respectively, in the shared data store. Similarly, Figure 2b shows workflows {*QCF, Asbly*}, {*Taxon, Annot}*, and *Comp* being published by users u_2, u_4, and u_5, respectively, in the workflow repository. A critical aspect of the CAMERA workflow environment is that these workflows can be organized into a systematic network (or *combined workflow*), in which outputs of one workflow execution can be used as inputs for subsequent workflows, as show in Figure 3.

This allows researchers to build a complete end-to-end analysis stream by choosing to use different combinations of workflows based on their specific data and analysis needs. For instance, one possible end-to end analysis stream (see Figure 3) for researchers with raw sequencing data may entail: (1) use of the QC filter for data quality control (*QCF*); (2) assembly of the resultant reads to longer contigs (*Asbly*); (3) assignment of taxonomic information to each contig (*Taxon*); (4) annotation of genes against COG, Pfam, TIGRFAM, and other reference databases and clustering of genes to the desired level (*Annot*); and (5) execution of a statistical comparison, obtaining a comparison graph (*Comp*); and so on. The first four steps of this workflow execution scenario are illustrated in Figure 2c, where a run node identifies the provenance of a previous workflow and the data dependencies between inputs and outputs of workflow execution are shown by dashed arcs between data nodes. One can identify the flow of workflow executions leading to a data artifact that is published as a "scientific discovery" by chaining together the interrelated runs (where outputs of runs can be used as inputs to other runs). The provenance information related to all these activities is captured in a common provenance store (see Figure 1).

In Figure 2c, user u_1 performs a run r_1 of workflow *QCF* with parameter settings p_1 and input datasets d_1 and d_2. Run r_1 produces data d_3 as its output, and dependencies between the output and inputs of run r_1 are shown using a dashed arc. Similarly, user u_2 performs a run r_2 of workflow *Asbly* with parameter setting p_1, and input dataset d_3. Run r_2 produces data d_4 as its output, and dependencies between the output and input of run r_2 are shown with a dashed arc. User u_3 performs a run r_3 of workflow *Taxon* with parameter setting p_1, using the output data d_4 from run r_2. User u_1 also performs run r_4, using data from a previous run, i.e., d_4 from r_2, along with other published data d_5, and produces d_7 as its output. In run r_4, d_7, depends on d_4 and d_5. Although not shown in this figure, in general, the output of a run may depend on some but not necessarily all inputs of a run [5]. The statistical comparison step, i.e., the *comp* workflow contributed by user u_4 in Figure 2b, merges and compares the outputs of the *Taxon* and *Annot* workflow runs and has been left out of the current scenario for simplicity.

2.2 Example Queries

Once the basic observables in Figure 2 are captured within a collaborative project like CAMERA, it becomes possible to answer collaborative queries, e.g.:

1) Which data artifacts were used directly or indirectly to generate d_7?
   Answer: {d_1, d_2, d_3, d_4, d_5}
2) Which runs were used in the generation of d_6?
   Answer: {r_1, r_2, r_3}
3) If data artifact d_2 is detected to be faulty, which users should be notified of the error?
   Answer: {u_1, u_2, u_3}
4) What are all the datasets that depended on d_2, in other words, what is the impact of d_2?
   Answer: {d_3, d_4, d_6, d_7}
5) Which users depended on data artifact d_1, directly or indirectly?
   Answer: {u_1, u_2, u_3}

[1] CAMERA Website: http://camera.calit2.net/.

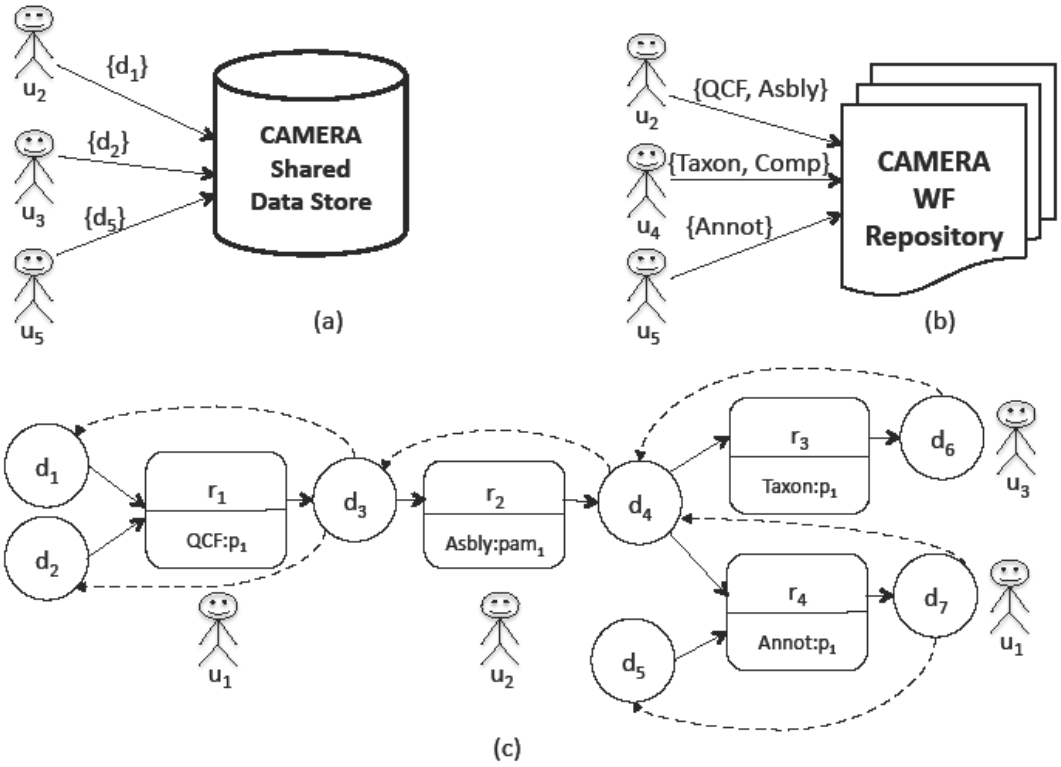

Figure 2: A typical scenario for different observables of shared data, workflows and workflow executions (runs) in CAMERA: (a) data $\{d_1, d_2, d_5\}$ published by users $\{u_2, u_3, u_5\}$; (b) workflows {*QCF*, *Asbly*, *Taxon*, *Annot*, *Comp*} published by users $\{u_2, u_4, u_5\}$; and (c} flow graph for workflow runs (customized through their parameters, p_1) and related provenance data $\{d_1 \ldots d_7\}$} in user space $\{u_1, u_2, u_3\}$. (Note that left to right arrows show run dependencies and right to left arrows show data dependencies.)

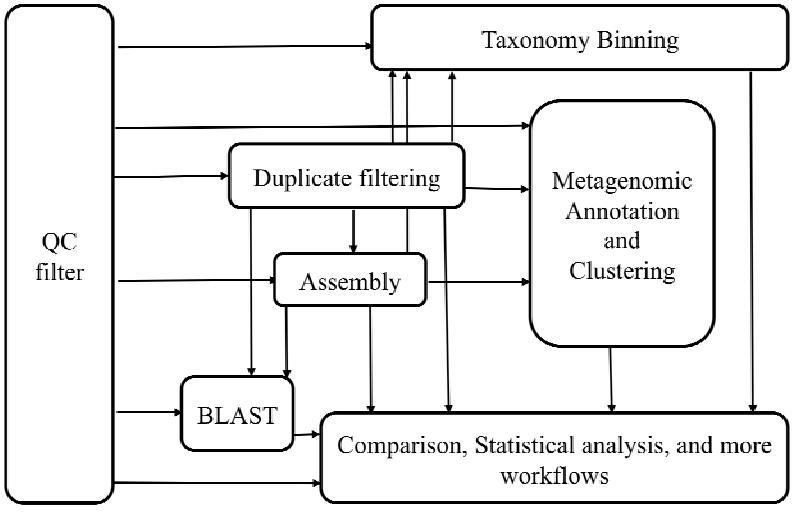

Figure 3: A possible end-to-end analysis stream in CAMERA with raw sequencing data

6) Which users did u_1 depend on, i.e., "collaborate with", directly? What is the nature and strength of each collaboration?
Answer: u_2, u_3 and u_5 via `WF`(1): `Data`(1):`Run`(1), `Data`(1), and `WF`(1):`Data`(1), respectively.

7) Who are the potential acknowledgments for a publication involving d_7, i.e., which user collaborations were involved in the derivation of d_7?
Answer: $\{u_1, u_2, u_3, u_5\}$

The next section further describes our collaborative provenance model and the views associated with the model for determining the various collaboration attributes we consider.

3 Collaborative Provenance Views

Our collaborative provenance model is based on three user actions: (1) users publishing data, (2) users publishing workflows, and (3) users running workflows. In addition, we make the following assumptions:

- All information (data, workflows and information related to workflow runs) is public.
- Data and workflows within the system are globally identified via unique identifiers within the system and/or through a network of cooperating repositories.
- The model of provenance is shared between different workflow systems (either directly or through mappings) and employs data and workflow identifiers. Thus, data produced by one workflow system and consumed by another is uniquely identified through the provenance repository.

3.1 Building Collaborative Provenance Views

Using observables from Figure 2, we can generate views for obtaining data dependencies, run dependencies, and user collaborations, as shown in Figure 4a, Figure 4b, and Figure 4c, respectively. Note that while data and run dependency edges are transitive, user dependencies are not transitive (since the collaboration depends on data and workflow use).

A *data dependency view* shows dependencies between outputs and inputs of workflow runs. These dependencies may span multiple related workflow runs. Figure 4a shows the data dependency view for the workflow execution scenario show in Figure 2c. Here, d_7, which is the result of run r_4, depends on its input dataset d_4 and d_5. Dataset d_4 is the result of run r_2, which depends on the input dataset d_3, directly, and d_1, indirectly through r_1. Combining the entire set of dependencies gives the complete data dependency view, which includes data from multiple related workflow runs.

