Towards Workflow Ecosystems Through Semantic and Standard Representations

Daniel Garijo
Ontology Engineering Group
Universidad Politécnica de Madrid
dgarijo@fi.upm.es

Yolanda Gil
Information Sciences Institute
University of Southern California
gil@isi.edu

Oscar Corcho
Ontology Engineering Group
Universidad Politécnica de Madrid
ocorcho@fi.upm.es

ABSTRACT

Workflows are increasingly used to manage and share scientific computations and methods. Workflow tools can be used to design, validate, execute and visualize scientific workflows and their execution results. Other tools manage workflow libraries or mine their contents. There has been a lot of recent work on workflow system integration as well as common workflow interlinguas, but the interoperability among workflow systems remains a challenge. Ideally, these tools would form a *workflow ecosystem* such that it should be possible to create a workflow with a tool, execute it with another, visualize it with another, and use yet another tool to mine a repository of such workflows or their executions. In this paper, we describe our approach to create a workflow ecosystem through the use of standard models for provenance (OPM and W3C PROV) and extensions (P-PLAN and OPMW) to represent workflows. The ecosystem integrates different workflow tools with diverse functions (workflow generation, execution, browsing, mining, and visualization) created by a variety of research groups. This is, to our knowledge, the first time that such a variety of workflow systems and functions are integrated.

1. INTRODUCTION

The interoperability of workflow systems is an active area of research, including standard languages for representing workflow executions as provenance [24] [18], integration efforts that demonstrate the exchange of workflows across different systems [21] [17], and architectures with modular design that can integrate alternative systems such as diverse execution engines [13].

As workflow technologies continue to mature, new tools are being developed with new functions that address different requirements for workflows, including design [26] [19] [11] [30] [25], validation [11], execution [5] [30] [25] [23], visualization [25] [11] [26], and mining [10] [14] [29]. The interoperability of all these different workflow tools remains an open research area. Ideally, given some requirements for an application it would be easy to assemble an end-to-end system out of the workflow tools available. Another scenario for interoperability is supporting users that have new requirements over time. For example, a user may execute workflows with a particular a tool for many months and then later be willing to import all the workflow runs into another tool to browse and to query provenance traces.

This paper describes a workflow ecosystem that integrates different workflow tools with diverse functions (workflow design, validation, execution, visualization, browsing and mining) created by a variety of research groups. This is, to our knowledge, the first time that such a variety of workflow systems and functions are integrated. In addition, we demonstrate that workflows produced by a given tool can be used by more than one other tool. Previous demonstrations of interoperability address some of these aspects, but our workflow ecosystem is the first of its kind in the multiple dimensions of heterogeneity that are addressed. Our approach is to use workflow representation standards and semantic technologies to enable each tool to import workflow templates and executions in the format they need. We use and extend the Open Provenance Model (OPM) [24], adopted by many workflow systems, and the recent W3C PROV [15] standard.

The paper begins introducing the notion of workflow ecosystems in Section 2. We describe relevant prior work on workflow interoperability in Section 3, highlighting the contributions that each interoperability demonstration makes to the design of workflow ecosystems. Then we describe in Section 4 the workflow ecosystem that we have developed, explaining the different tools that it integrates and the kinds of workflow structures that they exchange. We then show the workflow representation standards and semantic web standards used in our approach. Finally, we discuss the limitations of our approach in Section 5, and the plans for future work in Section 6.

2. WORKFLOW ECOSYSTEMS

The term "workflow system" is commonly used in the literature to refer to software that has a diversity of functions and typically includes a workflow execution capability of some sort. We wish to take a broader view on workflow software, and include other

software that has useful workflow capabilities, for example a workflow browser, a workflow editor, or a workflow mining system. Therefore we introduce here the terms "workflow tool", "workflow function", and "workflow ecosystem".

We refer to a ***workflow tool*** as any kind of software that consumes or generates workflows, and are typically called "workflow systems" in the literature. Systems such as Taverna [25], Pegasus [5], Kepler [19], Moteur [23], etc, are workflow tools in our terminology. But in our terminology, workflow tools also include software that is less comprehensive than a workflow system, such as workflow browsers, workflow mining tools, etc.

A ***workflow function*** indicates a unique capability regarding workflows, such as workflow execution, workflow browsing, or workflow mining. A given workflow tool can have several functions. For example, Taverna includes a workflow editor function, a workflow execution function, and a workflow provenance recording function. When workflow tools are integrated, the integration may involve only a particular function of the tool. For example, when workflow tool A sends a provenance record to workflow tool B which stores it, the execution engine of A would be involved but not its workflow editor or other functions.

As workflows technologies and workflow interchange standards mature, these kinds of integration efforts will give raise to ***workflow ecosystems*** that scale up integration efforts by demonstrating the integration along three important dimensions of heterogeneity:

1. **Functional heterogeneity:** The diversity of workflow tools and workflow functions involved. The workflow tools have diverse functions, be developed by independent parties in different organizations. The integrations demonstrated to date involve typically one or two workflow tools encompassing 2-3 functions developed by 2-3 parties. A workflow ecosystem would include tools with very diverse functions (e.g., workflow execution plus a workflow repository plus a workflow browser plus a workflow miner), each coming from a different development group.
2. **Usage heterogeneity:** A workflow output by a workflow tool can be consumed by at least two other workflow tools. In the integrations demonstrated to date, the output of a workflow tool is consumed by at most one other tool, and when it is consumed by several tools these tools have the same function. In a workflow ecosystem it is possible to import the same workflow into tools that have different functions (e.g., a workflow execution is mined and visualized by different tools).
3. **Abstraction heterogeneity:** A workflow tool can import abstract or detailed views of workflows based on the level of granularity that the tool is able to handle. In the integrations to date, a workflow tool exports a workflow representation that has to be ingested in its entirety by another workflow tool. This feature is central to scaling up workflow ecosystems because it allows a loose coupling between tools that act as workflow producers and those that act as workflow consumers.

These characteristics of workflow ecosystems have been partially addressed by prior work, but none of the prior interoperability efforts has addressed the heterogeneity and functional scale up requirements of a workflow ecosystem.

Table 1. Dimensions to characterize workflow ecosystems.

Dimension of Heterogeneity	*Description*
Functional	Include tools with very diverse functions (e.g., editor, execution, provenance, browsing, mining, etc.)
Use	The same workflow output by a tool is consumed by tools with different functions
Abstraction	Each workflow tool can import abstract or detailed views of the same workflow based on the tool's function.

3. PRIOR WORK ON WORKFLOW INTEROPERABILITY

There is a lot of prior work that has addressed interoperability across workflow systems through workflow interchange languages, system integrations, and workflow repositories.

There have been several efforts to develop workflow interchange languages. The Open Provenance Model [24] is the result of a community effort to create a common representation for workflow executions, and is used by many workflow systems as an interchange format [21]. The recent W3C PROV standard for provenance [15] continues this work, and several approaches extend it to represent scientific workflow provenance [10] [2] [22]. Another significant effort to develop a common workflow language is the IWIR language [28]. Workflows represented in IWIR can be partitioned so that each partition is executed in a different workflow execution engine. WS-BPEL[1] is a language for specifying business processes using web services, and BPMN[2] provides a standard graphical notation for business processes.

