Collaborative Scientific Workflow Composition as a Service

-An Infrastructure Supporting Collaborative Data Analytics Workflow Design and Management

[1]Jia Zhang, [1]Qihao Bao, [1]Xiaoyi Duan, [2]Shiyong Lu, [1]Lijun Xue, [1]Runyu Shi, [3]Pingbo Tang
[1]Carnegie Mellon University -Silicon Valley, USA
[2]Wayne State University, USA
[3]Arizona State University, USA
{jia.zhang; qihao.bao; xiaoyi.duan; lijun.xue; runyu.shi}@sv.cmu.edu; shiyong@wayne.edu; tangpingbo@asu.edu

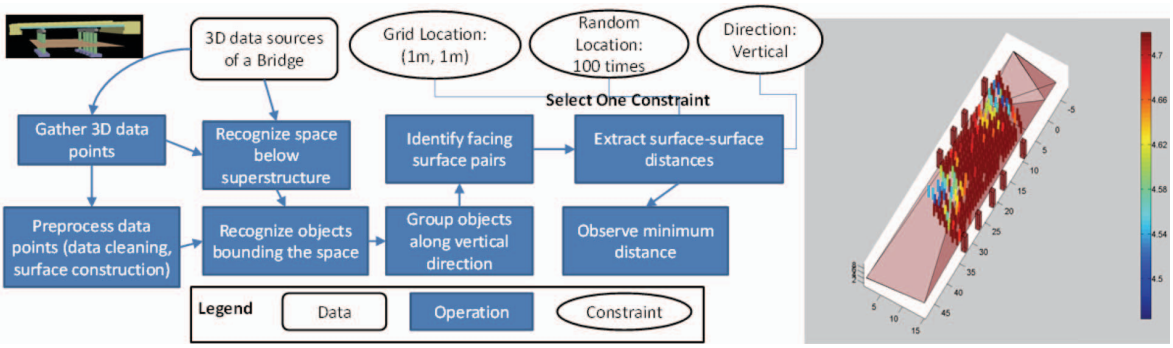

Figure 1. (left): A Workflow for monitoring the safety of a bridge; (right) vertical clearances generated by the workflow (color-coded by distance values).

***Abstract*—The need for collaborative data analytics increases significantly when confronted with the challenges of big data. Although workflow tools offer a formal way to define, automate, and repeat multi-step computational procedures, designing complex data processing workflow requires collaboration from multiple people with complementary expertise. Existing tools are not suitable to support collaborative design of comprehensive workflows. To address such a challenge, this paper reports the design and development of a software infrastructure with the capability of supporting collaborative data-oriented workflow composition and management, adding a key component to existing cyberinfrastructure that will support big data collaboration through the Internet. A collaborative provenance query model (CPM) is presented together with graph-based patterns and algebra. A hypergraph theory-based provenance mining technique is reported. The research extends an existing open-source workflow tool, by adding system-level facilities to support human interaction and cooperation that are essential for an effective and efficient scientific collaboration.**

Keywords—collaborative workflow design; scientific workflow; big data analytics; collaborative provenance

I. INTRODUCTION

Facing the big data challenge, the need for collaborative data analytics increases significantly. Figure 1 shows a highly simplified scenario from modern civil engineering. It aims to continuously monitor and evaluate the height of a bridge under traffic (also known as "under-clearance of a bridge"), one of the National Bridge Inventory (NBI) data items critical for traffic safety, through analyzing streaming 3D sensor data gathered at real time from the bridge under investigation. It comprises a four-step workflow (a formal way to define, automate, and repeat multi-step computational procedures): (1) gather 3D data points collected by various sensors deployed on and around the bridge; (2) preprocess the datasets; (3) reconstruct geometric surfaces of bridge components from the enormous amount of 3D data points; and (4) derive a dimensional model to measure the height of a bridge based on the constructed surfaces. Each step in turn may comprise another sub-workflow. Figure 1 illustrates two sub-workflows: one identifies which surfaces are on the bridge superstructure and which ones are on the highway below the bridge ("recognize objects bounding the space"); the other one samples vertical distances between two clusters of reconstructed surfaces belonging to the superstructure and the highway below, respectively.

This entire data analytics workflow apparently requires a variety of expertise that may not be realized by a single person. For example, Steps 1 and 2 typically require computer science and computer engineering expertise, while Steps 3 and 4 require civil engineering domain expertise. Meanwhile, the individual designs of these data analytics steps have to rely on each other, in accordance with specific accuracy and efficiency requirements of bridge inspection. While engineers with different capabilities focus on finding a local optimal design of a particular data analysis step, collaboration between the teams will find a global optimal design of the entire workflow.

In recent years, a number of dataflow-oriented scientific workflow management systems (SWFMSs) have emerged, represented by VisTrails [1], Taverna [2], and Kepler [3]. However, these systems focus on helping individual scientists and engineers define, automate, and repeat workflows from available applications and services [4]. They are not effective in supporting collaborative design of the aforementioned workflow. First, such tools do not support real-time co-design. Second, locking granularity remains at the level of the entire

978-1-5090-4607-2/16 $31.00 © 2016 IEEE
DOI 10.1109/CIC.2016.37

workflow. One engineer has to wait for the token to manipulate the master workflow. It is difficult to allow two engineers to work on two sections of a workflow and merge their changes later on. Third, it is difficult to track how a collaboratively designed workflow has become as it is. For example, after some time, it is difficult to answer a question such as "why did we adopt this algorithm and who added it." Fourth, it is difficult to capture and retrieve collaboration provenance. Collaboration is knowledge, especially for scientific research. Discussions that lead to a design may be critical for scientific discovery. For example, a question such as "why did we make that change in Step 2" may help justify an important discovery and suggest new research directions. Fifth, scientific workflow design is exploratory in nature. It is unlikely that a scientific workflow is well defined and decided in the beginning. The current workflow tools do not provide system-level support for multiple users to design multiple versions of particular components in a workflow.

To address these challenges, we aim to design and develop a technique supporting *collaborative data-oriented workflow composition*, as a key component toward supporting big data collaboration through the Internet. As the first step, we address one major research challenge of collaborative provenance management. We focus on issues unique to collaborative composition provenance with regard to modeling, gathering, versioning, storing, and querying of workflows. The contributions of the paper are three-fold. First, we have developed a collaborative provenance data model equipped with a graph-level provenance querying formalism. Second, we have developed hypergraph theory-based algorithms for provenance management and mining. Third, we have developed a novel software tool supporting (a)synchronous collaborative scientific workflow design, composition, reproduction, and visualization. Instead of reinventing the wheel, we have extended an existing open-source workflow tool VisTrails as a proof of concept.

The remainder of the paper is organized as follows. In Section II, related work is discussed. In Section III and IV, we present our collaborative provenance query model, and hypergraph theory-based provenance mining techniques, respectively. In Section V, we present a prototyping system and discussions. In Section VI, we draw conclusions.