A *run dependency view* shows dependencies between runs depending on whether the runs used the output of previous runs as their inputs. Figure 4b shows the run dependency view for the workflow execution scenario in Figure 2c. Here, run r_4 used the output data of run r_2 as part of its input dataset resulting in a run dependency view where r_4 depends on r_2.

A *user collaboration view* shows whether users used entities (data, workflows) published by other users during workflow executions. Figure 4c shows the users collaboration view for published data, published workflow, and workflow runs based on the scenario in Figure 2. A collaboration view is created based on a number of underlying collaborative relationships (shown as edge labels in Figure 4c. Figure 5 represents an example showing the different types of collaborative relationships we consider. *A workflow collaboration* (`WF`) is established between two users whenever the first user (e.g. u_a in Figure 5) executes a workflow (wf_x) that is published by a

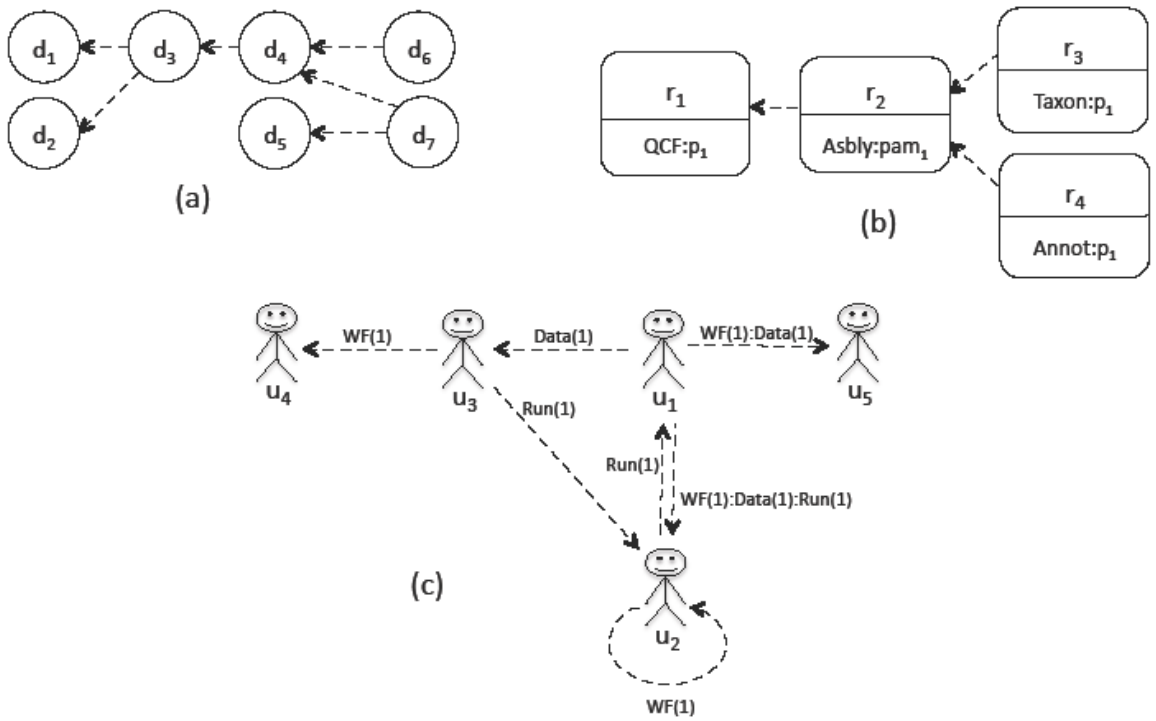

Figure 4: Collaborative provenance views in CAMERA: (a) data dependency; (b) run dependency; and (c) user collaboration (based on Figure 2)

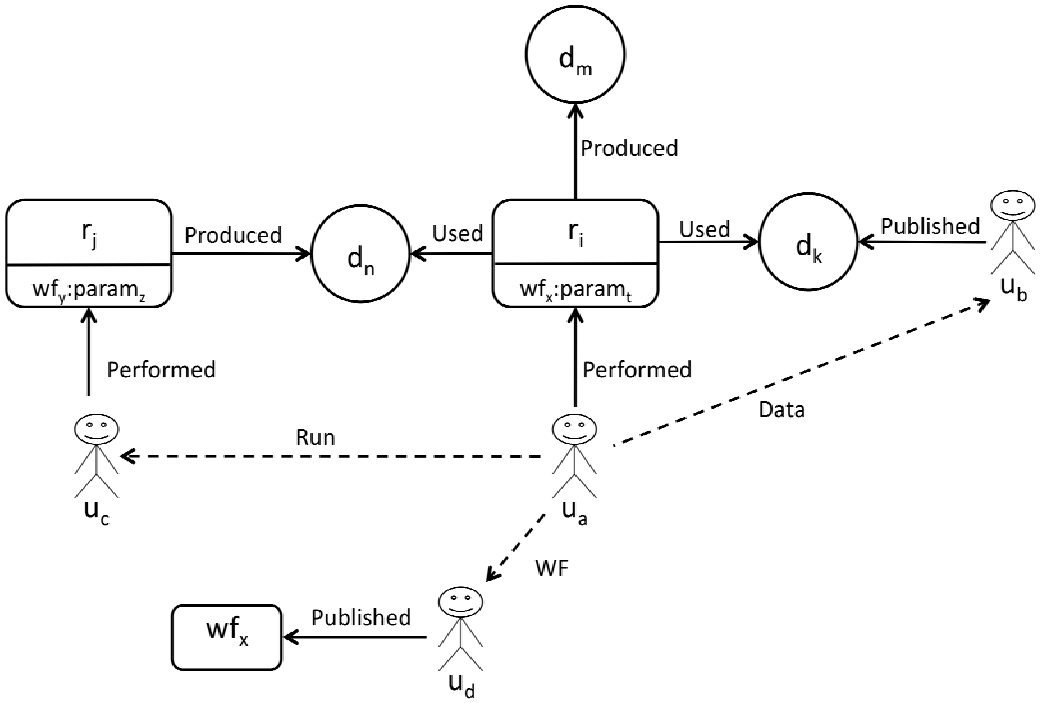

Figure 5: A collaborative provenance model where a user can share collaborations with other users when he performs a workflow execution and uses data and workflows published by other users (collaborations are shown with directed dashed lines).

second user (u_d). A *data collaboration* (`Data`) is established between two users whenever the first user (u_a) performs a run (r_i) in which the input of the run (d_k) is published by the second user (u_b). Finally, a *run collaboration* (`Run`) is established between two users whenever the first user (u_a) performs a run (r_i) in which the input of the run was generated by a run (r_j) performed by a second user (u_c).

3.2 Analyzing User Collaborations

Based on the collaborative provenance model, we introduce three attributes on top of the user collaboration view depicted in Figure 4c:

- **Nature of Collaborations** (C_N): In a user collaboration graph, all the collaborations from one user to another user are reduced to a single directed edge, represented by a dashed arc. This edge may represent multiple collaborations between the same set of users based on the collaboration relationship (i.e., `WF`, `Run`, and `Data` collaborations). We label the collaboration edges to explicitly denote the nature (type) of collaboration labeled `WF`(1): `Data`(1):`Run`(1) to indicate that (i) while performing a run (r_1), u_1 used a workflow (*QCF*) published by u_2; (ii) while performing a run (r_1), u_1 used a dataset (d_1) published by u_2, and (ii) while performing a run (r_4), u_1 used a dataset (d_4) generated by a run (r_2) performed by u_2.
- **Weight Collaboration** (C_W): Each collaboration edge can be assigned a weight that shows the strength of collaboration between the two users. Each collaboration is assigned a value "1" irrespective of the nature of collaboration. Thus, the weight of the dependency between u_x to u_y is proportional to their respective number of collaborations. When combined with C_N, each type of collaboration is weighted separately as shown in Figure 4c. In Figure 4c, the dependency edge between u_1 to u_2 would be reduced to 3 to indicate u_1 established three collaborations with u_2 namely, `WF`(1):`Data`(1):`Run`(1).
- **Self Collaboration** (C_S): A "*self* collaboration" is a special case of collaboration where a user uses a self-published dataset, a self-published workflow, or a data item that was generated by one of her/his previous runs. Self collaboration is by default disabled in a collaborative graph and can be activated to keep track of a users' independent research or to show how often a user made use of their own published workflows or data. Figure 4c shows a self collaboration for u_2 since u_2 used the workflow *Asbly* while performing run r_2.

The following section describes a relational schema for implementing the model presented here, and demonstrates how the schema can be used to answer the queries of Section 2.

4 Modeling Collaborative Provenance

Figure 6 shows the basic model we use for storing the various relationships that exist among users, datasets, workflows, and workflow runs. Data artifacts are assumed to be globally identified via the *Data* class, where data artifacts

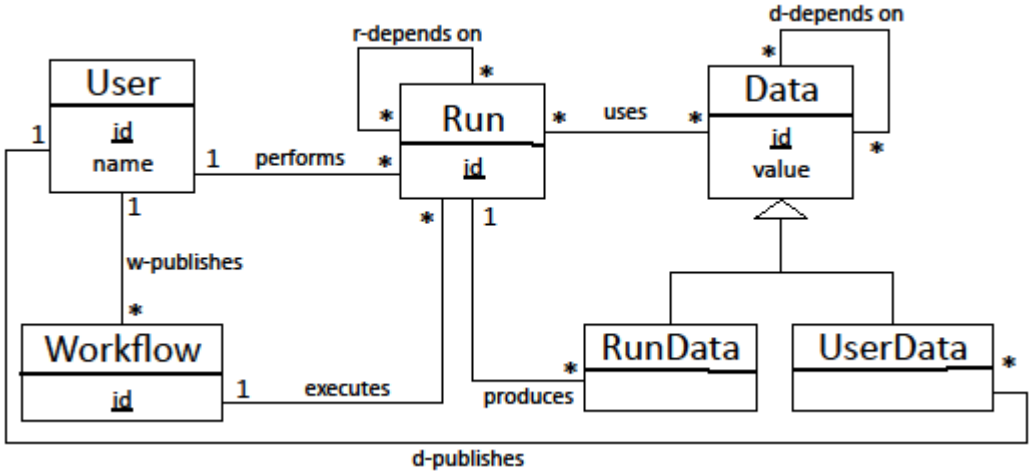

Figure 6: UML-based model for representing collaborative entities and their relationships (key attributes are underlined)

directly published by a user are represented in *UserData* and artifacts generated as a result of a workflow run are represented in *RunData*. Thus, all artifacts represented by the *Data* class represent either *UserData* or *RunData*. As shown, data artifacts can depend on other data artifacts, and similarly runs can depend on other runs.