Workflow repositories such as myExperiment [6] and CrowdLabs [20] can be used for sharing scientific workflows created with different systems. These repositories store workflows in their native languages, that is, without requiring their conversion to a common language. Tools that interoperate with these repositories tend to import only the workflows that are in a language that they understand. For example [14] and [29] mined myExperiment to find subworkflows or analyze workflow reuse, but only extracted workflows that were implemented in Taverna's format. Our goal is to build on these efforts and facilitate further interoperation across workflow systems.

Other efforts are focused on integrating workflow systems. System integrations have been done with Pegasus [5], Taverna [25], Vistrails [26], Kepler [19], etc. Taverna was integrated with a workflow mining tool [14], as well as the myExperiment workflow repository [6]. Also, Taverna was able to export workflows to the Galaxy workflow system[3] [1]. Pegasus has been integrated with the Condor execution system, with the WINGS workflow generation system [11], and with the PASOA provenance store [21]. More recently, the Wf4ever project[4] has developed tools for preserving Workflow Research Objects [2], including tools for designing, executing, visualizing and storing scientific workflows, along with the models for accomplishing it.

[1] http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html
[2] http://www.omg.org/spec/BPMN/2.0/
[3] http://www.taverna.org.uk/documentation/taverna-galaxy/
[4] http://www.wf4ever-project.org/

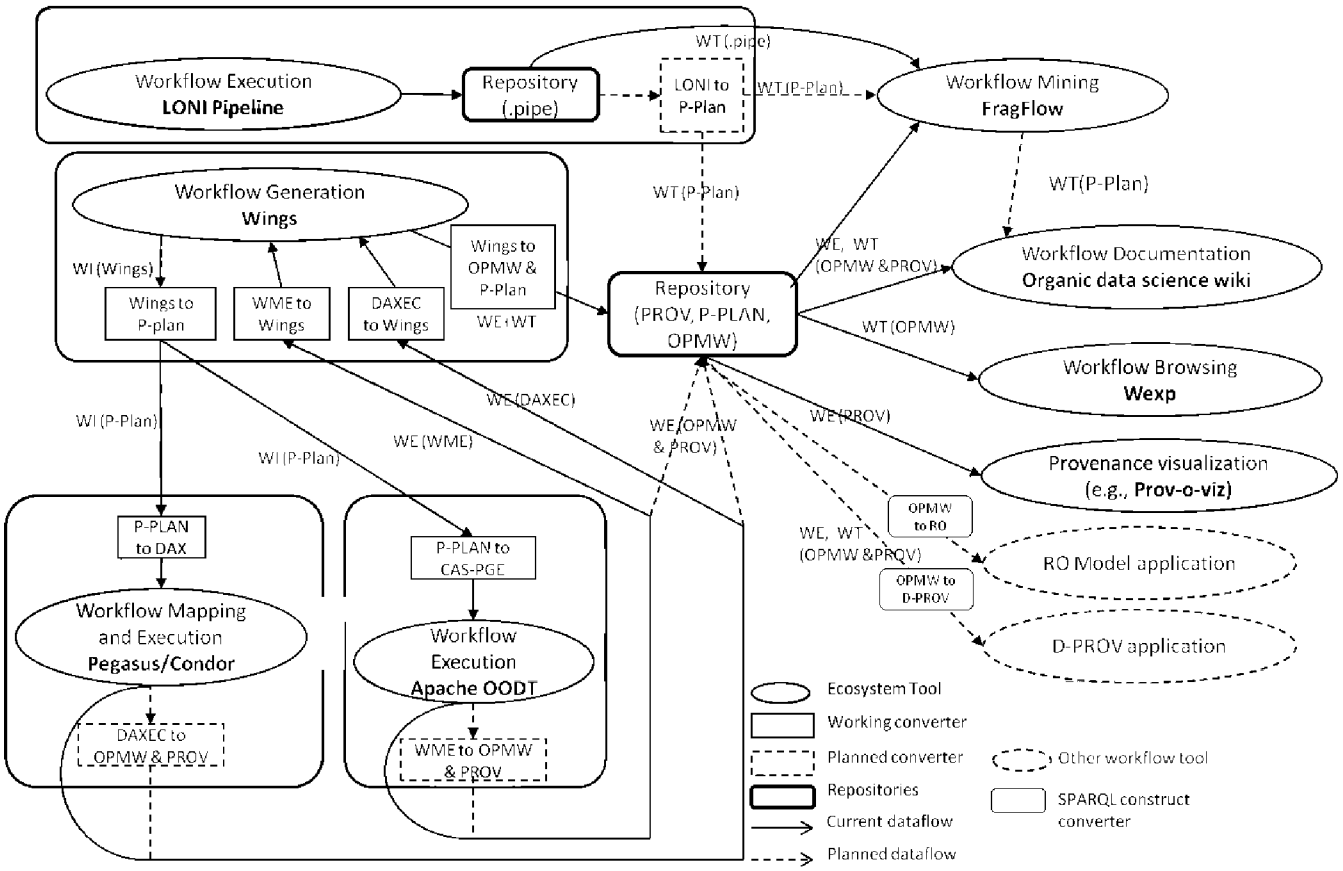

Figure 1: The WEST workflow ecosystem. Workflow Templates (WT), Workflow Instances (WI) and Workflow Executions (WE) are interchanged among the different components of the ecosystem. Blue lines group converters with their corresponding workflow tools.

4. THE WEST WORKFLOW ECOSYSTEM

WEST (Workflow Ecosystems through STandards) addresses our three dimensions of heterogeneity: 1) functional diversity through the integration of nine workflow functions coming from six different research groups, 2) usage diversity through the integration of tools that produce workflows consumed by at least two other tools, and 3) abstraction diversity through the use of workflow representation standards and semantic web standards that enable query-based retrieval of workflow views.

Figure 1 shows an overview of the current incarnation of the WEST workflow ecosystem, showing also planned future interoperability pathways. The different tools that WEST integrates are represented in ellipses, while the workflow repositories are represented with rounded boxes with thick border. The small rectangular boxes depict the converters of the internal vocabularies used within the individual tools into the standard representations used by WEST. Converters planned but not yet implemented are indicated with dashed lines. Lightweight converters adapting our vocabulary to other extensions of the standards are shown in small rounded boxes (their respective tools are represented as dotted ellipses). The directionality of the arrows between the different tools indicates what tools produce and consume the workflows exchanged across tools in the ecosystem.

We describe first the kinds of workflow structures exchanged across the different tools integrated in WEST, and then describe in detail each of the tools in the WEST workflow ecosystem.

4.1 Workflow Structures

Different workflow tools produce and consume workflows with different levels of specificity. We distinguish three major types of workflow structures that are exchanged across tools within our WEST workflow ecosystem:

- *Workflow Instance (WI)*: A workflow that specifies the application algorithms to be executed and the data to be used. Workflow instances are sometimes called abstract workflows because they do not specify execution resources. A workflow instance can be submitted to a workflow mapping and execution system, which will identify and assign available resources at run-time, submit it for execution, oversee its execution, and return a workflow execution trace. Because different resources may be available at different times or in different execution environments, the same workflow instance could result in very different workflow execution traces.

