Ontology-based Semi-automatic Workflow Composition

Daniel de Oliveira[1,2], Eduardo Ogasawara[1,2], Jonas Dias[1], Fernanda Baião[3], Marta Mattoso[1]

[1] COPPE/UFRJ, Brazil
[2] CEFET/RJ, Brazil
[3] NP2Tec/UNIRIO, Brazil
{danielc, ogasawara, jonasdias, marta}@cos.ufrj.br, fernanda.baiao@uniriotec.br

Abstract. Due to the growing complexity of scientific workflows, it is important to provide abstraction levels to aid scientists to compose these workflows. By doing this, we isolate scientists from infrastructure issues and let them focus on their domain of expertise when composing the workflow. Although using abstract workflows is a first step, there are many open issues, such as the ones related to semantics. Adding semantics to abstract workflows enables the explicit representation of which activities can be linked to each other, or which activities are similar to each other. Existing approaches address either the representation of abstract workflows or using domain ontologies to add semantics to workflow activities, but not both. In the latter case, these approaches focus only on adding semantics to executable workflows, instead of abstract ones. This makes it difficult to group executable workflows into a common abstract representation in the conceptual level. This article proposes coupling a workflow ontology, named SciFlow, to an abstract workflow representation named Experiment Line and implemented in the GExpLine tool. This is a step towards semantic mechanisms, helping scientists to identify equivalent activities or grouping executable activities into one abstract activity with the same semantics.

Categories and Subject Descriptors: H. Information Systems [**H.2.8 Database Applications**]: Scientific databases

Keywords: Ontologies, Scientific Experiments, Scientific Workflows, Semantics

1. INTRODUCTION

In the last decade, the use of scientific workflows to model scientific experiments became a reality. A scientific experiment is characterized by the composition and execution of several variations of workflows [Mattoso et al. 2009]. These variations include changing input data, parameters, programs, or even a combination of all previous changes. This turns the management of a scientific workflow into a very complex task, especially in large scale scientific projects. Workflows are managed by Scientific Workflows Management Systems (SWfMS), which enable the specification and execution of a chain of executable activities represented by programs, and are responsible for enacting and controlling the workflow execution. There are innumerous SWfMS [Taylor et al. 2007], but they focus on managing the execution of a workflow in an isolated way, disregarding the relationship among executions of workflows variations.

When scientists model their experiments using workflows, one important problem they worry about is the composition process. Composing a scientific workflow is not a simple task. During composition, scientists structure and define the entire experiment, establish the logical sequence of activities, plan variations that have to be explored, and define types of input and output data for each activity. There are not many approaches for workflow composition [Mattoso et al. 2009]; currently, composition considers workflows in a low level of abstraction. This means that scientists are required to compose their workflows directly in the SWfMS specification language, and to design their workflows in terms of programs, which poses several limitations to scientists and may prevent them from planning and

The authors thank CNPq and CAPES for partially sponsoring this research.

controlling alternatives. Composition tasks, however, should ideally be carried out before defining a concrete workflow,*i.e.* a workflow specified using SWfMS language. Some initiatives to workflow composition encompass the representation of workflows in many abstract levels. This facilitates workflow composition, but this abstract representation is still an open issue [Shoshani 2009] [Ludascher 2009] [Gil et al. 2007]. Current limitations include: (i) the tacit knowledge of which activities can be linked to each other; (ii) lack of a standard vocabulary used in the abstract representation; (iii) lack of representation of which activities are equivalent to each other; and (iv) lack of representation of dependencies among activities.

These problems can be addressed by adding semantics to the composition process, using ontologies. Some existing approaches propose adding semantics to workflows by associating domain ontologies to the specification, but they associate ontologies to concrete workflows [Wolstencroft et al. 2007]. However, it does not help in grouping workflows that share the same algorithm or method, for example, since abstract concepts, such as algorithm or method, are not associated to abstract workflow representations. Therefore, workflow metadata is needed to represent the abstract workflow and to help workflow resource searching according to these constructs (activities, algorithms, methods) in higher levels of abstraction. According to [Gomez-Perez et al. 2004], task and domain ontologies are complementary when applied to a specific application or scenario. Domain ontologies model a specific domain, or part of the world. They represent the particular meanings of terms as they apply to that domain. On the other hand, a task ontology describes the vocabulary related to a generic task, independent of its domain of application. Workflow composition in a specific scientific domain (*e.g.*, Bioinformatics or Oil exploitation) requires the ontology to include both domain-specific terminology and workflow composition concepts, thus providing semantics for scientists. Associating the ontology with abstract representations provides the following benefits: (i) controlled vocabulary that formalizes domain terminology, which is also coupled to abstract representation of activities; (ii) a checking mechanism to verify program compatibilities prior to execution time; (iii) formalization of knowledge related to which activities can be linked to each other; (iv) verification of similarity among activities (if they perform the same conceptual role in the experiment).