- a *User* performs one or more *Run*s, publishes (*w-publishes*) one or more workflows, and publishes (*d-publishes*) one or more data,
- a *Run* executes a *Workflow* (where as a workflow can be executed by one or more runs), produces one or more *RunData*, uses one or more *Data*, and depends (*r-depends*) on one or more other runs, and
- a *Data* depends (*d-depends*) on one or more other data (and gets used by one or more runs).

4.1 Collaborative Provenance Schema

We consider the following relational schema based on Figure 6:

- `user`(*u*, *n*) denotes that *u* is a user with name *n*.
- `workflow`(*w*, *u*) denotes that *w* was a workflow published by user *u*.
- `run`(*r*, *w*, *u*) denotes that *r* was a run of workflow *w* and was executed by user *u*, i.e., *u performed r*).
- `data`(*d*, *v*, *t*) denotes that artifact identifier *d* had the value *v* and type *t*, where *t* is either `rdata`(RunData) or `udata`(UserData).
- `publishes`(*d*, *u*) denotes that artifact id *d* was published by user *u*. If `published`(*d*, *u*) then we require that *d* is user data, i.e., that there is a value *v* such that `data` *(d, v, udata)* is true.
- `uses`(*r*, *d*) denotes that artifact *d* was input to run *r*. Note that *d* can be either user or run data.
- `produces`(*r*, *d*) denotes that artifact *d* was output by run *r*. If `produces`(*r*, *d*) then we require that *d* is run data, i.e., that there is a value *v* such that `data`(*d*, *v*, *rdata*) is true.
- `ddep`(d_{to}, d_{from}) denotes a dependency between output *data* d_{from} and input *data* d_{to}.
- `ddep`* (d_{to}, d_{from}) denotes the transitive closure of the `ddep` relation.

Note that above we represent the *publishes* relationship of Figure 6 between a User and a Workflow using the `workflow` relation. Similarly, we represent the *performs* relationship of Figure 6 between a User and a Run using the `run` relation. The *w-publishes* relationship between User and Workflow, and *d-publishes* relationship between User and Data are defined as one to many, but can be extended to many to many without a significant impact to the model in the case of co-authored workflows and data.

The schema captures the explicit dependency between outputs and inputs of a run (denoted *d-depends* on in Figure 6) using the `ddep` relation. Most workflow engines capture this as provenance information. The *r-depends on* relationship is not defined as a base table since it can be derived from `ddep`. We show below how run dependencies can be computed from `ddep`. Typically, this information can be inferred by performing a transitive closure of dependency relations between inputs and outputs of each invocation (i.e., process execution) for a given run. We also perform pre-processing steps to compute the transitive closure of data dependencies and store the result in the `ddep`* relation. This pre-computed transitive closure relation allows faster query execution though it has expensive storage overhead. The exact storage overhead depends on how the workflow and the data dependencies are structured, and can go up to $O(n^2)$, where n is the number of dependencies. Although it does not impact the model discussed in this paper, as future work we intend to use the reduction techniques discussed in [5] for storing both `ddep` and `ddep`* in reduced form. An instance of this schema is show in Figure 7, which corresponds to the example of Figure 2.

$$
\begin{aligned}
\texttt{publishes} &:= \{(\texttt{d}_1, \texttt{u}_2), (\texttt{d}_2, \texttt{u}_3), (\texttt{d}_5, \texttt{u}_5)\} \\
\texttt{workflow} &:= \{(\texttt{QCF}, \texttt{u}_2), (\texttt{Asbly}, \texttt{u}_2), (\texttt{Taxon}, \texttt{u}_4), (\texttt{Comp}, \texttt{u}_4), (\texttt{Annot}, \texttt{u}_5)\} \\
\texttt{run} &:= \{(\texttt{r}_1, \texttt{QCF}, \texttt{u}_1), (\texttt{r}_2, \texttt{Asbly}, \texttt{u}_2), (\texttt{r}_3, \texttt{Taxon}, \texttt{u}_3), (\texttt{r}_4, \texttt{Annot}, \texttt{u}_1)\} \\
\texttt{uses} &:= \{(\texttt{r}_1, \texttt{d}_1), (\texttt{r}_1, \texttt{d}_2), (\texttt{r}_2, \texttt{d}_3), (\texttt{r}_3, \texttt{d}_4), (\texttt{r}_4, \texttt{d}_4), (\texttt{r}_4, \texttt{d}_5)\} \\
\texttt{produces} &:= \{(\texttt{r}_1, \texttt{d}_3), (\texttt{r}_2, \texttt{d}_4), (\texttt{r}_3, \texttt{d}_6), (\texttt{r}_4, \texttt{d}_7)\} \\
\texttt{ddep} &:= \{(\texttt{d}_3, \texttt{d}_1), (\texttt{d}_3, \texttt{d}_2), (\texttt{d}_4, \texttt{d}_3), (\texttt{d}_6, \texttt{d}_4), (\texttt{d}_7, \texttt{d}_4), (\texttt{d}_7, \texttt{d}_5)\} \\
\texttt{ddep}^* &:= \texttt{ddep} \cup \{(\texttt{d}_6, \texttt{d}_3), (\texttt{d}_6, \texttt{d}_2), (\texttt{d}_6, \texttt{d}_1), (\texttt{d}_4, \texttt{d}_2), (\texttt{d}_4, \texttt{d}_1), (\texttt{d}_7, \texttt{d}_3), (\texttt{d}_7, \texttt{d}_2), (\texttt{d}_7, \texttt{d}_1)\}
\end{aligned}
$$

Figure 7: Relation instances of the provenance schema corresponding to the example in Figure 2

4.2 Answering Collaborative Provenance Views

The schema described above can be used to express a number of different provenance views (see Figure 4) using standard relational query languages.

Data Dependency View. We can directly retrieve the data dependency view (DATA-DEP), e.g., shown in Figure 4a, using the ddep relation.

DATA-DEP(d_{to}, d_{from}) :- ddep(d_{to}, d_{from})

Run Dependency View. We can retrieve the run dependency view (RUN-DEP), e.g., as shown in Figure 4b by performing a join between the ddep and produces relations.

RUN-DEP(r_{to}, r_{from}) :- ddep(d_{to}, d_{from}),
produces(r_{from}, d_{from}),
produces(r_{to}, d_{to}).

User Collaboration View. As described in Figure 5, user collaborations can be due to a published workflow (C-WF), published data by users (C-Data), or due to run data being used as inputs (C-Run). The following query can be used to generate user collaboration view, C(u_{to}, e, u_{from}), where e denotes the nature (WF, *Data*, *Run*) of the collaboration:

C-WF(u_{to}, WF, u_{from}) :-run(r, w, u_{from}),
workflow(w, u_{to}).

C-Data(u_{to}, Data, u_{from}) :- run(r, w, u_{from}),
uses(r, d),
published(d, u_{to}).

C-Run(u_{to}, Run, u_{from}) :-run(r_1, w, u_{from}),
uses(r_1, d),
produces (r_2, d),
run(r_2, w, u_{to}).

Note that the union of all of these collaborations C(u_{to}, e, u_{from}) = C-WF ∪ C-Data ∪ C-Run gives the user collaborations with edges labeled according to these attributes, where user u_{from} shares a collaboration of type e with another user u_{to}. User collaboration C also can be used to capture the "*self* collaboration". We can extend C to include the nature and weight of each kinds of collaborations by performing the following operations on C: (1) perform a group by operation over C with concatenation of u_{to}, e and u_{from}, such that 'u_{to}, e, u_{from}' becomes a column; and (2) over the group condition generated in (1), retrieve the number of occurrences (n) for each unique tuple. The number of occurrences of each type can be displayed as $e(n)$ for each kind of edge, and appended with a colon ":" to show the full collaboration label, e.g., WF(1):Data(2). The "group by" and "count" operations are the standard operations in many relational database query languages, e.g., SQL. We denote such a user collaboration graph as C-NWS.

Next, using simple Datalog rules, we show how these views can be used to answer the example queries in Section 2.

5.3 Answering Example Queries

Below we provide the Datalog queries for answering the example collaborative provenance questions posed for Figure 2. We denote query results below via the ans relation.

1) Which data artifacts were used directly or indirectly to generate $\texttt{d}_7$?
   ans(d_{to}) :- ddep*(d_{to}, $\texttt{d}_7$).
2) Which runs were used in the generation of $\texttt{d}_6$?
   ans(r) :- ddep*(d_{to}, $\texttt{d}_6$), produces(r, d_{to}).
3) If data artifact $\texttt{d}_2$ is detected to be faulty, which users should be notified of the error?
   ans(u) :- ddep*($\texttt{d}_2$, d_{from}), produces(r, d_{from}), run(r, w, u).
4) What are all the datasets that depended on $\texttt{d}_2$, in other words, what is the impact of d_2?
   ans(d_{from}) :- ddep*($\texttt{d}_2$, d_{from}).
5) Which users depended on data artifact $\texttt{d}_1$, directly or indirectly?
   ans(u) :- ddep*($\texttt{d}_1$, d_{from}), produces(r, d_{from}), run(r, w, u).
6) Which users did u_1 depend on, i.e., "*collaborate with*", directly? What is the nature and strength of each collaboration?
   ans^{u}_6 (u_{to}) :- C(u_{to}, e, u_1).
   ans^{n}_6 (e) :- C(u_{to}, e, u_1).
   ans^{w}_6 (e, n) := SELECT e, COUNT (e) AS n

FROM C
WHERE $u_{from} = u_1$
GROUP BY e.