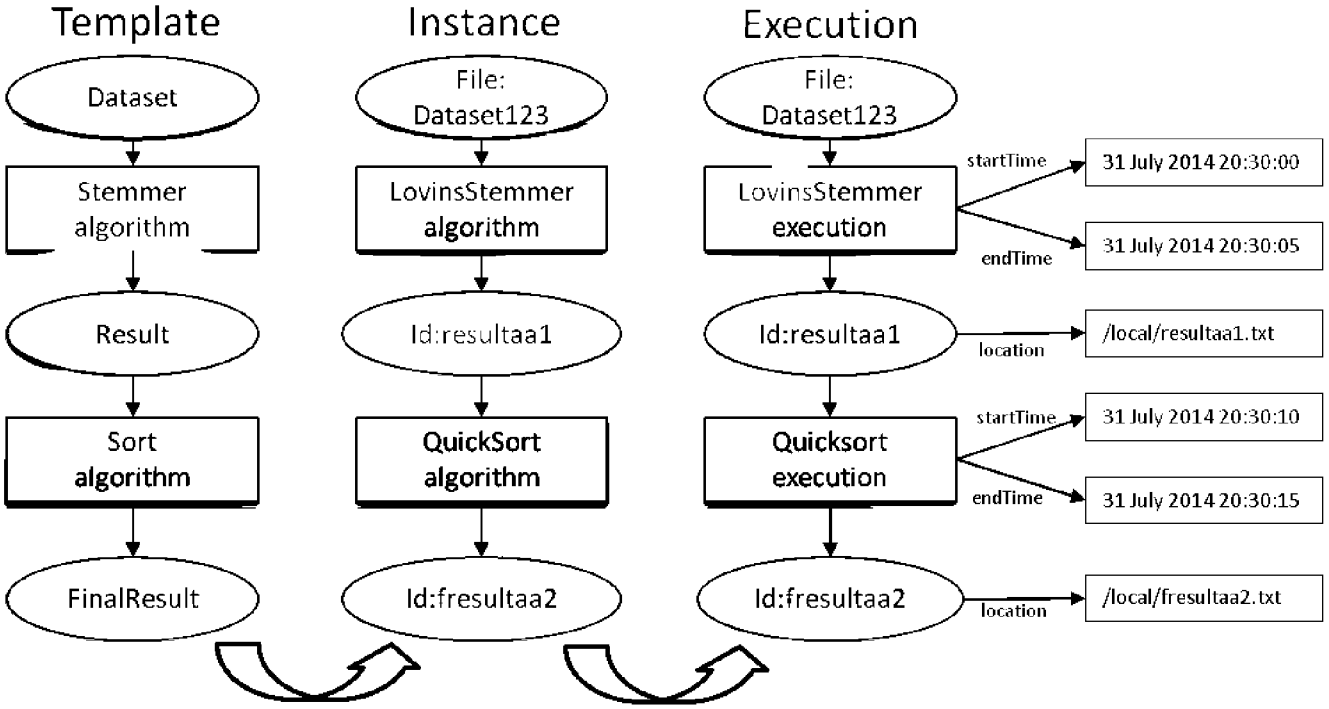

Figure 2: A Workflow Template (left), a Workflow Instance (center), and a Workflow Execution (right).

- *Workflow Execution (WE)*: Also known as a workflow execution trace or a provenance trace, a workflow execution contains the details of what happened during the execution of a workflow, including what resources it was executed on, execution time for each step, links to the intermediate results, and possibly other execution details such as steps for data movements. When workflow steps fail to execute, a workflow execution contains annotations of what failed and in this way its dataflow structure may be different from the dataflow in a workflow instance.
- *Workflow Template (WT)*: A generic reusable workflow specification that indicates the types of steps in the workflow and their dataflow dependencies. A workflow instance can be created from a workflow template when datasets are identified. A workflow generation tool can take the types of step specified in the workflow template (e.g., "sort") and specialize them to implemented algorithms and codes (e.g., "Quicksort in Python") to create the workflow instance. Since a type of step can have different implementations, the same workflow template could be used to generate very different workflow instances.

Figure 2 illustrates with an example the difference between templates, instances and executions. On the left of the figure we can see a template with two steps (Stem and Sort), an input (Dataset) and an output (FinalResult). A workflow instance built from this template is shown in the middle of the figure, specifying Dataset123 as the input and specific executable codes for each of the steps (the LovinsStemmer algorithm and Quicksort algorithm respectively). When this workflow instance is executed, the workflow execution engine produces the execution trace shown on the right (each step has its start and end time, and each intermediate result identifier, e.g., Id:resultaa1, has associated the path of the file it represents).

Figure 1 shows the kinds of workflow structures exchanged by the different tools in WEST with the labels WT, WI, and WE, along with the vocabulary representing it in brackets (PROV, P-Plan and OPMW).

4.2 Workflow Representation Standards

We use the W3C PROV standard [15] and OPM [24] to represent workflow executions as provenance records. In addition, we use two extensions of these languages that are more specific to workflows and enable us to represent workflow templates and instances as well as further details of workflow executions: P-plan [8], and a revised version of OPMW[5] [9] .

The tools and systems in WEST have been created by different institutions, and each of them produces and consumes different types of information about workflow templates and executions, and at different granularities. For example, a PROV visualizing tool exposes a view of a workflow execution that is simpler than the one required for a visualization tool made for a particular workflow system. The use of semantic technologies facilitates this kind of selective extraction of the needed information.

In this section we introduce all the previous models, and explain how they facilitate interoperation.

4.2.1 Representing the provenance of Workflow Executions: OPM and PROV

After the Third Provenance Challenge[6], the Open Provenance Model (OPM) [24] consolidated itself as a *de facto* standard for representing provenance, and was adopted by many workflow

[5] http://www.opmw.org/model/OPMW/

[6] http://twiki.ipaw.info/bin/view/Challenge/FirstProvenanceChallenge

systems[7]. The interest in having a standard for provenance interchange vocabulary led to the W3C Provenance Incubator Group[8], which was followed by the Provenance Working Group[9]. This effort finally materialized in the family of PROV specifications [15], a set of W3C recommendations on how to model and interchange provenance in the Web. In this section we briefly describe OPM and PROV, along with their main similarities.

4.2.1.1 The Open Provenance Model

OPM models the resources created as *artifacts* (immutable pieces of state), the steps used as *processes* (action or series of actions performed on artifacts), and the entities that control those processes as *agents*. Their relationships are modeled in a provenance graph with five main causal edges: *used* (a process used some artifact), *wasControlledBy* (an agent controlled some process), *wasGeneratedBy* (a process generated an artifact), *wasDerivedFrom* (an artifact was derived from another artifact) and *wasTriggeredBy* (a process was triggered by another process). It also introduces the concept of *roles* to assign the type of activity that artifacts, processes or agents played when interacting with one another, and the notion of *accounts* and *provenance graphs*. An account represents a particular view on the provenance of an artifact based on what was executed. A provenance graph groups sets of related OPM assertions. OPM does not specify any concept for the modeling of plans, so it can only be used to describe workflow executions and it cannot be used to describe workflow instances or workflow templates.