II. Related Work

The major related works for this project are existing single user-oriented scientific workflow management systems, provenance management, and preliminary collaborative workflow study in the business world.

A. Scientific Workflow Management Systems

Taverna supports limited interaction patterns [5]. The WS-HumanTask model [6] is introduced to integrate humans into service-oriented business workflows. However, none of these task models support comprehensive modeling of collaboration behaviors and patterns required by a scientific workflow task. Existing scientific workflow languages [2, 7] are primarily designed to support automated scientific processes based on Web/Grid services. Human user intervention and interactions are not supported.

Several single user-oriented SWFMSs have been developed over the past few years. Taverna [2] uses an XML-based workflow language for workflow representation. Kepler [3] models a scientific workflow as a collection of actors controlled by a director. VisTrails [1] focuses on workflow visualization supporting provenance tracking of workflow evolution and data product derivation history. Each of these SWFMSs provides a platform to support a single scientist in composing scientific workflows. In our previous work, we have extended Taverna into a multi-user version, focusing on transaction management on shared artifacts [8].

B. Provenance Management

Provenance management has been acknowledged as a critical functionality for any SWFMS; see [9, 10] for surveys. Kepler [11] implements a provenance recorder to track information about workflow runs. Taverna [2] uses Semantic Web technologies to represent provenance metadata. VisTrails [12] records provenance for workflow evolution as well as that of data products [13]. Heinis and Alonso create an interval-based representation for provenance storage [14]. Chapman et al. propose a set of factorization processes and inheritance-based methods to reduce the size of actual provenance datasets [15]. To facilitate focused query and navigation over large amounts of provenance, Biton et al. develop a provenance abstraction technique to return only relevant and abstracted provenance information to a user [16].

Several stand-alone provenance systems have also been developed, including the PReServ system developed under the Provenance Aware Service Oriented Architecture (PASOA) project [17] and the Karma system [18]. To promote the interoperability of provenance among different systems, the Open Provenance Model (OPM) was initiated in 2007 [19]. Recently, PROV [20] framework, endorsed by the World Wide Web Consortium (W3C), formalizes inter-operable interchange of provenance information in heterogeneous environments. A number of emerging applications use either OPM or PROV model for capturing provenance traces [21, 22]. While OPM and PROV only represent retrospective provenance, [23, 24] model prospective provenance. Besides these two types of provenance, evolution information is also important for scientific workflow design and version control in collaborative work, which is what we focus on in this paper.

Provenance information can be extracted from log files generated by SWFMSs [1-3], and then stored into relational database for query [25-27]. To query provenance, Gadelha Jr et al. [28] studied query patterns to simplify query design, Anand et al. [29] proposed a query language to query both lineage and structures on provenance graph but store in relational database. Because queries on evolution provenance focus on relationships, we use graph database (neo4j, https://neo4j.com/) to store both evolution and structure data, and query by the Cypher language (https://neo4j.com/developer/cypher-query-language).

C. Collaborative Workflows

The term "collaborative workflow" is used in the business workflow field to imply collaboration between workflows [30-33]. The business workflow community has started to consider human interaction and collaboration [34-36]. The BPEL4People [34] workflow model is proposed to extend the *de facto* industry

standard business workflow language BPEL [37] to standardize the interaction between automated and human workflows. However, the model is not suitable to be used for supporting collaborative scientific workflows because: it is controlflow-oriented and thus lacks dataflow constructs for interaction, movement, and processing of large datasets.

Aalst [38] proposes the Bill of Materials (BOM) approach to automatically generate a workflow process based on the expected product and its environment. However, the BOM approach is limited to a tree-like structure, where sub-components cannot be shared by different components. Zhang et al. [39] propose to apply pattern knowledge modeling and optimization techniques for workflow generation. In contrast, our approach emphasizes the outcome of a workflow and aims to optimize the performance of generated workflows with correctness guarantee.

Collaboration and versioning have been thoroughly explored in the field of Software Engineering, where the notion of a project represents a tree structure of files. However, scientific exploration is a process that differs from software engineering, as it requires having a frontier of experimental workflow versions, with each version exploring one hypothesis or idea. In contrast, software engineering focuses on maintaining a fewer number of versions, usually aggregating everything into a master version and a handful of temporary new release branches. For example, VisTrails [1] builds a history tree to organize different versions generated by each step of evolution. Furthermore, the issues of data provenance and data ownership are of increasing concern in large-scale scientific collaboration projects. In our paper, we propose collaboration provenance model which support evolution store and prospective provenance store especially for stable versions generated by each "save" operation.

III. Collaborativce Provenance Query Model

A. Movivating Scenario

Recall the civil engineering motivating example in Figure 1. Figure 2 illustrates a simplified collaboration scenario where the two sub-workflows are designed by two scientists s_1 and s_2, respectively. After s_1 designed the first artifact (step of identifying object surfaces bounding the space under the bridge) in version A_1^{v1} and s_2 designed the second step (bridge height derivation) in version A_2^{v1}, the system merged the two steps into a workflow version W^{v1}. However, the resulting workflow did not satisfy specific accuracy requirements. After discussions, s_1 refined A_1^{v1} into A_1^{v2}, and the system generated a new workflow version W^{v2}. Meanwhile, s_2 refined his artifact into a new version A_2^{v2}, and the system generated a final workflow version W^{v3}. Note that in this scenario, s_1 and s_2 designed A_1 and A_2 in parallel and the system is in charge of consistent merging; and discussions (in W^{v1} and W^{v2}) record participating scientists' wisdom.

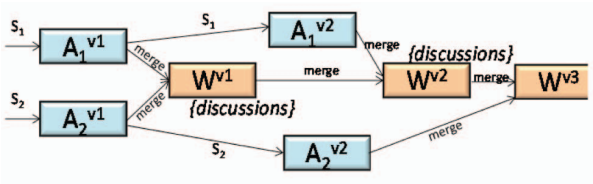

Figure 2 A collaborative workflow composition example scenario.

Based on this motivating scenario, we pose the following six provenance queries that drive the design and development of our project.

Q1: Show all the details about how W^{v3} has been designed and evolved as it is;

Q2: Return all the designers who contributed to the design of W^{v3};

Q3: Return the sub-workflows designed or refined by user s_1;

Q4: Return all user pairs who designed some workflows collaboratively;

Q5: For workflow W^{v3}, which versions of comprising steps 1 and 2 are used? Who designed the two steps? How are they designed or refined? How are they merged?

Q6: What are the previous versions of W^{v3}? Why was it refined?

B. Collaborative Workflow Composition Provenance

We argue that the details of such a collaboration process should be recorded as provenance. In contrast to normal workflow provenance capturing derivation history of data products at run time [16], we propose *collaborative workflow composition* provenance as follows:

Collaborative workflow composition provenance records knowledge allowing participants: 1) to validate a workflow by tracking how a workflow has become as it is from multiple collaborators; 2) to acknowledge credits by recording who has done what at what time; 3) to capture and retrieve collaboration knowledge (annotations and discussions); and 4) to form the basis for merging workflow changes from distributed multiple users.