Note that we use SQL above to perform the necessary grouping and aggregation for $\texttt{ans}^{w}_{6}$, which gives the number of different kinds of collaborations *e(n)* between users u_1 and u_{to}.

7) Who are the potential acknowledgements for a publication involving $\texttt{d}_{7}$, i.e., which users' collaborations were involved in the derivation of $\texttt{d}_{7}$?
   (7a) ans(u) :- produces(r, $\texttt{d}_{7}$), run(r, w, u).
   (7b) ans(u_2) :- produces(r, $\texttt{d}_{7}$), run(r, w, u_1), workflow(w, u_2).
   (7c) ans(u) :- ddep*(d_{to}, $\texttt{d}_{7}$), produces(r, d_{to}), run(r, w, u).
   (7d) ans(u) :- ddep*(d_{to}, $\texttt{d}_{7}$), publishes(d_{to}, u).
   (7e) ans(u_2) :- ddep*(d_{to}, $\texttt{d}_{7}$), produces(r, d_{to}), run(r, w, u_1), workflow(w, u_2).
   ans: *7a* ∪ *7b* ∪ *7c* ∪ *7d* ∪ *7e*

Note that a data artifact can either be published by a user or be the result of a workflow execution. So, we formulate queries that create a union of these conditions. The above query is explained as: (a) return the user that created the data $\texttt{d}_{7}$ through the execution of a workflow; (b) return the user who published the workflow whose execution created data $\texttt{d}_{7}$; (c) return the users who ran workflows, such that the workflow results shared a transitive dependency (ddep*) relation with data $\texttt{d}_{7}$; (d) return users who published data such that those data shared a transitive dependency relation with data $\texttt{d}_{7}$; and (e) return users who published the workflows such that their execution by other (or same) users resulted in data that shared a transitive dependency relation with data $\texttt{d}_{7}$.

Although we used Datalog as a query language to demonstrate the generation of collaborative views and the example queries in Section 5, we show an evaluation of the collaborative provenance model using a relational database implementation and queries in PostgreSQL. In this implementation, we rely on the capabilities of the PostgreSQL database engine for executing the queries. Please see Appendix A for the SQL equivalents of these Datalog queries that were used in the evaluation. As of now we are not using any query optimization techniques for evaluating these queries, but, as a part of future work, we intend to devise a set of optimization techniques to make collaborative queries faster.

6 Implementation and Evaluation

To validate the data model and queries presented in Section 3 and Section 4, we have implemented a collaborative provenance database in PostgreSQL based on a snapshot of the CAMERA Provenance Database. In this section, we explain the preparation of this database, the reasons for simplification of the CAMERA database, and an evaluation of the performance of the queries.

6.1 Database Implementation

CAMERA Workflows and Provenance Database. For the testing of the collaborative provenance database, we used the existing workflows in CAMERA [4] and the CAMERA Provenance Database [3] associated to runs of the workflows. Currently, CAMERA supports 27 metagenomics workflows, including QC Filter, 454 Duplicate Clustering, different versions of BLAST, Gamma and Alpha Diversity (Rohwer), and RAMMCAP for Metagenomic data annotation and clustering. These workflows take metagenomics sequences (NT, protein, etc.) in one or more (~10) FASTA files, and can handle the reads (processing) of 1 million sequences. As illustrated by Figure 3, the CAMERA workflows are designed to fit together, allowing a user to pick a few of them and create her own methodical scientific process by executing the workflows of interest in the preferred order. To date, the workflows have been executed using from a few thousand to hundreds of thousands of sequences as input over Sun Grid Engine-enabled resources. The provenance for each workflow execution is stored in an Oracle database. Over the past year, around 6,000 workflow executions have been performed in the system, and the size of the provenance information for all workflow executions amount to around 3.7 GB.

With its large user base, diverse set of workflow executions and ever-growing data submissions, CAMERA is an ideal infrastructure for the testing of collaborative provenance model. Through a mapping tool [3] that was built to map workflow data identifiers to global data identifiers in CAMERA, the users can export workflow data (outputs) to the CAMERA database and workflows can exchange data with other workflows. However, the current CAMERA system is not ready for being used as it is for testing the collaborative usecase

Scenarios as discussed here. The current Kepler Provenance Schema[2] in CAMERA does not have complete data to answer the collaborative provenance queries. As mentioned in Section 4, the ddep and ddep* tables should be generated in order to answer collaborative queries in addition to collecting systemlevel information about users, data they published or workflows they ran. Therefore, we based our database implementation on the actual runs and created a synthetic scenario based on these runs. The usecase scenario and the data model discussed in this paper are being used as a basis for the development of a collaborative provenance analysis framework in CAMERA.

Preparation of Collaborative Provenance Experimental Dataset. While Kepler's Provenance Schema keeps track of process-level data dependencies, collaborative provenance model requires dependencies to be captured (or inferred) at the workflow execution level. For instance, Figure 8 illustrates a workflow run with inputs d_1, d_2, and d_3, and the final output d_6.

[2] The latest Kepler provenance schema is explained at: https://code.kepler-project.org/code/kepler/trunk/modules/provenance/docs/provenance.pdf.

Through a set of Kepler API calls, the process-level dependencies can be determined as *{(d_4, d_1), (d_4, d_2), (d_4, d_3), (d_5, d_4), (d_6, d_5)}*. However, the collaborative provenance data model needs a mapping of these process-level dependencies to run-level dependencies, i.e., dependency between initial inputs and final outputs of a workflow (`ddep`). This relationship is described in the ddep table with the two attributes: $data_{to}$, and $data_{from}$. Thus, the `ddep` table associated to the scenario in Figure 8 consists of *{(d_6, d_1), (d_6, d_2), (d_6, d_3)}*.

Figure 9 depicts the tables in the relational schema of the Kepler Provenance Database. Using the CAMERA database

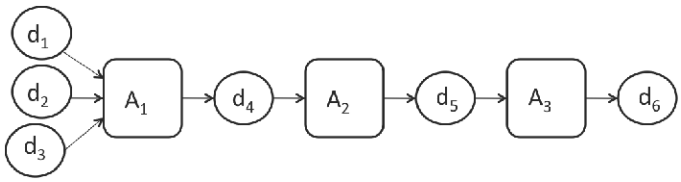

Figure 8: An example workflow execution; A_i illustrates Kepler's processing components (*Actors*), and d_i illustrates data

based on this schema, we retrieve the data for collaborative provenance from four tables, namely, *workflow*, *workflow_exec*, *actor_fire*, and *port_event*. *workflow* and *workflow_exec* provide the *w* for the `workflow`(*w, u*) table and *r* in the run*(r, w, u)* table in the collaborative provenance schema. The remaining two tables (*actor_fire* and *port_event*) provide the direct data dependencies, i.e., data for ddep(d_{to}, d_{from}), in the collaborative provenance schema. The table*actor_fire* records information about actor firings for a particular actor (*actor_id*) in a particular workflow execution (*wf_exec_id*). When a workflow is executed, there are many intermediate inputs and outputs. Multiple components (actors) in the workflow might produce intermediate outputs as intermediate inputs for other components before the final output(s) is produced. The actors that process the data and produce the output(s) for a particular workflow have the same foreign key (*wf_exec_id*) in the *actor_fire* table. However, for populating the collaborative provenance database, we are only interested in the data dependencies between the initial inputs and final outputs of the particular workflow run. In order to retrieve these run-level input and output data dependencies, we need to determine which data among the data processed by

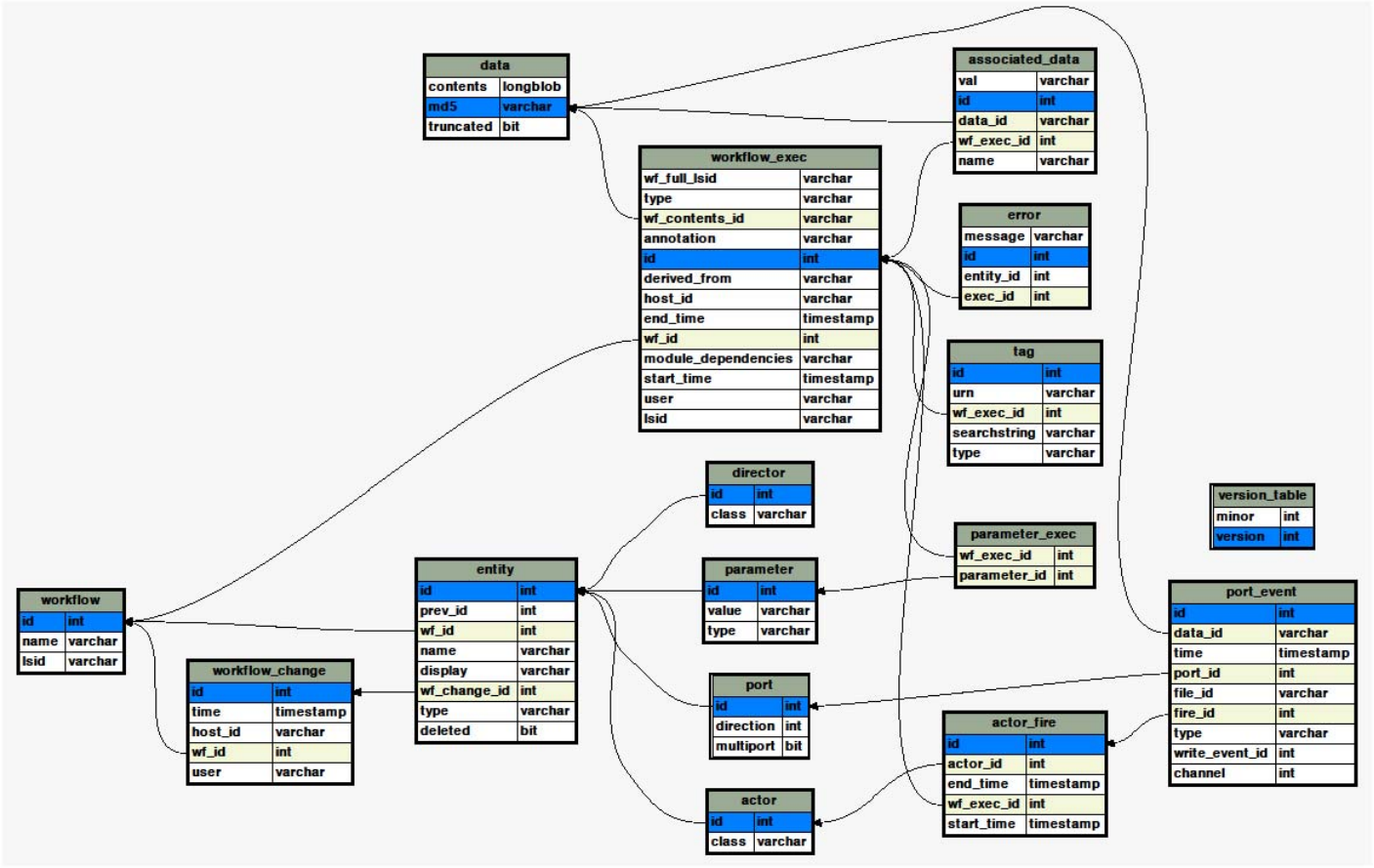

Figure 9: A snapshot of the tables in the Kepler Provenance Database Schema

[3]Actor inputs and outputs are wrapped as tokens in Kepler.