OPM is available as two different ontologies that are built on top of each other. The OPM Vocabulary (OPMV)[10] is a lightweight RDF vocabulary implementation of the OPM model that only has a subset of the concepts in OPM but facilitates modeling and query formulation. The OPM Ontology (OPMO)[11] covers the full functionality of the OPM model, and can be used to represent OPM concepts that are not in OPMV, such as Account or Role.

4.2.1.2 The W3C PROV Standard

The PROV model is heavily influenced by OPM. PROV models resources as *entities* (which can be mutable or immutable), the steps using and generating entities as *activities*, and the individuals responsible for those activities as *agents*. The relationships are also modeled in a provenance graph with seven main types of edges: *used* (an activity used some entity), *wasAssociatedWith* (an agent participated in some activity), *wasGeneratedBy* (an activity generated an entity), *wasDerivedFrom* (an entity was derived from another entity), *wasAttributedTo* (an entity was attributed to an agent), *actedOnBehalfOf* (an agent acted on behalf of another agent) and *wasInformedBy* (an activity used the entity produced by another activity).

PROV also keeps the notion of *roles* to describe how entities, activities and agents behaved in a particular event (usage, generation, etc.); and provides the means to qualify each of the aforementioned relationships using an n-ary pattern. PROV allows to state the *plan* associated to a certain activity, although the plan definition itself is out of the scope of the model (since it is not something that necessarily happened). PROV statements can be grouped in named sets called *bundles*, which are entities themselves (thus allowing for their provenance to be described). The PROV standard is available as an ontology (PROV-O [18]).

4.2.1.3 Comparison Between OPM and PROV

There is a very clear correspondence between OPM and PROV, and this facilitates the reuse of workflows represented in one language by tools that consume the other.

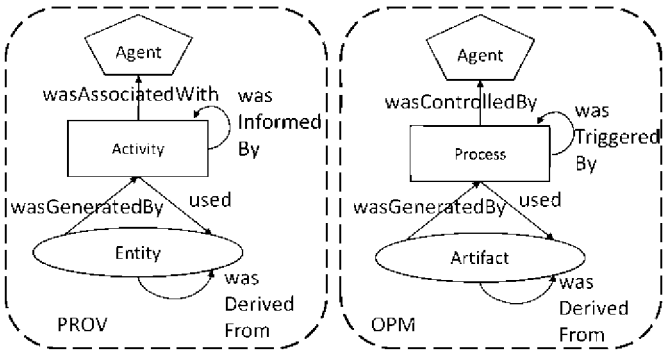

Figure 3: The commonalities between PROV (left) and OPM (right) facilitate mappings across both representations.

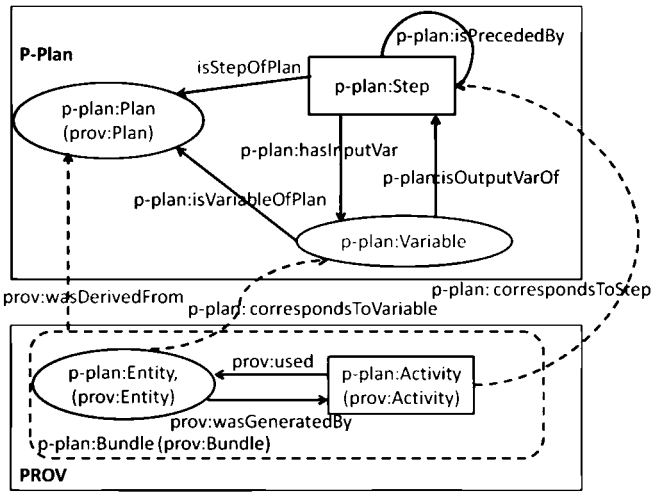

Figure 4: P-plan is an extension of PROV.

Figure 3 illustrates the main commonalities between OPM and PROV that are relevant to our scenario. Both models represent resources being used and generated by processes or activities which are controlled by an agent responsible for its execution. Entities (or artifacts, respectively) can be derived from other entities. Also, in OPM processes might be *triggered by* other processes, while in PROV Activities might receive an input created by another activity (being *informedBy*).

4.2.2 Representing Workflow Templates and Instances: P-PLAN

We cannot use a provenance language like OPM or PROV to represent workflow templates and workflow instances. We need a language that can represent process models or plans that when executed lead to a provenance trace that can be expressed in OPM or PROV.

[7] http://openprovenance.org/
[8] http://www.w3.org/2005/Incubator/prov/
[9] http://www.w3.org/2011/prov/wiki/Main_Page
[10] http://purl.org/net/opmv/ns
[11] http://openprovenance.org/model/opmo

P-Plan [8][12] is an extension of PROV for representing plans. Figure 4 shows an overview of the P-PLAN vocabulary. A p-plan:Plan is a subclass of prov:Plan. The p-plan:Steps represent the planned execution activities. Plan steps may be bound to a specific executable step (p-plan:correspondsToStep) or refer to a class of steps, providing an abstraction layer over the execution. As a result, a plan step could be carried out in different ways in different executions of the same plan. A step may not have a corresponding activity, (as in an execution failure). A p-plan:Variable represents the inputs of a step and can have

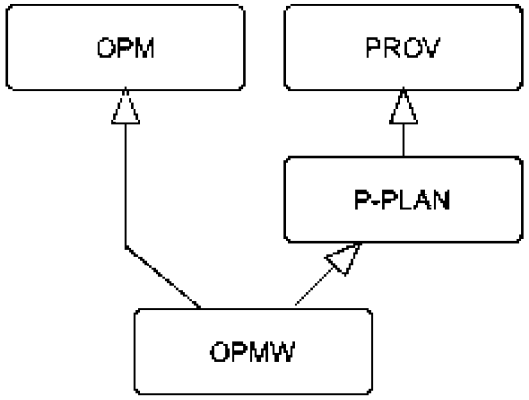

Figure 5: OPMW and its relationship to the OPM, PROV, and P-PLAN vocabularies.

properties (i.e., type, restrictions, metadata, etc.). p-plan:Steps may be preceded by other p-plan:Steps (p-plan:isPrecededBy), and have p-plan:Variables as input. p-plan:Variables are output of p-plan:Steps. Both steps and variables are associated to a p-plan: Plan (p-plan:isStepOfPlan and p-plan:isVariableOfPlan respectively). The relation of the plan with agents is not specified in P-PLAN, since it can be modeled with PROV.

4.2.3 Representing Workflows at Fine Granularity: OPMW

Workflow templates, instances, and executions can be represented with the OPMW model. The new OPMW release is the fifth revision of the OPMW vocabulary [9], designed to represent scientific workflows. OPMW extends P-plan, PROV and OPM. OPMW supports the representations of workflows at a fine granularity with a lot of details pertaining to workflows that are not covered in those more generic languages. OPMW also allows the representation of links between a workflow template, a workflow instance created from it, and a workflow execution that resulted from an instance. Finally, OPMW also supports the representation of attribution metadata about a workflow, which some applications consume.