In order to catch the collaborative workflow composition provenance, we have developed a data model. Instead of reinventing the wheel, we extend the PROV-DM (PROV Data Model, https://www.w3.org/TR/prov-dm) and develop a Collaborative Provenance Model (CPM) as shown in Figure 3. A workflow contains one to many processors (tasks), which in turn may contain sub-workflows. Processors can be connected together by data links through their input ports and output ports. It should be noted that the author information is caught separately, which is associated with every activity over any entity. Each entity also carries evolving history.

C. Graph-Level Querying Formalism and Optimization

We have studied various aspects of collaborative provenance management including modeling, gathering, versioning, storing, querying, and visualization. Most existing workflow systems [29] store provenance data in provenance stores and conduct provenance querying using query languages, such as SQL, SPARQL, and XQuery over their lower-level physical provenance stores (i.e., RDB, RDF, and XML). Such query languages are closely coupled to the underlying provenance storage strategies and therefore suffer from several serious issues. First, users have to know the structures or schemas of such provenance stores and the semantics of provenance models applied to the provenance storages to formulate provenance queries. Second, users require expertise

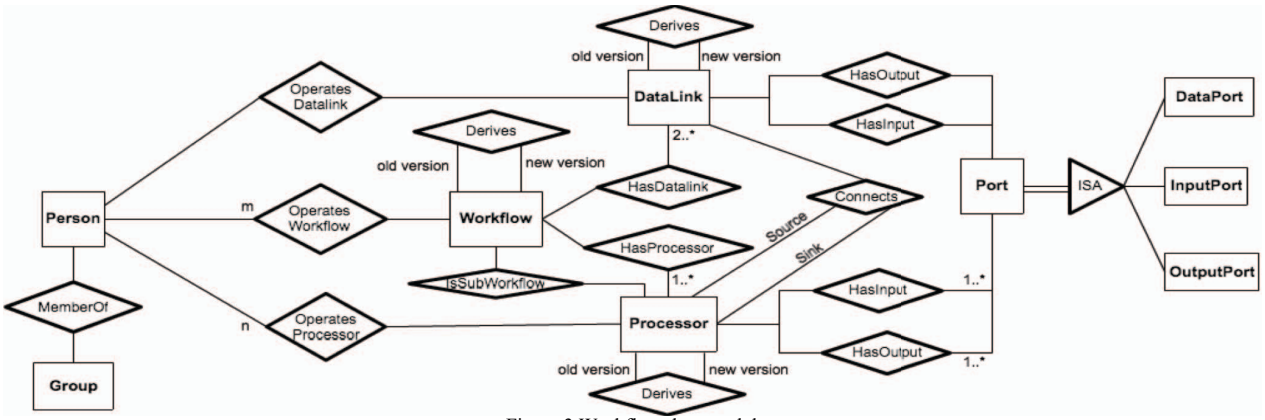

Figure 3 Workflow data model.

about grammars, syntax, and semantics of such languages to formulate complicated provenance queries. For example, using existing approaches, provenance lineage queries (queries for tracking ancestor nodes) often require users to write recursive queries (directly typing recursive statements or using recursive functionality), which obviously is not a trivial task. Third, queries formulated in such systems are fragile and hardly portable. Queries formulated in one system cannot be ported to another even both of them use the same query language due to the differences at the underling schema designs. If physical provenance storage changes, for example as a result of a better design, then existing queries must be rewritten, leading to a painful experience for end users.

To address the aforementioned issues, we have developed a **graph-level** querying formalism and efficient query optimization techniques for managing the lifecycle of collaborative provenance. In contrast to existing approaches, our higher graph-level query formalization will not be tightly coupled to the underlying provenance storage strategies, while featuring the native support for query processing at the provenance graph level.

Our Collaborative Provenance Model (CPM) can be formalized as G= (V, E), where:

- V represents a set of two types of vertices V= EN∪AG, in which EN is a set of entities generated or used, and AG is either one person or a group of collaborators that performs an activity. In the scenario of collaborative workflow design, we define six subtypes extended from entity in PROV-DM: dataport, inputport, outputport, datalink, processor and workflow; and two subtypes extended from Ag: group and person.
- E represents a set of four types of edges E = Ei∪Em ∪Eo∪Ed, in which: i) Ei⊆EN×EN and $(en_1,en_2)\in$Ei states that entity en_1 is included by another entity en_2; ii) Em⊆AG×AG and $(ag_1, ag_2)\in$Em states that designer ag_1 is a member of group ag_2; iii) Eo⊆EN×AG and (en, ag)∈Eg states that entity *en* is operated by agent *ag*, including the operation of adding/editing/deleting (some entity), and merging/saving (workflow); iv) Ed⊆EN×EN and $(en_1,en_2)\in$Ed states that a new version of entity en_1 is derived from an old version of entity en_2.

To describe the evolution of entities and relations among collaborators, we add two new relations that are undefined in PROV-DM, i.e., i) and ii). iii)~vi) are extended from PROV-DM.

Based on CPM formalism, we have developed an XML-based description language called CPML. Following common strategy, we leverage XML-Schema to define the CPML schema, as illustrated in Figure 4. For example, extending the "entity" term defined in PROV-DM, CPM defines *DataPort, InputPort, OutputPort, DataLink, Processor, and Workflow*. The "attribution" term defined in PROV-DM is extended by user operations including *IsAddedBy, IsDeletedBy, IsEditedBy, IsMergedBy,* and *IsSavedBy*. The "agent" term in PROV-DM is extended into individual *Person* and *Group*, to represent collaboration relationships.

CPM CPM [@id, @name, (Add|Delete|Edit|Save)*]
Add add [@timestamp, @user, (Processor|DataPort|InputPort|OutputPort|DataLink)]
Delete delete [@timestamp, @user, (Processor|DataPort|InputPort|OutputPort|DataLink)]
Edit edit [@timestamp, @user, (Processor|DataPort|InputPort|OutputPort|DataLink), IsDerivedBy]
Save save [@timestamp, @user, Workflow, IsDerivedBy]
IsIncludedBy isIncludedBy [@timestamp, @WorkflowId|@ProcessorId|@DataLinkId]
IsDerivedBy isDerivedBy [@timestamp, @WorkflowId|@ProcessorId|@PortId|@DataLinkId]
Workflow workflow [@version, @ProcessorId|@DataLinkId, IsIncludedBy]*]
Processo processor [@id, InputPort*, OutputPort*, Function]
InputPort inputport [@id, @type, IsIncludedBy]
OutputPort outputport [@id, @type, IsIncludedBy]
DataPort dataport [@id, @type]
DataLink datalink [@id, (@InputPortId|@OutputPortId|@dataPortId, IsIncludedBy)*]

Figure 4 CPML specification.