actors are the initial inputs and final output(s). In this case, *port_event* table records the read or write event of a particular actor whenever the actor fires. Each token[3] read or write is stored as a row in this table. A port event occurs at a time, on port *port_id*, and on channel from actor firing *fire_id*. The token's value is referenced by *data_id*. If the data is a file, a reference to the contents of the file is in *file_id*. If the portevent represents a read, *write_event_id* is the *port_event.id* of the port event that generated the token, otherwise (port event is a write) *write_event_id* is -1.

For the transformation of the process-level data into a workflow execution level dataset that conforms to the designed collaborative provenance data model, we implemented scripts to infer run level data dependencies from process level data dependencies. The algorithm we followed in these scripts to retrieve records for ddep table is as follows:

1) Retrieve all the tokens involved in a workflow execution
2) Retrieve all the actors involved in this specific run
3) For each token, determine whether there has been a write event from an actor that processed it. If there has not been a write event, which processed the token, then it is the initial input. In Figure 8, data tokens d_1, d_2, and d_3 do not have an actor that processed a write event to these tokens. Therefore, they are initial inputs.
4) Also for each token, determine whether there has been a read from an actor that processed it. If there is not a read event, which processed the token, then it is the final output. In Figure 8, data token d_6 is the final output since there was no read event that processed it.

In addition, to make the query execution faster, we decided to compute and materialize the transitive closure of `ddep` relation (`ddep`*). Since the Kepler Provenance Schema does not have information about user specific actions through the CAMERA portal (publishing data and workflows, executing workflows), we also created a mapping to capture this information from the CAMERA database.

Implementation. The collaborative provenance schema was implemented as a PostgreSQL database. We retrieved fifty workflow executions from the CAMERA provenance database and determined the run-level data dependencies (`ddep`) for each workflow. After inserting the retrieved data in the `ddep` table, we used the *WITH RECURSIVE* function in PostgreSQL to compute the transitive closure on the `ddep` table and populate the result in `ddep*` table. After we had all the necessary data for the collaborative schema, we expressed and ran all the queries and views in SQL (See Appendix A.) against the PostgreSQL database.

To measure the scalability of the implementation, we gradually increased the datasets by having the run-level dependencies expanding to 10, 25, and 50 workflow runs. Table 1 shows the number of rows in each table of the experimental database for each increment. Note that DB_5 in Table 1 matches the example scenario provided in Figure 2 with an extra workflow run, and the rest of the database is populated similarly to expand the run and corresponding data dependencies.

Table 1: The size of database (in # of tuples) for different data sets

Tables	DB_5	DB_10	DB_25	DB_50
users	5	10	25	50
workflow	5	10	25	50
run	5	10	25	50
data	11	16	31	56
publishes	6	6	6	6
uses	9	14	29	54
produces	5	10	25	50
ddep	12	17	32	57
ddep*	17	52	307	1232

5.2 Evaluation

In this section, we present a short evaluation of the proposed collaborative provenance model and the implemented PostgreSQL database. This evaluation has two primary goals:

1) A validation of the possibility of implementing the proposed data model in correctly answering collaborative queries.
2) An analysis of the changes in the cost of the collaborative views and example queries over an increasing number of run and data dependencies.

We execute the collaborative provenance views, and the example queries on datasets DB_5 through DB_50 to measure the feasibility and effectiveness of our implementation. Table 2 shows the execution times for collaborative provenance views, DATA-DEP, RUN-DEP and USER-COLLAB, along with the execution times for queries Q1 through Q7. The columns represent the query response time in milliseconds for SQL queries over PostgreSQL and the rows represent the datasets that were used to run these queries. Note that the execution time for Q6 and Q7 show the sum of the execution times for each sub-query, specifically, three sub-queries for Q6 and five sub-queries for Q7.

Figure 10a shows the query response time for data dependency (DATA-DEP), run dependency (RUN-DEP), and user collaboration (USER-COLLAB) views on the datasets DB_5 through DB_50 on a linear time scale. Although the execution time for the DATA-DEP view looks like it grows exponentially, Figure 10b on a logarithmic scale shows that all three queries scale linearly. However, the response time clearly shows that the large number of datasets that share dependency relationships, as expected, affects the execution time for DATA-DEP view.

Similarly, Figure 11 is a plot of the query times for example queries Q1 through Q7. Q6 and Q7 take a longer time compared to queries Q1 through Q5 as expected since they are combined (added) cost of multiple sub-queries. Although the linear time scale in Figure 11a looks like the query execution times do not grow too fast, a plot of the data on a logarithmic scale in Figure 11b shows that the execution times for test

Table 2: The query execution time (in ms) for collaborative provenance views and example queries (Q1 to Q7) over different datasets

	DATA-DEP	RUN-DEP	USER-COLLAB	Q1	Q2	Q3	Q4	Q5	Q6	Q7
DB_5	0.121	0.439	1.117	0.12	0.215	0.426	0.091	0.426	1.777	1.215
DB_10	0.252	0.933	1.999	0.146	0.251	0.476	0.104	0.472	1.961	1.544
DB_25	0.981	1.128	2.785	0.254	0.444	0.671	0.189	0.671	2.539	2.090
DB_50	3.872	1.187	2.917	0.542	0.967	0.743	0.483	0.743	3.058	3.748

Figure 10: Query execution time cost for data dependency view, run dependency view and collaboration view in: (a) linear time scale, and (b) logarithmic time scale. 5, 10, 25 and 50 indicate the number of run dependencies in the dataset.

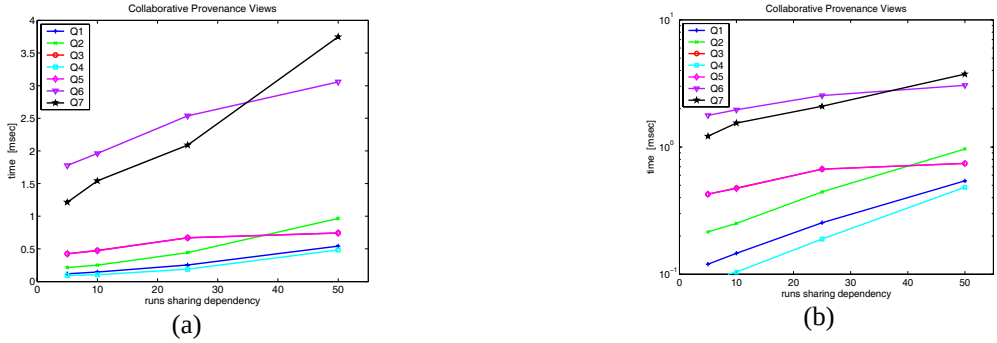

Figure 11: Query execution time cost for evaluation queries 1 through 7 in: (a) linear time scale, and (b) logarithmic time scale. 5, 10, 25 and 50 indicate the number of run dependencies in the dataset

queries grow exponentially with increasing number of data and run dependencies except for the execution times for queries Q5 and Q6.

Analysis. Through this database implementation, we have shown that the model can be implemented over a larger number of runs relative to the example scenario. The model is able to answer the collaborative provenance views and example queries in a reasonable time for our experimental datasets. Although the data dependencies in the dataset were not very complicated, the evaluation demonstrates that as the number of data dependencies increase the queries that rely onthese dependencies take longer. We identified the views and queries that are taking longer time as potential queries to beoptimized, which we endeavor to do as part of our future work. We also plan to expand the evaluation dataset with more dependencies.

Further Evaluation and Collaborative Provenance Browser Development. In addition, a preliminary implementation of an online collaborative provenance browser based on HTML5 and CSS3 is currently under development. The browser provides three different querying interfaces for visualizing data dependencies, run dependencies and user collaborations. We have developed a set of scripts to randomly create an experimental collaborative provenance dataset in MySQL for further evaluation of the data model and initial testing of the browser. The implemented database is queried based on the parameter selections by the users for visualization of collaborative scenarios.

The number of rows per table in the evolving MySQL database currently is as follows: *workflow*: 200, *uses*: 800,

users: 300, *run*: 800, *publishes*: 750, *produces*: 1,200, *data*: 1,950, *ddep*: 1,200, and *ddep_star*: 1,817. We have tested the collaborative queries in this database and observed a performance scaling comparable to our tests in PostgreSQL.

6 Relationship Between the Collaborative Model and OPM

The Open Provenance Model (OPM) [25] has emerged from the e-Science community, and has evolved as a standard representation to facilitate the exchange of information between multiple provenance systems. OPM is based on a model and set of inference rules for directed acyclic provenance graphs, which represent causal dependencies between data products and processes. OPM defines three primary entities (nodes): (1) *Artifacts*: immutable piece of data; (2) *Processes*: actions or series of actions performed on or caused by artifacts; and (3) *Agents*: entities that enable, facilitate, control, or affect execution of processes. OPM also defines five primary types of causal dependencies (edges) that comprise provenance graphs: (1) *used*: a process used artifact(s); (2) *wasGeneratedBy*: an artifact was generated by a process; (3) *wasTriggeredBy*: a process was triggered by another process(es); (4) *wasDerivedFrom*: an artifact was derived from another artifact(s); and (5) *wasControlledBy*: a process was controlled by an agent. In this section, we explain a mapping of the basic OPM entities to the collaborative provenance model and our extensions to the OPM model to represent the relationships for publishing of data and workflows.