Figure 5 shows the relationships between the different vocabularies used in WEST, and how OPMW extends PROV, OPM and P-PLAN.

OPMW separates the issue of workflow representation in two different aspects: linkage between templates and executions and attribution.

4.2.3.1 Linking templates and executions in OPMW

In OPMW, an opmw:WorkflowTemplate is a subclass of p-plan:Plan (since it is a particular type of plan), opmw:WorkflowTemplateProcess is a subclass of p-plan:Step and opmw:WorkflowTemplateArtifact extends p-plan:Variable respectively (since both of them refer to a particular domain).

On the execution side, each *WorkflowExecutionProcess* (subclass of prov:Activity and opmo:Process) is bound to a *WorkflowTemplateProces* via the *correspondsToTemplateProcess* relationship (subproperty of *p-plan:correspondsToStep*). Similarly, each *WorkflowExecutionArtifact* (subclass of prov:Entity and opmo:Artifact respectively) is linked to its abstract *WorkflowTemplateArtifact* with the *correspondsToTemplateArtifact* relationship (subproperty of *p-plan:correspondsToVariable*). Finally, the *WorkflowExecutionAccount* containing all the provenance statements of the execution is linked to the *WorkflowTemplate* that contains all the assertions of the template with the *correspondsToTemplate* relationship.

Figure 6 shows an example of the OPMW vocabulary extending OPM, PROV and P-PLAN. A workflow template with one sorting step, an input and an output (on the left of the figure, represented using P-Plan) is linked to its provenance trace on the right of the figure (depicted with PROV and OPM). Each activity and artifact is linked to its respective step and variable. Additional metadata of the variables (e.g., constraints), steps (e.g., conditions for execution), activities (e.g., used code), artifacts (e.g., size, encoding), accounts (e.g., status) and templates (e.g., associated diagram) is modeled with OPMW, but has been omitted from the figure for simplicity.

4.2.3.2 Workflow Attribution in OPMW

Attribution is crucial for scientists who create and publish workflows, and OPMW can be used to represent such metadata in workflow templates, instances, and executions. For this, OPMW reuses terms from the Dublin Core (DC) Metadata Vocabulary[13], namely author, contributor, rights and license. OPMW also defines additional terms for referring to the start and end of the whole execution of the workflow, the size of the produced files, the status of the final execution, the tool used to design the workflow, the tool used to execute the workflow, etc.

[12] http://purl.org/net/p-plan

[13] http://dublincore.org/documents/dcmi-terms/

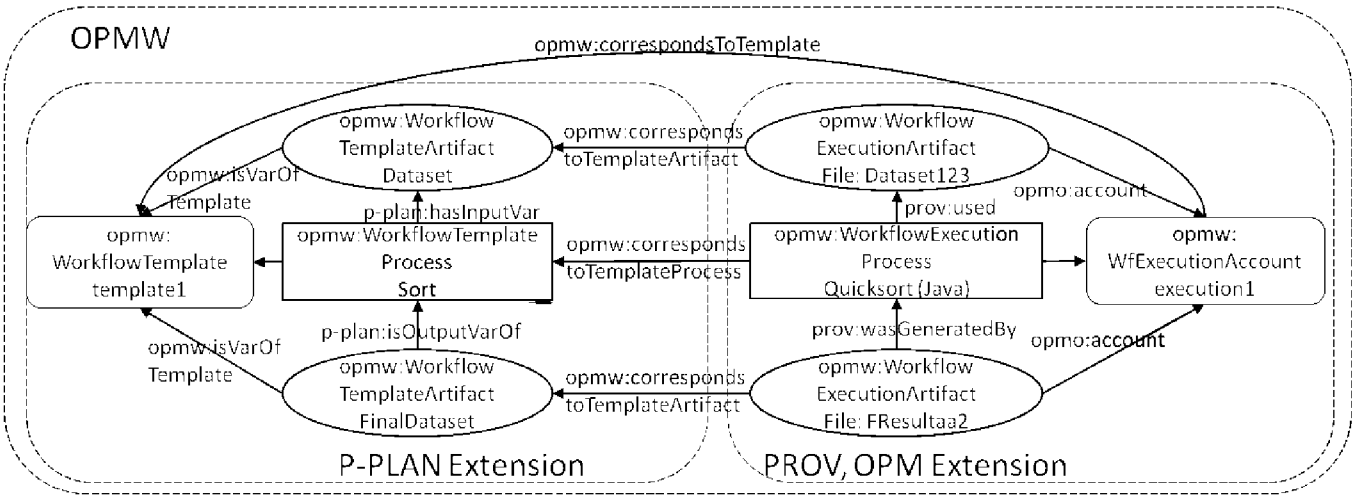

Figure 6: Example of OPMW as an extension of PROV, OPM and P-Plan. A workflow execution (right) is linked with its workflow template (left). Other details like attribution metadata have been omitted to simplify the figure.

Figure 7: PROV-O-viz visualization of part of a workflow execution trace.

4.3 Functional Heterogeneity in WEST

Figure 1 indicated all the workflow tools that we have integrated in the WEST ecosystem to date. They represent a wide variety of functions:

- *Workflow Generation*: WEST integrates the WINGS workflow generation tool [11]. Users create workflow templates, which WINGS can specialize to generate workflow instances. WINGS can submit the workflow instances for execution by different workflow mapping and execution engines.
- *Workflow Mapping and Execution*: WEST includes three workflow execution engines: Pegasus [5], Apache OODT[14], and the LONI Pipeline [30]. These systems map the workflow tasks to available execution resources, and then manage their execution. It would be easy to include other workflow execution engines, since many of them use OPM and are beginning to use PROV.
- *Workflow Mining*: WEST includes the FragFlow system for workflow mining [10], which integrates several algorithms for extracting common workflow fragments from repositories of workflow templates and workflow executions. Including other workflow mining tools would not be hard, as they are typically based on graph mining or process mining algorithms that operate on data structures that are similar to those used in PROV and OPM.
- *Workflow Visualization*: WEST uses the Prov-o-viz tool [16] for visualizing provenance structures expressed in the W3C PROV standard. A screenshot is shown in Figure 7. Many tools for browsing, storing, visualizing and validating provenance have been developed for PROV. A workflow ecosystem that uses PROV can benefit from these applications, in order to check the consistency of an execution trace or to gather a general insight on how the inputs influence the final results.
- *Workflow Browsing*: WEST uses a Workflow Explorer tool, WExp[15], which allows for exploring different workflow templates, their metadata and their workflow execution results. A snapshot is shown in Figure 8. WExp loads dynamically the workflow information stored in a repository, and allows the user to search it on demand.
- *Workflow Documentation*: WEST includes the Organic Data Science Wiki[16], an extension of semantic wikis designed to develop meta-workflows that result in many workflow explorations and runs. A snapshot of the interface is shown in Figure 9. Workflow templates and workflow executions can be imported into this framework to generate documentation pages that link to data and algorithm descriptions in the wiki, and users can augment this

[14] http://oodt.apache.org/

[15] http://purl.org/net/wexp

[16] http://www.organicdatascience.org/index.php/Main_Page

documentation as they relate to the overall meta-workflow.