Figure 5 illustrates an automatically generated CPM graph (in neo4j graph database) that represents the scenario of the collaborative workflow design in Figure 2.

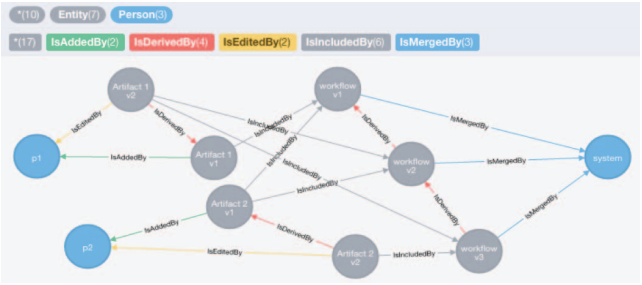

Figure 5 CPM graph for scenario in Figure 2.

D. CPM-based Graph Patterns and Graph Algebra

Based on the CPM graph model, we have developed a graph pattern matching-based querying formalism. More specifically, we formalize a graph pattern P as a triple (M, O, C), where:

- M is a CPM graph.
- O is an inverse-functional one-to-many mapping that returns a set of nodes that have direct causal dependencies associated with a node. O is composed of eight types of mapping functions, corresponding to backward and forward matching patterns over the four kinds of edges defined in CPM) (i.e., O ∈ {Oi, O^i, Om, O^m, Oo, O^o, Od , O^d} as defined by $Oi(en_1) = \{en_2 \mid (en_1,en_2) \in Ei\}$, $O^i(en_2) = \{en_1 \mid (en_1,en_2) \in Ei\}$, $Om(ag_1) = \{ag_2 \mid (ag_1, ag_2) \in Em\}$, $O^m(ag_2) = \{ag_1 \mid (ag_1, ag_2) \in Em\}$, $Oo(en) = \{ag \mid (en, ag) \in Eo\}$, $O^o(ag) = \{en \mid (en, ag) \in Eo\}$, $Od(en_1) = \{en_2 \mid (en_1,en_2) \in Ed\}$, and $O^d(en_2) = \{en_1 \mid (en_1,en_2) \in Ed\}$. Graph patterns are combination of these four types of mapping functions.
- C is a predicate on the properties of the motif.

Based on the graph patterns, we further define the following four operators to manipulate and query a CPM graph: extract, union, intersection and difference. The later three operators enable composition of various graph patterns, allowing querying formulation for collaborative provenance.

1) Extract operator (δ), which extracts a set of nodes or edges from a CPM graph using a graph pattern. An extract operator is defined using a graph pattern P. It takes one CPM graph (G) as input and produces a new CPM graph that matches the graph pattern as output, denoted by $\delta_P(G)$.

2) Union operator (∪), which calculates the union of two CPM subgraphs. A union operation is defined by $G_1 \cup G_2$, resulting in a CPM graph $G' = (V', E')$, where:

$$V' = \{v | v \in V_1 \; or \; v \in V_2\}$$

$$E' = \{e | e \in E_1 \; or \; e \in E_2\}$$

3) Intersection operator (∩), which calculates the intersection of two CPM subgraphs. An intersection operations defined by $G_1 \cap G_2$, resulting in an CPM graph $G' = (V', E')$, where

$$V' = \{v | v \in V_1 \; and \; v \in V_2\}$$

$$E' = \{e | e \in E_1 \; and \; e \in E_2\}$$

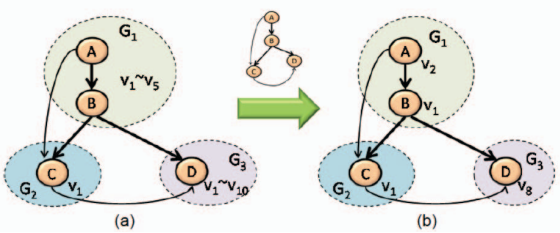

Figure 6 Derive a workflow from versioned artifacts.

4) Difference operator (–), which calculates the difference of two CPM subgraphs. A difference operation is defined by $G_1 - G_2$, resulting in an CPM graph $G' = (V', E')$, where

$$V' = \{v | v \in V_1 \; and \; v \notin V_2\}$$

$$E' = \{e | e \in E_1 \; and \; e \notin E_2\}$$

IV. Hypergraph Theory-Based Provenance Mining

Collected provenance data can be leveraged to support optimized workflow co-design. Figure 6 illustrates how new workflow designs can be automatically derived from our fine-grained versioning provenance. The middle section of Figure 6 illustrates a conceptual workflow template, where each node represents an abstract task described in desired functions. As shown in Figure 6(a), the conceptual design of the workflow comprises three local workflows (sub-workflows), each being designed by individual groups G_1~G_3, respectively. Each local workflow maintains its own version history. For example, local workflow C has one version; and D carries 10 versions. All such historical versioning is caught by our CPM as described in Section III.

Note that versions in the context of workflows are different from traditional versions in the context of software engineering, in which the most recent versions are usually mostly preferred. The exploratory nature of scientific workflows decides that, different versions may be equally important as they may represent different strategies and/or algorithm implementations. For example, assume that the local workflow D represents the step of extracting vertical distances at sampled points in the motivating example described in Figure 1. Group G_3 may have implemented D using 10 different algorithms and stored them in 10 versions in provenance, respectively. In addition, the versioning of comprising components can be different than that of the workflow. For example, components (e.g., tasks) A and B may maintain its own versioning history independent of the local workflow to which they belong, versions 1 to 5.

Workflow designs can be automatically derived from such provenance oriented to different query requirements. Given a specific user query, figure 6(b) shows an instance of workflow design recommendation, while the comprising components A, B, C, and D take corresponding versions v_2, v_1, v_1, and v_8, respectively (local workflow operated by G_1 is presented by v_2). Note that not all combinations of local workflow versions are appropriate to derive a new workflow design. For example, local workflows B and D in Figure 6 may have to be compatible with each other before they can be merged into the global workflow.

A. Hypergraph-based CPM Modeling

In order to realize such automatic workflow derivation and composition from provenance mining, our strategy is to model the above workflow graphs, versioning, and derivation history as hypergraphs. While pairwise relations are naturally captured in a graph, some complex relational objects cannot be straightforwardly represented by an ordinary graph, such as, in our case particularly, the relation between actors and different versions of artifacts. In contrast to normal graph where each edge links between two nodes at two ends, hypergraph as a generalization of normal graph can have edges that connect together more than two nodes.

Intuitively, an execution that is invoked by some actor through certain combination of versions of artifacts can be represented as a hyperedge in a hypergraph, $G = (V, E)$. With each hypergraph e associated with a non-negative number $w(e)$ that acts as the weight of e, we call such hypergraph a weighted hypergraph, $G = (V, E, w)$.