As it stands, the OPM specification focuses on the provenance for past executions of workflows. The nodes and dependencies related to past workflow runs in the collaborative provenance model can be mapped one-to-one to the basic OPM model as shown in Table 3. *Artifact* and *Process* nodes in OPM associate to *Data* and *Run* in the collaborative provenance model, respectively. The *used* dependency in OPM is mapped to the *Used* edge between a *Run* and *Data* in our model. Similarly, *Users* in the collaborative model can be viewed as a form of *Agents* in OPM, where *Performed* edges are similar to *wasControlledBy* edges in OPM. *Produced* relationship can be captured by the *wasGeneratedBy* edge in OPM. Note that the directions of the edges for *Performed* and *Produced* relationships change when depicted as *wasControlledBy* and *wasTriggeredBy*. *r-depends on* and *d-depends on* (see Figure 6) relationships can be captured using *wasTriggeredBy* and *wasDerivedFrom*. For example, $\{d_4, r_1, d_1\}$ lineage relation stating that artifact d_1 was *used* by the run r_1 to produce artifact d_4 can be captured as artifact d_4 *wasDerivedFrom* artifact d_1 and artifact d_4 *wasGeneratedBy* workflow run r_1. Adjacent lineage relations, e.g., $\{d_7, r_3, d_4\}$

Table 3 also shows the extensions to the OPM to represent the publishing relationships. A special *Workflow* node was added and is defined as "a specific kind of artifact that refers to the workflow description that is published by the user, and gets executed in one or many processes (workflow runs)". Using a *wasPublishedBy* edge between an agent and workflow or between an agent and data is added to the model. We also capture the relationship between a workflow and a run (process) that executes this workflow explicitly using the *wasExecutedIn* edge. Figure 12 illustrates the nodes and edges that were added to the OPM to complete the mapping to the collaborative model.

Finally, Figure 13 shows the collaborative provenance model in Figure 5 using the defined OPM extensions. *hasWFCollaborationWith*, *hasRunCollaborationWith*, and *hasDataCollaborationWith* in Figure 13 can be inferred using the extended nodes and edges as follows:

Table 3: Mapping of the collaborative entities and relationships to the basic open provenance model nodes and edges

	Collaborative Provenance Model	Open Provenance Model
Nodes	Data	Artifact
	Run	Process
	User	Agent
	Workflow	-
Dependencies	Used	used
	Produced	wasGeneratedBy
	Performed	wasControlledBy
	r-depends on	wasTriggeredBy
	d-depends on	wasDerivedFrom
	wasPublishedBy	-
	wasExecutedIn	-
Inference Rules	hasDataCollaborationWith	-
	hasRunCollaborationWith	-
	hasWFCollaborationWith	-

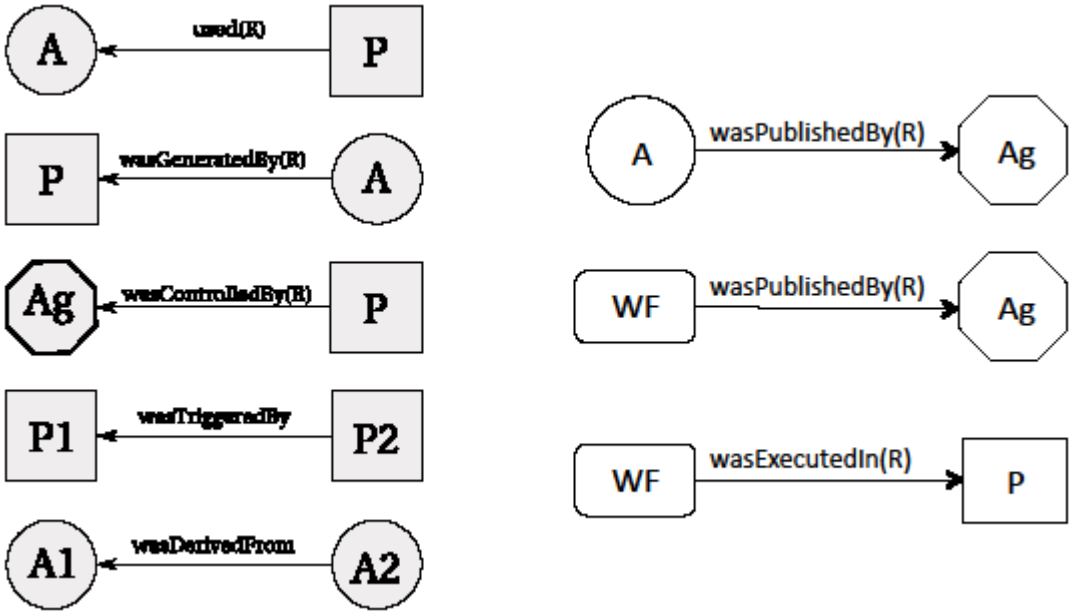

Figure 12: The entities and edges in the standard OPM model are extended by *Workflow (WF)* entity, and *wasPublishedBy* and *wasExecutedIn* edges in the collaborative provenance model

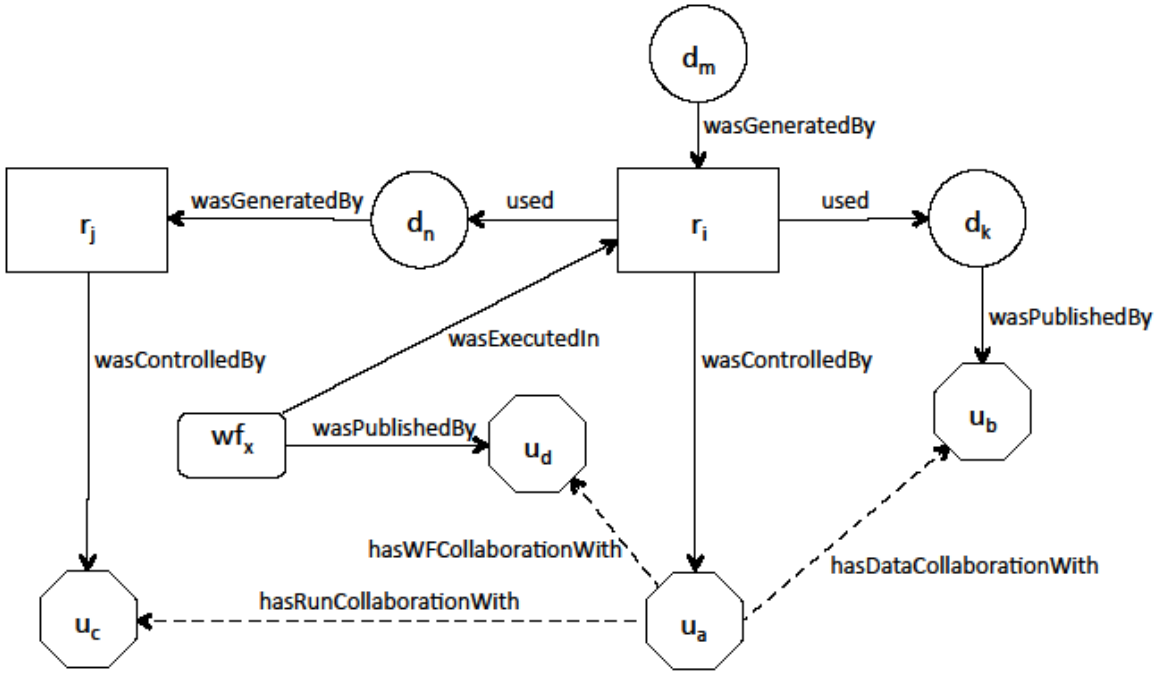

Figure 13: An abstract model of collaborative provenance nodes and dependencies using the extended open provenance model

- If process p_1 *wasControlledBy* agent a_1, and workflow w_1 *wasPublishedBy* agent a_2 and *wasExecutedIn* p_1, then we can infer that a_1 *hasWFCollaborationWith* a_2.
- If process p_1 *wasControlledBy* agent a_1 and *used* artifact A_1 that *wasPublishedBy* agent a_2, then we can infer that a_1 *hasDataCollaborationWith* a_2.

If process p_1 *wasControlledBy* agent a_1 and *used* artifact A_1 that *wasGeneratedBy* process p_2 that *wasControlledBy* agent a_2, then we can infer that a_1 *hasRunCollaborationWith* a_2.

7 Related Work

To help with the understanding of the presented ideas, we review some of the related work in this section. To the best of

our knowledge, none of these projects have collected provenance coming out of different workflow systems, integrated their provenance and used the integrated provenance in collaborative provenance views and scenarios similar to the ones described in this paper. The broad work on scientific workflow development was presented in several special issues [22], [16], books [31], and workshops [18]. In addition, the scientific workflow taxonomy [33] by Yu et al. introduced a general overview and taxonomy of state-of-the-art in scientific workflow development.

Significant current efforts on gathering provenance information targeted at keeping data and the associations of the data products are summarized in recent surveys [28], [17] and a taxonomy of provenance in scientific workflows was presented in [14]. Capturing provenance information in scientific workflows has proven useful in many ways including determining data dependencies, following the steps in workflow design, and even smart reruns and error recovery [24], [15]. In addition, if such provenance information is available, analytical and data mining techniques can be used to learn workflow design methods and rules to assist users in designing similar scientific workflows [17]. The framework described in [10] records associations between multiple related workflow runs. However, our work is based on capturing associations not only across workflow runs, but also across users, where users play an active role of publishing data, or publishing workflows, or executing workflow runs. [20] describes a collaborative scientific workflow design approach where multiple users work on building the same scientific workflow together, and the information on how the workflows were built jointly is recorded. In comparison, in our approach we establish the link across related workflows based on the their execution information (provenance), where outputs of a run might be used as inputs to another subsequent run. In addition, our approach also analyzes the collaboration across participating users across various dimensions of nature, weight, and self-collaboration.