- *Workflow Storage and Sharing:* WEST has a workflow repository that includes workflow templates, workflow instances, and workflow executions. This is a public repository[17], implemented using Virtuoso, and populated by WINGS. All workflows are Web objects that are openly accessible to any application that queries the repository. We illustrate this further in Section 4.5.

Therefore, WEST includes nine different tools that provide seven distinct functions. Many of these tools were developed by other research groups: Pegasus by the USC/ISI Collaborative Science group, LONI Pipeline by the USC Laboratory of Neuroimaging, Prov-o-viz by the VUE in Amsterdam, and Apache OODT was originally developed at NASA/JPL.

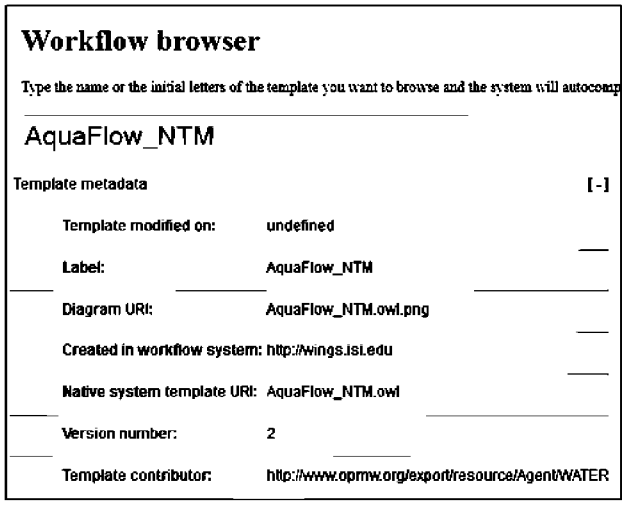

Figure 8: Snapshot of the WExp browser showing some metadata of a water metabolism workflow.

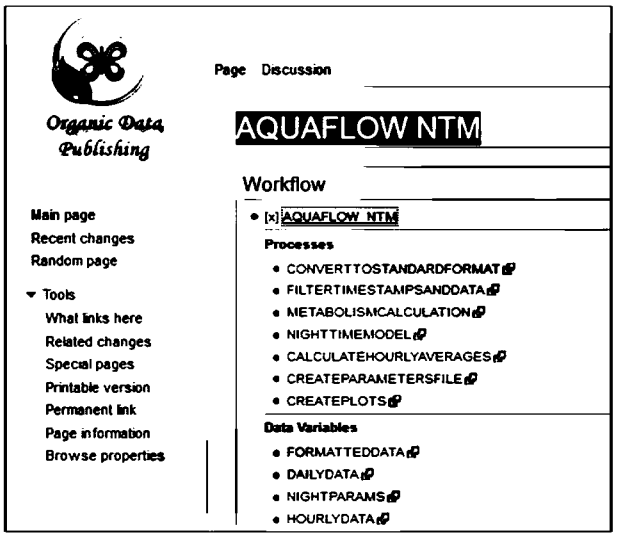

Figure 9: Snapshot of an Organic Data Science Wiki page created from the OPMW representation of the water metabolism workflow mentioned in Figure 8.

[17] http://www.opmw.org/sparql

4.4 Usage Heterogeneity in WEST

Every workflow output in the WEST workflow ecosystem is either consumed by at least two different applications or stored in the repository where it is later consumed by one or more applications.

Table 1 summarizes the tools in the WEST workflow ecosystem consuming and producing workflow templates, instances and executions. For example, the workflow instances generated by WINGS are used by both Pegasus and Apache OODT.

Workflow output	Consumed by
Workflow Instance (from WINGS)	Pegasus, Apache OODT
Workflow Execution (from Apache OODT)	WINGS, Repository, Organic Data Science Wiki
Workflow Instance (from Pegasus/Condor)	WINGS, Repository, Organic Data Science Wiki
Workflow Execution (from WINGS)	Repository, Organic Data Science Wiki, FragFlow, WExp, Prov-o-viz
Workflow Template (from WINGS)	Repository, Organic Data Science Wiki, FragFlow, WExp, Prov-o-viz
Workflow Template (from LONI)	Repository, Organic Data Science Wiki, FragFlow, WExp, Prov-o-viz

Table 1: Workflow use and consumption in WEST

Since the output of a workflow tool is consumed by several others, it is more likely that the representation used to exchange the workflows is not dependent of the specifics of two tools doing a pair wise exchange.

4.5 Abstraction Heterogeneity in WEST

The applications of the workflow environment have different needs. For example, mining and presentation applications typically care for workflow templates or workflow executions and their provenance, while execution engines need the workflow instances for their execution. In this Section we illustrate with an example how to access the contents of the same workflow by retrieving different artifacts at different granularities, showing the output after being consumed by different tools. All the workflow contents being referenced are publicly available online[18].

To illustrate our approach, we have selected one of the workflows stored in the WINGS repository for water metabolism estimation[19]. This workflow performs an analysis over sensor data collected in a river over a period of time as input, and plots the results. In order to illustrate our queries over the workflow, we will use the following namespace prefixes through the section:

```
@prefix prov: <http://www.w3.org/ns/prov#>.
@prefix opmv: <http://purl.org/net/opmv/ns#>.
@prefix opmo: <http://openprovenance.org/model/opmo#>.
@prefix p-plan: <http://purl.org/net/p-plan#>.
@prefix opmw: <http://www.opmw.org/ontology/>.
```

[18] http://opmw.org/sparql

[19] http://www.opmw.org/export/resource/WorkflowTemplate/AQUAFLOW_NTM

```
@prefix wfprov: <http://purl.org/wf4ever/wfprov#/>.
```

Since we are interested in showing how the template and execution traces are linked at the different levels of granularity, we first retrieve the available executions with the following query (urls and namespaces have been omitted for simplicity, the complete queries are available online[20]):

```
SELECT DISTINCT ?execution WHERE {
  ?execution  opmw:correspondsToTemplate  <url Aquaflow>.
}
```

This query returns two results, which correspond to the different executions of the workflow. Each execution corresponds to an OPM Account or a PROV Bundle (stored as a named graph), so we are able to retrieve the contents of the workflow in any of those vocabularies. For example, if we want to retrieve all the prov:Activities contained in the template, we might issue the following query:

```
SELECT DISTINCT ?activity
FROM <executionAccountURI> #Named graph for the Bundle
WHERE {
  ?activity a prov:Activity.
}
```

Similarly, if we want to retrieve all the OPM Processes within the workflow execution (which are the same as the prov:Activities), the query would be as follows:

```
SELECT DISTINCT ?activity WHERE {
 ?activity a  opmv:Process.
 ?activity opmo:account <executionAccountURI>.
}
```

This facilitates retrieving the contents of the workflow in a flexible way, as applications consuming PROV or OPM are capable of consuming the contents of the workflow without additional transformations. An example can be seen in Figure 7, where the Prov-o-viz tool (developed by another institution) has been used to show part of one of the water metabolism workflow executions. The tool consumes PROV, and accepts SPARQL endpoints as input. No additional format conversions are necessary for producing the diagram.