Formally, we formalize CPM data as a hierarchical hypergraph, H(V, E, w, f_c), where:

- V is a set of nodes, representing various versions of an artifact and actors;
- E is a hyperedge set, $e \in E$, $e \subseteq V_1 \times V_2 \times \ldots \times V_k, k \leq |V|$. Each $e \in E$ represents an Execution Package, which contains series of different versions of artifacts that compose a workflow along with the possible actors of the workflow.
- w is a set of non-negative numbers which acts as the weight for each $e \in E$, which represents the influence of each Execution Package.
- fc: V → Bool is a consistency function that given a workflow v∈V, fc(v) will return the consistency value (true or false) of v.

With such hypergraph structure, average commute time distance similarity measure [40] can be applied for discovery of latent associated artifacts and actors. Unlike usual shortest path distance, commute time distance takes into account the connectivity of nodes, so that a pair of nodes strongly connected are more close to each other, compared to weakly connected pair of nodes. The average commute time distance similarity $d(i, j)$ between node i and j is positively correlated to the number of paths connecting these two nodes increases and negatively correlated to the total length of the path decreases.

Here we explain how versioning can be modeled in such a hypergraph. The basic nodes are artifacts, which represent global workflows, local workflows, or tasks. Versions are modeled as hyperedges, that is, each version of an artifact is represented by one node, and all versions of the same artifact are connected by one single hyperedge. Such a modeling strategy provides an indexing technique that enables a user to quickly identify all versions of a particular artifact without navigating a whole potentially large-scale graph. Finally, a workflow derived from multiple versions of artifacts must be consistent. Such consistency property can be model as the property of hypergraphs, leading to a hypergraph-based consistency formalism in which ambiguity is eliminated and correctness is ensured.

Based on a provenance hypergraph, we have applied and developed hypergraph-theory based algorithms and mining strategies to support collaborative workflow composition.

B. Graph Distrance Calculation

Based on our CPM hypergraph structure, we have developed a similarity measurement that describes the relevance of nodes in the hypergraph. Our similarity computation is based on the Average Commute Time Similarity calculation method. We apply the Laplacian matrix follows Liu et al's hypergraph Laplacian equation [40].

We first define three matrixes. The first matrix D is defined as a diagonal matrix. D (vertices) contains all the vertices elements as row and column elements, while the value on the diagonal is the number of edges containing the vertex. Similarly, D (edges) contains all the edge elements as row and column elements, the value on the diagonal is the number of vertexes contained by the edge.

The second matrix M is defined as an incidence matrix, where the row elements are vertices and column elements are edges elements. The value of M is defined as follows:

$$m(v, e) = \begin{cases} 1, & v \in e \\ 0, & otherwise \end{cases}$$

The third matrix W is defined as a diagonal matrix with diagonal value as the weight for the edges. The weight is calculated based on the frequency of the execution.

Based on the three defined matrices, the Laplacian equation is calculated as follows:

$$L = D_{vertices} - M \times W \times {D_{edge}}^{-1} \times \mathrm{M}^T$$

Pseudoinverse is computed as follows:

$$\mathrm{L}^+ = (L - e \times e^T/n)^{-1} + e \times e^T/n$$

Let's define e_i as the *ith* column of I. The similarity is thus calculated using the following formula:

$$S(i, j) = V\left({\mathrm{L}^+}_{ii} + {\mathrm{L}^+}_{jj} - 2{\mathrm{L}^+}_{ij}\right) = V(e_i - e_j)^T \times \mathrm{L}^+ \times (e_i - e_j)$$

V = tr(D) is the volume of the hypergraph, it calculates the sum on diagonal value of D.

The pseudocode of the similarity calculation is summarized in Table I.

Table I Hypergraph similarity calculation algorithm.

Alg. 1. Hypergraph similarity calculation **Input**: A hypergraph graph and the frequency of the execution as weight **Output**: The similarity matrix of hypergraph graph
1. Initialize graph A input 2. **While** 3. Initialize each cell of matrix D,M,W 4. Calculate degree of A to set the D 5. Calculate frequency of the execution to set W 6. Calculate M according to the connection of A 7. **Until** all D, M, W are calculated 8. record D, M, W 9. **Calculate** Laplacian equation L 10.**Calculate** Laplacian equation L^+ 11. **Calculate** Laplacian equation similarity $S(i, j)$

V. Prototyping System

We have developed a prototyping system that realizes our design as a proof of concept.

A. Hypergraph-based Workflow Recommendation

We have implemented the hypergraph-based workflow mining algorithm. Here we explain how such an algorithm can be used for automatic workflow composition, using a simulation over the motivating example shown in Figure 5. The workflow template and related workflow evolution provenance in Figure 5 can be turned into a hypergraph, which denotes a network of execution runs of different versions of artifacts performed by several actors, $\{a_1, a_2, a_3, \dots, a_7\} \epsilon A$, where A denotes the set of actors. Since each workflow has its template where the required artifacts and workflow structures are specified, an actor could choose desired versioned artifacts as components in the workflow. For instance, in our experiment, we designed the workflow template as shown in Figure 5, where artifacts A and B together form a sub-workflow G_1, whose output are considered to be input of sub-workflow $G_2 and\ G_3$, which consist of artifacts C and artifact D accordingly. As sub-workflows can be regarded as individual modules, they can be treated as special artifacts following the formalization in the previous sections. Hence, we consider there are three artifacts V_1, V_2, V_3 in our experimental implementation and particularly, there are three candidates for V_1, $\{V_1^1, V_1^2, V_1^3\}$ and four candidates for V_2, $\{V_2^1, V_2^2, V_2^3, V_2^4\}$ and five candidates for $V_3, \{V_3^1, V_3^2, V_3^3, V_3^4, V_3^5\}$.

Table II Probability of each candidate being chosen in tasked executions.

A	V_1^1	V_1^2	V_1^3	V_2^1	V_2^2	V_2^3
P	0.9	0.1	0	0	0.9	0.1
A	V_2^4	V_3^1	V_3^2	V_3^3	V_3^4	V_3^5
P	0	0	0	0.9	0.1	0

Different actors would have their own strategies for choosing desired candidates to form a workflow. Overall, there are 100 trial executions of workflow. One trial execution could

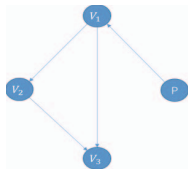

Figure 7 Bayesian network derived from Figure 6.

be uniformly randomly performed by anyone in the actor set, and for each artifact, the probability of any particular version being chosen is also uniformly distributed. Besides, there are 400 tasked executions of workflow that are performed. A tasked execution has a probability of 0.9 that it is performed by actor a_1 and a probability of 0.1 performed by actor a_2. The probability of each candidates being chosen in tasked executions is shown in Table II, respectively. A weighted hypergraph is generated based on experiment data, where each hyperedge denotes an Execution Package that includes the actor and the versioned workflow and the weight matrix represents the occurrence of one hyperedge. The weight would increase if the corresponding execution package are repeated more often, which reflects the information that this execution package could be as popular as it is valuable and that it would be captured by similarity measurement and represented in output similarity matrix. The similarity matrix $Sim \subseteq \{V \cup A\} \times \{V \cup A\}$, can be considered as the conditional probability table which would be utilized while solving the joint probability.