We often see infrastructure projects that make it possible to conduct a number of multi-disciplinary scientific studies for a particular domain, e.g., ViroLab [29], GEON [32], and CAMERA [4]. In the context of CAMERA and GEON projects, scientific workflows run through a GridSphere portal environment. The scientific products used and produced by the workflow are stored in data repositories that are also accessible through the project portal or users local computer. The provenance of workflow execution is stored in a data archive through the workflow execution portlet. However, these environments currently lack queries of collaborative provenance. An interesting opportunity arises from myExperiment [19], where workflow references can be combined through packs, but also any URL referring the data that is associated with the experiment, including other packs. For the collection of Scientific Discourse data, myExperiment follows RDF-encoded models emerging from the Semantic Web community. In the ViroLab virtual laboratory [11], scientific applications are executed as scripts that invoke distributed services wrapped as Grid Objects. Provenance is recorded by collecting events emitted by GridSpace engine that executes the experiment scripts [23]. RDFProv [12] is also a Semantic Web-driven system for storing and querying scientific workflow provenance metadata. RDFProv uses a set of mapping algorithms to map an OWL provenance ontology to RDBMS schema. It also uses translation algorithms to generate SPARQL-to-SQL queries. In contrast, in this paper, we have described a pure relational database schema for the collaborative provenance model, and use SQL to answer the provenance queries. An interesting opportunity arises as a part of our future work where we plan to model our collaborative provenance model by an OWL ontology. An idea is to use the RDFProv system for storing and querying collaborative provenance and comparing the performance of two systems.

8 Conclusion

In this paper, we described a data model to capture and query collaborative provenance. This model extends our earlier work [1] by supporting attributes for the nature and strength of collaboration between multiple users and analysis of a researcher's independent work, namely *nature*, *weight*, and *self*. We also evaluated the implementation of the collaborative provenance data model through a bioinformatics usecase scenario and showed that this data model is effective to answer queries over collaborative provenance attributes for nature, weight, and self, in addition to the standard provenance queries. The developed data model and its implementation serves as a proof-of-concept for the evaluation of the model, and is expected to lead to development of systems that will enable us to effectively understand and analyze users collaborations in scientific discoveries driven by scientific workflows.

Advantages of such a collaborative provenance approach are many since it is all about recording the eScience activity, specifically, capturing the relationships between human users, workflow executions and data. First, it builds upon existing knowledge and extends it without a re-architecting of components in a collaborative eScience platform. Second, it allows for extensions to collaborating entities, e.g., instruments and other system modules, as long as the provenance is kept as system-wide assertions that adhere to a global data model. Most importantly, while minimizing the interrupts to scientists and the way they do their work, it impacts the effectiveness of the collaborative research by adding value by assisting scientific work. This value comes from being able to analyze collaborations and track the footprint of data. Third, this approach brings together consumers and producers of data and other eScience objects together as a start for proper tracking and attribution mechanisms.

Future Directions. As future work, we plan to investigate and apply optimization approaches to the model described here (e.g., adopting the optimizations in [5] and [6]), adopt current provenance querying and visualization techniques (e.g., [7] and [8]), extend the collaborative model with user actions for co-design of workflows and workflow attribution chains, and apply quantitative social network analysis to collaborative

provenance.

Acknowledgements

This research was partly sponsored by NSF SDCI OCI-0722079 for Kepler/CORE, DOE DE-FC02-01ER25486 for SciDAC/SDM, NSF CEO:P DBI-0619060 for REAP, NSF IIS-0630033 for pPOD, NSF AGS-0619139 for COMET, NSF DBI-0753144 for INTEROP, the European INFSOIST-027446 for ViroLab, Gordon and Betty Moore Foundation award to Calit2 at UCSD for CAMERA, and a grant from the 'Leading Scientist Program' of the Government of the Russian Federation, under contract 11.G34.31.0019.

References

[1] I. Altintas, M. K. Anand, D. Crawl, S. Bowers, A. Belloum, P. Missier, B. Ludäscher, C. A. Goble, and P. M. Sloot, "Understanding Collaborative Studies through Interoperable Workflow Provenance," *Provenance and Annotation of Data and Processes (IPAW 2010)*, Troy, NY, LNCS, 6378:42-58, 2010.

[2] I. Altintas, *Collaborative Provenance for Workflow-Driven Science and Engineering*, Ph.D. Thesis, University of Amsterdam, 2011.

[3] I. Altintas, J. Chen, M. Sedova, A. Gupta, S. Sun, A. W. Lin, M. Gujral, M. K. Anand, W. Li, J. S. Grethe and M. Ellisman, "Extending the Data Model for Data-Centric Metagenomics Analysis using Scientific Workflows in Camera," *Proceedings of HPC for Life Sciences Workshop at the Sixth IEEE International Conference on eScience 2010*, Brisbane, Australia, pp. 49-56, 2010.

[4] I. Altintas, A. W. Lin, J. Chen, C. Churas, M. Gujral, S. Sun, W. Li, R. Manansala, M. Sedova, J. S. Grethe, and M. Ellisman, "Camera 2.0: A Data-Centric Metagenomics Community Infrastructure Driven by Scientific Workflows," *Proceeding of the IEEE 2010 4th International Workshop on Scientific Workflows at the 6th World Congress on Services*, Miami, Florida, pp. 352-359, 2010.

[5] M. K. Anand, S. Bowers, T. McPhillips, and B. Ludäscher, "Efficient Provenance Storage over Nested Data Collections," EDBT '09: *Proceedings of the 12th International Conference on Extending Database Technology*, ACM, 2009, Saint Petersburg, Russia, pp. 958-969, 2009.

[6] M. K. Anand, S. Bowers, T. McPhillips, and B. Ludäscher, "Exploring Scientific Workflow Provenance using Hybrid Queries over Nested Data and Lineage Graphs," SSDBM 2009, *Proceedings of the 21st International Conference on Scientific and Statistical Database Management*, Springer-Verlag, New Orleans, LA, USA, pp. 237-254, 2009.

[7] M. K. Anand, S. Bowers, and B. Ludäscher, "Techniques for Efficiently Querying Scientific Workflow Provenance Graphs," EDBT '10: *Proceedings of the 13th International Conference on Extending Database Technology*, ACM, Lausanne, Switzerland, pp. 287-298, 2010.

[8] M. K. Anand, S. Bowers, and B. Ludäscher, "Provenance Browser: Displaying and Querying Scientific Workflow Provenance Graphs (Demo)," *Proceedings of IEEE International Conference on Data Engineering (ICDE)*, Long Beach, California, pp. 1201-1204, 2010.

[9] F. Berman, "Got data?: A Guide to Data Preservation in the Information Age," *Communications of the ACM*, 51(12):50-56, December 2008.

[10] S. Bowers, T. McPhillips, M. W. Wu, and B. Ludäscher, "Project Histories: Managing Data Provenance Across Collection-Oriented Scientific Workflow Runs," *Data Integration in the Life Sciences*, Springer, Philadelphia, PA, USA, LNCS, 4544:122-138, 2007.

[11] M. T. Bubak, T. Gubala, M. Kasztelnik, M. Malawski, P. Nowakowski, and P. M. Sloot, "Collaborative Virtual Laboratory for e-Health," *Expanding the Knowledge Economy: Issues, Applications, Case Studies, eChallenges e-2007 Conference Proceedings*, P. Cunningham and M. Cunningham, Eds., IOS Press, Amsterdam, The Netherlands, pp. 537-544, 2007.

[12] A. Chebotko, S. Lu, X. Fei, and F. Fotouhi, "RDFProv: A Relational RDF Store for Querying and Managing Scientific Workflow Provenance," *Data & Knowledge Engineering (DKE)*, Elsevier, 69(8):836-865, 2010.

[13] Committee on Metagenomics: Challenges and F. Applications, *The New Science of Metagenomics: Revealing the Secrets of Our Microbial Planet*, The National Academies Press, 2007.

[14] S. M. Serra da Cruz, M. L. M. Campos, and M. Mattoso, "Towards a Taxonomy of Provenance in Scientific Workflow Management Systems," IEEE Congress on Services-I, Los Angeles, CA, USA, pp. 259-266, 2009.

[15] S. B. Davidson and J. Freire, "Provenance and Scientific Workflows: Challenges and Opportunities," *Proceedings of SIGMOD 2008*, ACM, Vancouver, BC, Canada, pp. 1345-1350, 2008.

[16] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-Science: An Overview of Workflow System Features and Capabilities," *Future Generation Computer Systems*, 25(5):528-540, 2009.

[17] J. Freire, D. Koop, E. Santos, and C. T. Silva, "Provenance for Computational Tasks: A Survey," *Computing in Science and Engineering*, 10(3):11-21, 2008.

[18] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, and J. Myers, "Examining the Challenges of Scientific Workflows," *IEEE Computer*, 40(12):24-31, December 2007.

[19] C. A. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, and D. De Roure, "Myexperiment: A Repository and Social Network for the Sharing of Bioinformatics Workflows," *Nucleic Acids Research*, vol. 38, no. doi: 10.1093/nar/gkq429, 2010.

[20] S. Lu and J. Zhang, "Collaborative Scientific

Workflows," IEEE International Conference on Web Services (ICWS), Los Angeles, CA, pp.527-534, 2009.

[21] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. Lee, J. Tao, and Y. Zhao, "Scientific Workflow Management and the Kepler System," Concurrency and Computation: Practice and Experience, Special Issue: *Workflow in Grid Systems*, 18(10):1039-1065, 25 August 2006.

[22] B. Ludäscher and C. Goble, Eds., "Special Section on Scientific Workflows," *ACM SIGMOD Record*, 34:3, 2005.

[23] M. Malawski, T. Bartynski, and M. Bubak, "Invocation of Operations from Scripted-Based Grid Applications," *Future Generation Computer Systems*, 26(1):138-146, 2010.