Other applications consume the fine grained representation of the workflow, expressed in OPMW. Figure 8 and Figure 9 show an example of two different applications (Workflow Explorer (WExp) and the Organic Data Science wiki respectively) showing details of the template metadata (Figure 8) and the template steps and variables (Figure 9). In each case, different metadata considered of value to the user is shown, but the representation and queries used to obtain the metadata are the same.

The repositories of the workflow environment may not always contain all the different levels of granularity for all workflows, as some of the information is redundant. For example, if the workflow is represented in OPMW, having the OPM, PROV and P-PLAN representation is good for flexibility and facilitating querying the repository, but when the number of workflows stored is high (e.g., several thousands of workflows and their executions), it might introduce scalability issues. In our case, we have decided to include the OPM and PROV representations because many applications are designed to work with them already, but we haven't done the same for P-plan. However, obtaining a P-PLAN representation is trivial with SPARQL construct queries and the documentation of the ontology. For example, the following query provides the P-Plan step dependency graph of the water metabolism workflow by querying the workflow ecosystem repository:

```
CONSTRUCT{
  ?activity2 p-plan:isPrecededBy ?activity.
}
WHERE{
 ?activity a opmw:WorkflowTemplateProcess.
 ?activity2 a opmw:WorkflowTemplateProcess.
 ?activity opmw:isStepOfTemplate <url AQUAFLOW_NTM>.
 ?activity2 opmw:isStepOfTemplate <url AQUAFLOW_NTM>    .
 ?activity2 opmw:uses ?u1.
 ?u1 opmw:isGeneratedBy ?activity.
}
```

The results show the dependency between the activities of the workflow, which is simpler than the usage/generation visualization adopted by the model.

As we introduced in the Related Work section, there are other vocabularies that use or extend PROV for scientific workflow execution and template representation (e.g., D-PROV [22] and the RO model [2]). In a workflow ecosystem each tool or application might have its own inner representation, and we find crucial for it to be able to interchange it with other systems. The advantage of the vocabularies extending PROV is that although the PROV serialization is often not exposed (only representations with the extended classes and properties are normally available) it is trivial to transform it to the standard representation. For example, the following construct query transforms Wfprov (RO model) to PROV:

```
CONSTRUCT{
 ?activity a prov:Activity.
 ?entity a prov:Entity.
 ?entity2 a prov:Entity.
 ?entity2 prov:wasGeneratedBy ?activity.
 ?activity prov:used ?entity.
}
WHERE{
 ?activity wfprov:usedInput ?entity.
 ?entity wfprov:wasOutputFrom ?activity.
}
```

With this PROV representation, the serialization would be compatible with the majority of the applications in WEST. Furthermore, with a similar query we could transform OPMW to Wfprov, and benefit from the applications exploiting it.

The template level transformation is a bit more complex, as different systems have different capabilities. For instance, some workflow systems like Kepler allow for loops and conditionals in their templates, while others do not. However, most of the

[20] http://purl.org/net/works2014materials

scientific workflows are modeled as Directed Acyclic Graphs (DAGs) and thus can be easily represented in P-PLAN with construct queries as we have shown in the previous examples.

5. DISCUSSION

Standard models for the representation of workflow executions as provenance (OPM, PROV) have been proposed by the community, but there are currently no standards for representing scientific workflow templates or instances. Several languages have been proposed. The IWIR syntax defines a representation language for workflows to be interoperable across different tools. The aim is to be able to take workflows designed in a tool and execute them in another one. IWIR has a similar function to P-Plan and OPMW, but IWIR is much more expressive than those languages because it has to cope with all the requirements of the different execution engines. In our work, we use P-Plan and OPMW, which have a less expressive but simpler interchange format. This is a feature but also a limitation, as it means that our approach is able to represent all scientific workflows modeled as DAGs, but cannot represent loops and conditionals, which are typical in business workflow languages such as WS-BPEL. The RO model [2] and D-PROV [22] are other workflow models for representing workflow templates and executions, and they also extend PROV. We use P-PLAN and OPMW because of their simplicity. OPMW and P-PLAN are sufficient for representing the workflow templates, instances and executions consumed by the workflow tools integrated in our ecosystem. In spite of this diversity, all these vocabularies extend PROV. Because P-PLAN and OPMW are effectively a subset of the constructs of IWIR, D-PROV, and the RO model, extracting workflows in those languages from our P-PLAN and OPMW representations amounts to making queries that return the representations desired, as we showed in Section 4.5.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have shown our approach towards creating an ecosystem of interoperable applications for scientific workflows. In order to achieve this purpose, we have extended OPM and the PROV standard with the P-plan model for representing workflow templates and the OPMW vocabulary to represent workflow templates and executions. We have also shown how to access and retrieve different information about workflows depending on what is needed by the different workflow tools.

In our choice for standard vocabularies, we have been driven by simplicity in the design rather than completeness. Other complex solutions (WS-BPEL, BPMN, D-PROV, etc.) can be adopted or extended for this purpose and easily mapped or combined with our model, as we have discussed in Section 4.5.

The main benefits of our approach are the interoperability among the applications in the ecosystem, the flexibility to interchange data and facilitating the integration of contents modeled in other vocabularies.

We have tested our approach by developing an ecosystem with applications created by different organizations and showing how the output produced by some of these tools can be consumed by others by means of SPARQL queries, which are also provided together with this paper. Our approach does not require that the applications in the ecosystem change their internal format representations. The used models are an interchange format for interoperability.

Regarding future work, we plan to continue developing converters for tools that are already integrated in the WEST workflow ecosystem but do not yet use the proposed models (depicted in dashed lines in Figure 1). This will greatly facilitate the addition of new workflow tools to the ecosystem. Another area of future work is to explore further converters for interoperability with other architectures and other models for representing workflows. Finally, an important aspect of future work is to address the scalability of workflow ecosystems through the development of common functions and APIs that will facilitate the rapid integration of new workflow systems and capabilities.

7. ACKNOWLEDGEMENTS

This work has been supported in part by the Spanish Science and Innovation Ministry (MICINN) with an FPU grant (Formación de Profesorado Universitario), and in part by the by Ministerio de Economía y Competitividad (Spain) under the project "4V: Volumen, Velocidad, Variedad y Validez en la Gestión Innovadora de Datos" (TIN2013-46238-C4-2-R) and the US Air Force Office of Scientific Research (AFOSR) with grant number FA9550-11-1-0104.

8. REFERENCES

[1] Abouelhoda, M.; Issa, SA. and Ghanem M. Tavaxy: Integrating Taverna and Galaxy workflows with cloud computing support. BMC bioinformatics 13: 77. 2012.

[2] Belhajjame, K.; Corcho,O.; Garijo, D.; Zhao, J.; et al. Workflow-Centric Research Objects: First Class Citizens in Scholarly Discourse. In SePublica2012 workshop at ESWC2012, 2012.