After retrieving the similarity matrix, we can provide the ranked recommendation by calculating the probability of each execution package, namely a typical joint probability $P(V_1, V_2, V_3. A)$, based on Bayesian Network, as shown in Figure 7. The Bayesian Network has a structure derived from workflow template.

B. VisTrails Plug-in

Without reinventing the wheel, we have built our software as a plug-in to VisTrails [1], a widely used scientific workflow

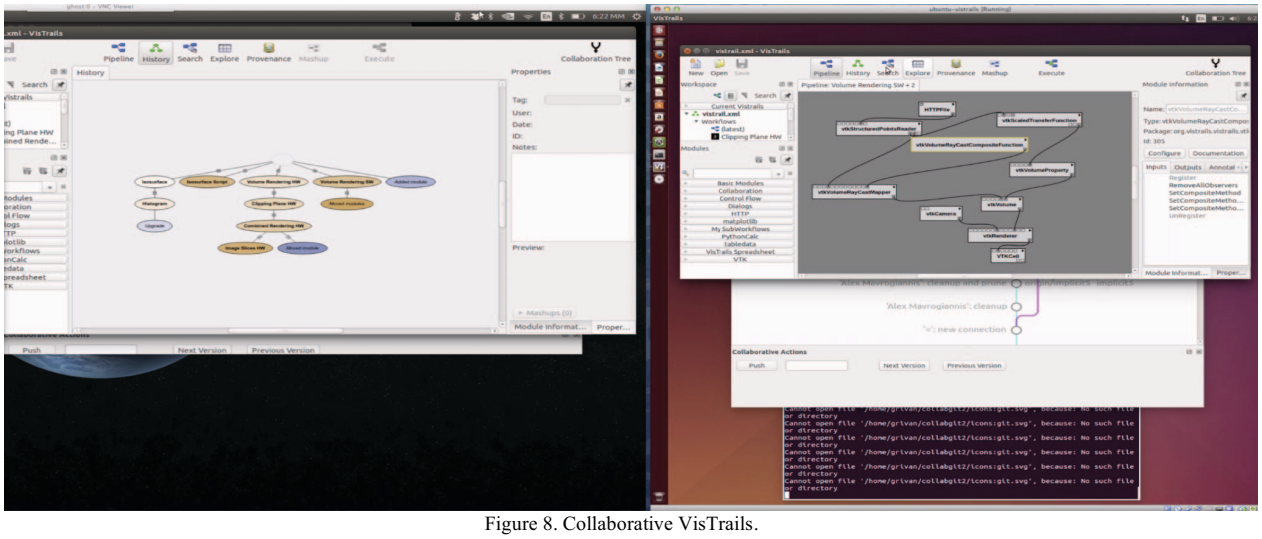

Figure 8. Collaborative VisTrails.

management system. Leveraging their software infrastructure in the last decade, we extended it into a collaborative version. Figure 8 illustrates a scenario when two researchers use our collaborative-VisTrails to design a workflow together. It shows two screens, left and right, representing two scientists working at individual VisTrails instances. Any change (adding or removal of components) made by one scientist will be immediately reflected on all other collaborators' screens. A backend version tree is established to ensure concurrency control.

We have implemented a plugin for VisTrails which utilizes Git to provide a new version tree over the existing History View. A collaboration unit is defined as the set of changes that are performed by a user between two versions of a version tree. VisTrails' History View, leaf nodes represent the latest versions of potential exploration paths. Based on our previous research on Internet-based collaboration techniques [8], we have designed and integrated a role-based collaboration protocol and technique and integrate them into our system to enable regulated scientific collaboration [41, 42].

Our plugin communicates with VisTrails during runtime in a non-intrusive manner, through the 3rd-party packages and generic VisTrails API functions that affect the current workflow and Vistrails. By using Git, we are able to implement the push(), update_repo() and checkout() operations through sequences of similar git actions, such as push, pull, checkout, branch and rebase. Git also enforces a pull-before-push policy by default, and features highly consistent locking mechanisms for high-throughput repository connections. Our collaboration plugin was implemented in Python 2.7, using the PyQt4 bindings to interact with the Qt GUI toolkit.

C. Discussions on Hypergraph Application

We have conducted a comparison of representing our CPM model between ordinary graph and hypergraph as used in our project. Shown in Figure 9, if we use ordinary graph to capture the relation between actors and versions of artifacts, the information of Execution packages would be lost. In other words, the edges in ordinary graph denote one single node has certain relation with the other in history. However, related nodes in historical records would not be kept in the graph, which happens to be important information useful for recommendation.

Hypergraph, however, has hyperedges which contain multiple nodes. It is a natural representation of tuples. In our case, one execution package would be captured as a hyperedge and occurrence of same execution package would be used as the value of hyperedge weight.

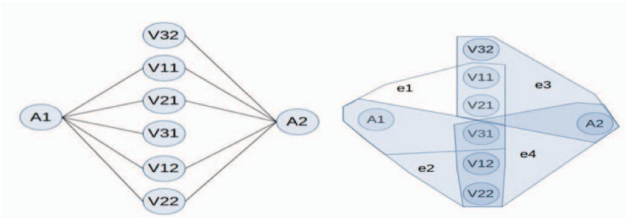

Figure 9 Comparison between ordinary graph and hypergraph.

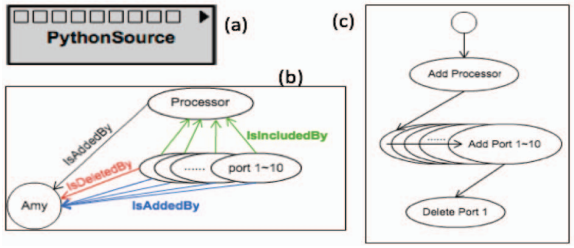

Figure 10 Example of CPM and VisTrails model. (a) workflow design result; (b) store with CPM; (c) store with VisTrails history tree

D. Discussions on CPM Model

Among three popular SWFMSs mentioned in the Introduction section, only VisTrails stores the workflow evolution history. In Figure 10, we use a simple example to compare our CPM model and VisTrails history tree.

VisTrails' history tree stores all workflow design operations. In order to obtain a snapshot of a workflow structure at a moment, one has to traverse the path starting from the root all the way to the node representing the operation at the time. Meanwhile, some operations in the path may not be shown in the resulted workflow, such as "delete" and multiple "edit" activities. Therefore, it is cumbersome to retrieve a simple entity by traversing the entire tree. In contrast, as shown in Figure 10(b), our CPM model not only stores all operations which can be traced by their timestamps, but also records the structure (through the "IsIncludedBy" relation) and the evolution (through the "IsDerivedBy" relation) of each entity. In this respect, CPM can retrieve entity structure and version history easily and directly. Furthermore, it is also easy to find contributors to a final or immediate version of a workflow. They are people who took actions over those included entities. In summary, our CPM model inherently carries more information while offering more efficient queries.