[24] L. Moreau, Ed., "Provenance and Annotation of Data: International Provenance and Annotation Workshop, IPAW 2006, Chicago, IL, USA, May 3-5, 2006, Revised Selected Papers," *LNCS*, Springer, ISBN-10: 354046302X, vol. 4145, 229 pages, November, 2006.

[25] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. Van den Bussche, "The Open Provenance Model Core Specification (v1.1)," *Future Generation Computer Systems*, ISSN 0167-739X, DOI: 10.1016/j.future.2010.07.005, 27(6):743-756, June 2011.

[26] S. Pettifer, K. Wolstencroft, A. Alper, T. Attwood, A. Coletta, C. Goble, P. Li, P. McDermott, T. Marsh, James Oinn, J. Sinnott, and D. Thorne, "MyGrid and UTOPIA: An Integrated Approach to Enacting and Visualizing in Silico Experiments in the Life Sciences," *Data Integration in the Life Sciences*, Springer, Philadelphia, PA, USA, *LNCS*, 4544:59-70, June 2007.

[27] J. P. Scott, *Social Network Analysis: A Handbook*, 2nd ed. Sage Publications Ltd, March 2000.

[28] Y. L. Simmhan, B. Plale, and D. Gannon, "A Survey of Data Provenance in e-Science," *SIGMOD Record*, 34(3):31-36, 2005.

[29] P. M. Sloot, P. V. Coveney, G. Ertaylan, V. Mueller, C. Boucher, and M. A. Marian, "HIV Decision Support: From Molecule to Man," *Phil. Trans. R. Soc*, 367(10.1098/rsta.2009.0043):2691-2703, 2009.

[30] S. Sun, J. Chen, W. Li, I. Altintas, A. Lin, S. Peltier, K. Stocks, E. E. Allen, M. Ellisman, J. Grethe, and J. Wooley, "Community Cyber Infrastructure for Advanced Microbial Ecology Research and Analysis: The CAMERA Resource," Nucleic Acids Research (2011) 39(suppl 1): D546-D551 first published online November 2, 2010 doi:10.1093/nar/gkq1102.

[31] I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, Eds., *Workflows for e-Science*, Springer, 2007.

[32] C. Youn, C. Baru, K. Bhatia, S. Chandra, K.Lin, A. Memom, G.Memon, and D. Seber, "GEONGrid Portal: Design and Implementations," *Concurrency and Computation: Practice and Experience*, 19(12):1597-1607, 2007.

[33] J. Yu and R. Buyya, "A Taxonomy of Scientific Workflow Systems for Grid Computing," *SIGMOD Rec*., 34(3):44-49, 2005.

[34] Z. Zhao, S. Booms, A. Belloum, C. de Laat, and B. Hertzberger, "Vle-wfbus: A Scientific Workflow Bus for Multi e-Science Domains," International Conference on e-Science and Grid Computing, Amsterdam, The Netherlands, 2006.

Appendix A

Table A: Collaborative provenance views for data dependency, run dependency and user collaboration expressed in PostgreSQL

DATA-DEP	SELECT * FROM ddep star ;
RUN-DEP	SELECT t.run, f.run FROM ddep AS d INNER JOIN produces AS f ON d.data from = f.data JOIN produces AS t ON d.data to = t.data;
USER-COLLAB	CREATE VIEW userCollab AS SELECT r.executed user AS uto, w.id AS e, w.published user AS ufrom FROM workflow AS w INNER JOIN run AS r ON w.id = r.workflow id UNION SELECT r.executed user AS uto, u.data AS e, p.users AS ufrom FROM run as r INNER JOIN uses AS u ON r.id = u.run JOIN publishes AS p ON u.data = p.data UNION SELECT r1.executed user AS uto, u.run AS e, r2.executed user AS ufrom FROM run as r1 INNER JOIN uses as u ON r1.id = u.run JOIN produces as p ON u.data = p.data JOIN run as r2 ON p.run = r2.id;

Table B: Example CAMERA queries Q1 through Q7 expressed in PostgreSQL

Q1	Which data artifacts were used directly or indirectly to generate data with id 194119?	SELECT data from FROM ddep star WHERE data to = 194119;
Q2	Which runs were used in the generation of data with id 183215?	SELECT run FROM ddep star INNER JOIN produces ON data=data from WHERE data to = 183215;
Q3	If data artifact with id 49774 is detected to be faulty, which users should be notified of the error?	SELECT distinct (executed user) FROM ddep star AS d INNER JOIN produces AS p ON d.data to= p.data JOIN run AS r ON p. run = r.id WHERE d.data from = 49774;
Q4	What are all the datasets that depended on data artifact with id 49774, i.e. the "impact" of 49774?	SELECT data to FROM ddep star WHERE data from = 49774;
Q5	Which users depended on data artifact 49775, directly or indirectly?	SELECT distinct(executed user) FROM ddep star INNER JOIN produces ON data to=data JOIN run ON run = id WHERE data from= 49775;
Q6	Which users did user with id 11111 depend on, i.e., "collaborate with", directly? What is the nature and strength of each collaboration?	SELECT uto FROM userCollab WHERE ufrom=11111; SELECT e FROM userCollab WHERE ufrom= 11111; SELECT uto, e, COUNT(e) FROM userCollab WHERE ufrom= 11111 GROUP BY (e,uto);
Q7	Who are the potential acknowledgements for a publication involving data with id 194119, i.e., which user collaborations were involved in the derivation of data with id 194119?	SELECT distinct (executed user) FROM produces INNER JOIN run ON run=id WHERE data = 194119; SELECT distinct (w.published user) FROM produces AS p INNER JOIN run AS r ON p. run = r.id JOIN workflow AS w ON r.workflow id =w. id WHERE data = 194119; SELECT distinct(executed user) FROM ddep star INNER JOIN produces ON data from = data JOIN run ON run = id WHERE data to = 194119; SELECT distinct(users) FROM ddep star INNER JOIN publishes ON data from = data WHERE data to = 194119; SELECT distinct (w.published user) FROM ddep star AS d JOIN produces AS p ON d.data from = p.data JOIN run AS r ON p.run = r.id JOIN workflow AS w ON r.workflow id = w.id WHERE data from = 194119;

Ilkay Altintas is the Director for the Scientific Workflow Automation Technologies Lab at the San Diego Supercomputer Center, UCSD where she also is the Deputy Coordinator for Research. She currently works on different aspects of scientific workflows in collaboration with various cross-disciplinary NSF, DOE and Moore Foundation projects. She is a co-initiator of and an active contributor to the open-source Kepler Scientific Workflow System, and the co-author of publications related to eScience at the intersection of scientific workflows, provenance, distributed computing, bioinformatics, observatory systems, conceptual data querying, and software modeling. Ilkay Altintas holds the BS and MS degrees in Computer Engineering, both from Middle East Technical University in Turkey, and a PhD degree from FNWI, University of Amsterdam in The Netherlands.

Manish Kumar Anand is a research developer at the San Diego Supercomputer Center, University of California, San Diego, USA, working closely with domain scientists in ecology, and bioinformatics. His research interests include scientific workflows, data and workflow provenance, storage and query optimizations, and e-commerce. His research expertise is in efficiently managing large provenance graphs (labeled DAGs), specifically, efficient storage, query, visualization, navigation, and summarization of large provenance graphs. He has also developed application tools to manage large provenance information that are successfully integrated into Kepler Scientific Workflow system. Manish holds a Ph.D. and M.Sc. in Computer Science from University of California, Davis, USA, and a B.Tech in Computer Science and Engineering from International Institute of Information Technology (IIIT), Hyderabad, India.

Trung N. Vuong is the chief of Client Training and Support Branch (CTSB) in the Office of the Dean at the United States Military Academy (USMA), West Point, New York, USA. He and his staff provide information technology support to students and faculty and staff. He also assists in testing and implementing the new technologies to enhance the USMA information technology infrastructure. In addition, he teaches in the Electrical Engineering & Computer Science department at USMA. His research interests include scientific workflows, data and workflow provenance, and network security. Trung holds a Master of Science in Computer Science from the University of California, San Diego, USA, and a Bachelor of Science in Computer Science from Towson University in Maryland, USA.

Shawn Bowers is an Assistant Professor in the Department of Computer Science at Gonzaga University, Spokane, WA. His research interests are in the areas of conceptual data modeling, data integration, and scientific workflows. He is a member of the Kepler Scientific Workflow project, where he has contributed to the design and development of Kepler extensions for managing complex scientific data, capturing and exploring data provenance, and ontology-based approaches for organizing and discovering workflow components. Shawn holds a Ph.D. and a M.Sc. in Computer Science from the OGI School of Science and Engineering at OHSU and a B.Sc. in Computer and Information Science from the University of Oregon. Prior to joining Gonzaga University, he was an Associate Project Scientist at the UC Davis Genome Center and was a Postdoctoral Researcher at the San Diego Supercomputer Center.

Bertram Ludaescher leads the Data and Knowledge Systems (DAKS) group at the Department of Computer Science and the Genome Center at the University of California, Davis. His current research interests include scientific workflows, data management, provenance, and declarative, parallel database languages. He studied computer science at the University of Karlsruhe (now a.k.a. KIT) and received his PhD from the University of Freiburg, Germany. From 1998 to 2004, Dr. Ludaescher was a research scientist at the San Diego Supercomputer Center, UCSD, focusing on scientific data integration and workflow technologies. During this time, he co-founded the open source Kepler project, an initiative emerging from two large, collaborative research projects (NSF-ITR/SEEK and DOE-SciDAC/SDM). Since 2004, he is a Professor of Computer Science at the Department of Computer Science and a faculty member of the Genome Center at UC Davis.

Peter M.A. Sloot studied chemistry and physics, finished his Computational BioPhysics PhD work at the Dutch Cancer institute (NKI) in 1988 and did various postdocs abroad. In 1996 he received the prestigious chair in Computational Physics from the Dutch Physics Society and since 2001 is a full professor in Computational Sciences at the Faculty of Science of the University of Amsterdam, the Netherlands. More: http://staff.science.uva.nl/~sloot/