[3] Callahan, S. P.; Freire, J.; Santos, E.; Scheidegger, C. E.; Silva, C. T. & Vo, H. T. Vistrails: Visualization meets data management In ACM SIGMOD, ACM Press, 2006, 745-747

[4] Couvares, P.; Kosar, T.; Roy, A.; Weber, J.; Wenger, K. Workflows for e-Science. Springer, New York (2007) Ch. Workflow Management in Condor, pp. 357–375

[5] Deelman, E.; Singh, G.; Su, M.; Blythe, J.; Gil, Y.; Kesselman, C.; Kim, J.; Mehta, G.; Vahi, K.; Berriman, G. B.; Good, J.; Laity, A.; Jacob, J. C. & Katz, D. S. Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems *Scientific Programming,* 2005*, 13.*

[6] De Roure, D.; Goble, C.; Stevens, R, The design and realization of the myexperiment virtual research environment for social sharing of workflows, Future Generation Computer Systems 25 (5) (2009) 561–567.

[7] Dong Huynh, T.; Groth, P.; Zednik, S. PROV implementation report. W3C Working Group Note. 30 April 2013.

[8] Garijo, D. and Gil, Y. Augmenting PROV with Plans in P-PLAN: Scientific Processes as Linked Data. In Proceedings of the 2nd International Workshop on Linked Science 2012, Boston, 2012

[9] Garijo, D. and Gil, Y. A new approach for publishing workflows: abstractions, standards, and linked data. Proceedings of the 6th workshop on Workflows in support of large-scale science. 2011. 47-56.

[10] Garijo, D.; Corcho, O. and Gil, Y. Detecting common scientific workflows using templates and executing provenance. In Proceedings of the seventh international conference on Knowledge capture (K-CAP), 33-40. 2013.

[11] Gil, Y.; Ratnakar, V.; Kim, J.; González-Calero, P.; Groth, P.; Moody, J.; Deelman, E. Wings: intelligent workflow-based design of computational experiments, IEEE Intelligent Systems 26 (1) (2011) 62–72.

[12] Gil, Y.; Gonzalez-Calero, P. A.; Kim, J.; Moody, J.; and Ratnakar, V. A Semantic Framework for Automatic Generation of Computational Workflows Using Distributed Data and Component Catalogs. Journal of Experimental and Theoretical Artificial Intelligence, 23 (4), 389-467 2011.

[13] Gil, Y. Mapping Semantic Workflows to Alternative Workflow Execution Engines. Proceedings of the Seventh IEEE International Conference on Semantic Computing (ICSC), Irvine, CA, 2013.

[14] Goderis, A.; Fisher, P; Gibson, A.; Tanoh, F.; Wolstencroft, K.; De Roure, D.; Goble, C. Benchmarking workflow discovery: a case study from bioinformatics. Concurrency and Computation: Practice and Experience 21(16): 2052-2069 (2009).

[15] Groth, P. and Moreau, L. PROV-Overview. W3C Working Group Note. April, 2013

[16] Hoekstra, R. and Groth, P. PROV-O-Viz - Understanding the Role of Activities in Provenance. To Appear in the Proceedings of the International Provenance and Annotation Workshop, Cologne (2014)

[17] Krefting, D.; Glatard, T.; Korkhov,V.; Montagnat, J; and Olabarriaga, S. Enabling grid interoperability at workflow level. Grid Workflow Workshop 2011, 2011.

[18] Lebo, T.; Sahoo,S.; McGuinness, D.; Belhajjame, K.; Cheney, J.; Corsar, D.; Garijo, D.; Soiland-Reyes, S.; Zednik, S.; Zhao, J. Prov-o: The prov ontology, W3C Recommendation, 30th April 2013.

[19] Ludäscher, B.; Altintas, I.; Berkley, C.; Higgins, D.; Jaeger, E.; Jones, M.; Lee, E. A.; Tao, J. & Zhao, Y. Scientific workflow management and the Kepler system Concurrency and Computation: Practice and Experience, 2006, 18, 1039-1065

[20] Mates, P.; Santos, E.; Freire, J. & Silva, C. T. CrowdLabs: Social Analysis and Visualization for the Sciences 23rd International Conference on Scientific and Statistical Database Management (SSDBM), Springer, 2011, 555-564

[21] Miles, S.; Deelman, E.; Groth, P.; Vahi, K.; Mehta, G.; Moreau, L. Connecting Scientific Data to Scientific Experiments with Provenance. Proceedings of IEEE International Conference on eScience 2007: 179-186

[22] Missier, P.; Dey,S.; Belhajjame, K.; Cuevas-Vicentín, V. and Ludäscher, B. D-PROV: extending the PROV provenance model with workflow structure. Computing Science, Newcastle University, 2013.

[23] Montagnat, J.; Isnard, B.; Glatard, T.; Maheshwari, K. and Fornarino, M. B. A data-driven workflow language for grids based on array programming principles. In Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science (WORKS '09). ACM, New York, NY, USA. Article 7, p 1-10. 2009.

[24] Moreau, L.; Clifford, B.; Freire, J.; Futrelle, J.; Gil, Y.; Groth, P.; Kwasnikowska, N.; Miles, S.; Missier, P.; Myers, J.; Plale, B.; Simmhan, Y.; Stephan, E.; and denBussche, J. V. The Open Provenance Model Core Specification (v1.1). *Future Generation Computer Systems*, 27(6). 2011.

[25] Wolstencroft, K.; Haines, R.; Fellows, D.; Williams, A.; Withers, D.; Owen, S.; Soiland-Reyes, S.; Dunlop, I.; Nenadic, A.; Fisher, P.; Bhagat, J.; Belhajjame, K.; Bacall, F.; Hardisty, A.; Nieva de la Hidalga,A.; Balcazar Vargas, M.; Sufi, S.; Goble, C. The Taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud, Nucleic Acids Research (2013).

[26] Scheidegger, C.E.; Vo, H. T.; Koop, D.; Freire, J.; Silva, C.T. Querying and re-using workflows with vistrails, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD'08, ACM, New York, NY, USA, 2008, pp. 1251–1254

[27] Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y.; Groth, P.; Kwasnikowska, N.; Miles, S.; Missier, P.; Myers, J.; Plale, B.; Simmhan, Y.; Stephan, E.; Van den Bussche, J.;, The Open Provenance Model core specification (v1.1), Future Generation Computer Systems, Volume 27, Issue 6, June 2011, Pages 743-756, ISSN 0167-739X.

[28] Plankensteiner, P.; Montagnat, J. and Prodan, R. IWIR: a language enabling portability across grid workflow systems. In Proceedings of the 6th workshop on Workflows in support of large-scale science (WORKS '11). ACM, New York, NY, USA, 97-106. 2011.

[29] Starlinger, J., S. Cohen-Boulakia and U. Leser. (Re)Use in Public Scientific Workflow Repositories. Proceedings of the 24th international conference on Scientific and Statistical Database Management, p 361-378. 2012.

[30] Torri, F. ; Clark, A.P.; Dinov, I.D.; Zamanyan, A.; Hobel, S.; Genco, A.; Petrosyan, P.; Liu, Z; Vawter, M.P.; Eggert, P.; Pierce, J.; Knowles, J.A.; Ames, J.; Kesselman, C.; Toga, A.W.; Potkin, S.G. and Macciardi F. 2012 Next generation sequence analysis and computational genomics using graphical pipeline workflows. Genes, 3:545-575