Recall the six queries posted in Section III(A), Q1~Q6. Here we provide the following cypher code in neo4j:

Q1: Show all the details about how W^{v3} has been designed and evolved as it is;

match (:Person)-[r]-()-[:IsIncludedBy]->(:Entity{id:"Wv3"}) return r order by r.time*

Q2: Return all the designers who contributed to the design of W^{v3};

match (:Entity{id:"Wv3"})<-[r:IsIncludedBy]-()-[:IsAddedBy|IsEditedBy|IsDeletedBy|IsMergedBy]->(p) return p*

Q3: Return the sub-workflows designed or refined by user s_1;

match ()-[r2:IsIncludedBy]-()-[r1:IsAddedBy|IsEditedBy]->(p:Person{name:"s1"}) return r1, r2

Q4: Return all user pairs who designed some workflows collaboratively;

match (p1)<-[:IsAddedBy|IsEditedBy|IsDeletedBy|IsMergedBy]-()-[:IsIncludedBy]->()<-[:IsIncludedBy*]-()-[:IsAddedBy|IsEditedBy|IsDeletedBy|IsMergedBy]->(p2) return p1,p2*

Q5: For workflow W^{v3}, which versions of comprising steps 1 and 2 are used? Who designed the two steps? How are they designed or refined? How are they merged?

match (:Person)-[r]-(e)-[:IsIncludedBy]->(:Entity{id:" Wv3"}) return e.name, e.version*

Q6: What are the previous versions of W^{v3}? Why was it refined?

match ()<-[r:IsDerivedBy]-(:Entity{id:"Wv3"}) return r*

VI. Conclusions

In this paper, we have reported our ongoing work that extends existing scientific workflow tools into a collaborative form that allows multiple people to cooperatively conduct data analysitcs. Extended PROV model, we present a collaborative provenance model (CPM) equipped with graph-level query formalism, pattern and algebra. On top of CPM, we have developed a hypergraph theory-based process mining technique.

In our future work, we plan to further study how CPM can answer a variety of types of queries. In addition, we will further explore hypergraph-based search algorithms. To improve the usability of cipher, which is challenging for regular users, we will explore a more high-level, user-friendly language for formulating provenance queries. Furthermore, we plan to move VisTrails online to develop an online, collaborative workflow development environment.

Acknowledgment

This work is partially supported by National Science Foundation, under grant NSF ACI-1443069. We appreciate Alexandros Mavrogiannis and Grivan Thapar for their software development efforts.

References

[1] J. Freire, C.T. Silva, S.P. Callahan, E. Santos, and C.E. Scheidegger, "Managing Rapidly-Evolving Scientific Workflows ″ Lecture Notes in Computer Science, vol. 4145/2006, col., pp. 10–18, 2006.

[2] T. Oinn, M. Greenwood, M. Addis, M.N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M.R. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe, "Taverna: Lessons in Creating a Workflow Environment for the Life Sciences,″ Concurrency and Computation: Practice & Experience, vol. 18, col. 10, pp. 1067–1100, 2006.

[3] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E.A. Lee, J. Tao, and Y. Zhao, "Scientific Workflow Management and the Kepler System,″ Concurrency and Computation: Practice & Experience, vol. 18, col. 10, pp. 1039-1065, 2006.

[4] S. Lu and J. Zhang, "Collaborative Scientific Workflows Supporting Collaborative Science,″ International Journal of Business Process Integration and Management (IJBPIM), vol. 5, col. 2, pp. 185-199, 2011.

[5] A. Lanzen and T. Oinn, "The Taverna Interaction Service: Enabling Manual Interaction in Workflows,″ Bioinformatics Applications Note, vol. 24, col. 8, pp. 1118-1120, 2008.

[6] A. Agrawal, M. Amend, M. Das, M. Ford, C. Keller, M. Kloppmann, D. König, F. Leymann, R. Müller, G. Pfau, K. Plösser, R. Rangaswamy, A. Rickayzen, M. Rowley, P. Schmidt, I. Trickovic, A. Yiu, and M. Zeller, *Web Services Human Task (WS-HumanTask), Version 1.0*, 2007 Jun.; Available from: http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-bpel4people/WS-HumanTask v1.pdf.

[7] Y. Zhao, M. Hategan, B. Clifford, I. Foster, G. Laszewski, V. Nefedova, I. Raicu, T. Stef-Praun, and M. Wilde, "Swift: Fast, Reliable, Loosely Coupled Parallel Computation," in Proceedings of *IEEE International Workshop on Scientific Workflows*, Salt Lake City, UT, USA, 2007, pp. 199–206.

[8] J. Zhang, D. Kuc, and S. Lu, "Confucius: A Tool Supporting Collaborative Scientific Workflow Composition,″ IEEE Transactions on Services Computing (TSC), vol. 7, col. 1, pp. 2-17, 2014.

[9] R. Bose and J. Frew, "Lineage Retrieval for Scientific Data Processing: A Survey,″ ACM Comput. Surv., vol. 37, col. 1, pp. 1-28, 2005.

[10] Y. Simmhan, B. Plale, and D. Gannon, "A Survey of Data Provenance in e-Science,″ SIGMOD Record, vol. 34, col. 3, pp. 31–36, 2005.

[11] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E.A. Lee, J. Tao, and Y. Zhao, "Scientific workflow management and the Kepler system,″ Concurrency and Computation: Practice and Experience, vol. 18, col. 10, pp. 1039-1065, 2006.

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

[13] "The Joint Task Force on Computing Curricula, Computing Curricula 2001,″ Journal of Educational Computing Research, vol. 1, col. 3, pp. 1-240, 2001.

[14] T. Heinis and G. Alonso, "Efficient Lineage Tracking for Scientific Workflows," in Proceedings of *ACM International Conference on Management of Data (SIGMOD)*, Vancouver, Canada, 2008, pp. 1007-1018.

[15] A. Chapman, H.V. Jagadish, and P. Ramanan, "Efficient Provenance Storage," in Proceedings of *ACM International Conference on Management of Data (SIGMOD)*, Vancouver, Canada, 2008, pp. 993-1006.

[16] O. Biton, S.C. Boulakia, S.B. Davidson, and C.S. Hara, "Querying and Managing Provenance through User Views in Scientific Workflows," in Proceedings of *IEEE 24th International Conference on Data Engineering (ICDE)*, Cancun, Mexico, 2008, pp. 1072-1081.

[17] P. Groth, S. Miles, W. Fang, S.C. Wong, K.-P. Zauner, and L. Moreau, "Recording and Using Provenance in a Protein Compressibility Experiment," in Proceedings of *the 14th IEEE International Symposium on High Performance Distributed Computing (HPDC)*, Washington, DC, USA, 2005, pp. 201-208.

[18] Y. Simmhan, B. Plale, and D. Gannon, "A Framework for Collecting Provenance in Data-Centric Scientific Workflows," in Proceedings of *IEEE International Conference on Web Services (ICWS)*, Chicago, IL, USA, 2006, pp. 427–436.

[19] *The Open Provenance Model*; Available from: http://openprovenance.org/.

[20] W3C, *An Overview of the PROV Family of Documents*; Available from: https://www.w3.org/TR/prov-overview.

[21] P. Missier, S. Dey, K. Belhajjame, V. Cuevas-Vicenttín, and B. Ludäscher, "D-PROV: Extending the PROV Provenance Model with Workflow Structure," in Proceedings of *The 5th USENIX Workshop on the Theory and Practice of Provenance (TaPP)*, Lombard, IL, USA, 2013, pp.

[22] V. Cuevas-Vicenttin, S. Dey, M. Wang, T. Song, and B. Ludäscher, "Modeling and querying scientific workflow provenance in the D-OPM," in Proceedings of *High Performance Computing, Networking, Storage and Analysis, SC Companion*, 2012, pp. 119-128.

[23] S.J.H. Yang, J. Zhang, L. Lin, and J.J.P. Tsai, , 36(3), 2009, pp. , "Improving Peer-to-Peer Search Performance through Intelligent Social Search,″ Expert Systems with Applications, vol. 36, col. 3, pp. 10312-10324, 2009.

[24] C. Lim, S. Lu, A. Chebotko, and F. Fotouhi, "Prospective and Retrospective Provenance Collection in Scientific Workflow Environments," in Proceedings of *IEEE International Conference on Services Computing (SCC)*, 2010, pp. 449-456.

[25] A. Chebotko, X. Fei, C. Lin, S. Lu, and F. Fotouhi, "Storing and Querying Scientific Workflow Provenance Metadata Using an RDBMS," in Proceedings of *IEEE International Conference on e-Science and Grid Computing*, 2007, pp. 611-618.

[26] O. Biton, S. Cohen-Boulakia, S.B. Davidson, and C.S. Hara, "Querying and Managing Provenance through User Views in Scientific Workflows," in Proceedings of *IEEE 24th International Conference on Data Engineering*, 2008, pp. 1072-1081.

[27] C. Scheidegger, D. Koop, E. Santos, H. Vo, S. Callahan, J. Freire, and C. Silva, "Tackling the Provenance Challenge One Layer at a Time,″ Concurrency and Computation: Practice and Experience, vol. 20, col. 5, pp. 473-483, 2008.

[28] L.M.G. Jr, M. Mattoso, M. Wilde, and I.T.F. ", "Provenance Query Patterns for Many-Task Scientific Computing," in Proceedings of *The 3rd USENIX Workshop on the Theory and Practice of Provenance (TaPP)*, Heraklion, Crete, Greece, 2011, pp.

[29] K. Anand, S. Bowers, and B. Ludäscher, "Techniques for Efficiently Querying Scientific Workflow Provenance Graphs," in Proceedings of *EDBT*, 2010, pp. 287-298.

[30] G. Fakas and B. Karakostas, "A Workflow Management System Based on Intelligent Collaborative Objects," Information & Software Technology, vol. 41, col. 13, pp. 907-915, 1999.
[31] H. Song, J.J. Dong, C. Han, W.R. Jung, and C.-H. Youn, "A SLA-Adaptive Workflow Integrated Grid Resource Management System for Collaborative Healthcare Services," in Proceedings of *the 3rd International Conference on Internet and Web Applications and Services (ICIW)*, Athens, Greece, 2008, pp. 702-707.
[32] L. Pudhota and E. Chang, "Collaborative Workflow Management Using Service Oriented Approach," in Proceedings of *International Conference on E-Business, Enterprise Information Systems, E-Government (EEE)*, Las Vegas, USA, 2005, pp. 167-173.
[33] C.-J. Huang, C.V. Trappey, and C.C. Ku, "A JADE-Based Autonomous Workflow Management System for Collaborative IC Design," in Proceedings of *the 11th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Melbourne, Australia, 2007, pp. 777-782.
[34] A. Agrawal, M. Amend, M. Das, M. Ford, C. Keller, M. Kloppmann, D. König, F. Leymann, R. Müller, K. Plösser, R. Rangaswamy, A. Rickayzen, M. Rowley, P. Schmidt, I. Trickovic, A. Yiu, and M. Zeller, *WS-BPEL Extension for People (BPEL4People), Version 1.0*, 2007 Jun.; Available from: http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-bpel4people/BPEL4People_v1.pdf.
[35] N. Ayachitula, M.J. Buco, Y. Diao, M. Surendra, R. Pavuluri, L. Shwartz, and C. Ward, "IT Service Management Automation - A Hybrid Methodology to Integrate and Orchestrate Collaborative Human Centric and Automation Centric Workflows," in Proceedings of *IEEE International Conference on Services Computing (SCC)*, Salt Lake City, UT, USA, 2007, pp. 574-581.
[36] D. Russell, P.M. Dew, and K. Djemame, "Service-Based Collaborative Workflow for DAME," in Proceedings of *IEEE International Conference on Services Computing (SCC)*, Orlando, FL, USA, 2005, pp. 139-146.
[37] D. Jordan and J. Evdemon, *Web Services Business Process Execution Language, Version 2.0*, 2007 Apr.; Available from: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.
[38] W.M.P.v.d. Aalst, "On the Automatic Generation of Workflow Processes based on Product Structures," Computers in Industry, vol. 39, col. 2, pp. 97–111, 1999.
[39] S. Zhang, Y. Xiang, Y. Shen, and M. Shi, "Knowledge Modeling and Optimization in Pattern-Oriented Workflow Generation," in Proceedings of *CSCWD*, 2008, pp. 636-642.
[40] H. Liu, P. LePendu, R. Jin, and D. Dou, "A Hypergraph-based Method for Discovering Semantically Associated Itemsets," in Proceedings of *11th IEEE International Conference on Data Mining*, Vancouver, Canada, 2011, pp. 398-406.
[41] J. Zhang, C.K. Chang, and J. Voas, "A Uniform Meta-Model for Mediating Formal Electronic Conferences," in Proceedings of *the 28th IEEE Annual International Computer Software and Applications Conference (COMPSAC)*, Hong Kong, China, 2004, pp. 376-383
[42] J. Zhang, C.K. Chang, and J.-Y. Chung, "Mediating Electronic Meetings," in Proceedings of *the IEEE 27th Annual International Computer Software and Applications Conference (COMPSAC)*, Dallas, TX, USA, 2003, pp. 216-221.