This article proposes associating ontologies to abstract workflow representation. We propose the use of SciFlow ontology [Oliveira et al. 2009] coupled to Experiment Line [Ogasawara et al. 2009] abstract representation to improve abstract workflows composition before using a SWfMS. SciFlow was incorporated into GExpline [Oliveira et al. 2010]. GExpline is an Experiment Line composition tool that guides scientists in composing the abstract workflow and maps each activity of the workflow to the chosen format of a SWfMS for workflow execution. By using SciFlow in GExpline in a real scenario we observed the four previously mentioned benefits of coupling workflow ontologies to abstract representations. This article is organized as follows. Section 2 presents SciFlow. Section 3 explains the concepts of Experiment Lines and GExpLine tool. Section 4 discusses the advantages and benefits of coupling ontologies to abstract representations of scientific experiments and presents the proposed approach. Section 5 presents a case study for evaluating SciFlow coupled to GExpline. Section 6 discusses related work. Finally, in section 7, we conclude this article and point to future work.

2. SCIFLOW: A SCIENTIFIC WORKFLOW ONTOLOGY

This section presents SciFlow [Oliveira et al. 2009], an ontology for scientific workflows. SciFlow represents a generic model that includes the main concepts and axioms related to scientific workflows. Scientists may specialize SciFlow ontology and the domain concepts in order to represent a specific scientific experiment. SciFlow models the workflow ontology in two levels: a super class level and a domain specific level. The super class level contains classes that represent general concepts that are shared by many scientific domains, while the domain specific level is composed by classes that should be specialized by scientists (or ontology engineers) for each scientific domain. Scientists need to specialize the super classes with domain terminology, without concerning about scientific workflow concepts

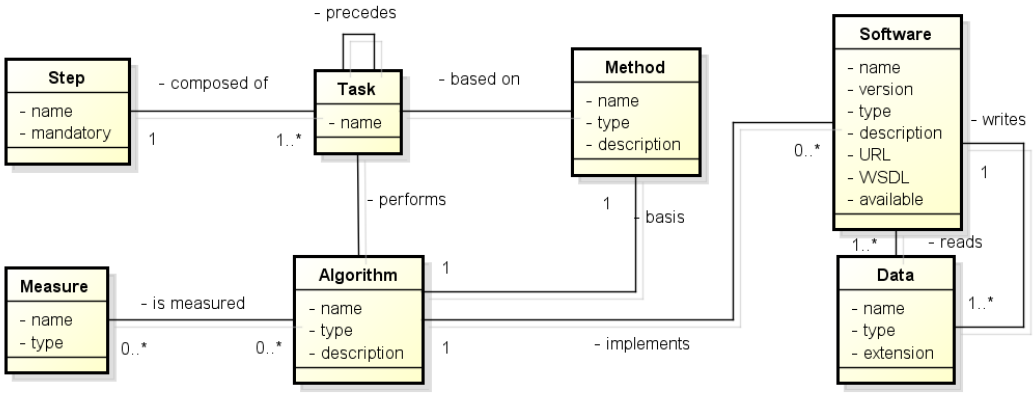

Fig. 1. An excerpt of the SciFlow ontology

already modeled. SciFlow was implemented in OWL using Protégé. Based on scientific workflow definitions, some concepts are extracted and modeled as super classes in SciFlow. SciFlow extends the DAMON ontology [Cannataro and Comito 2003] taking into account workflows requirements, such as input and output data and chaining of tasks. Its key concepts are: (i) *Step*: it represents a macro activity of the workflow, and models its highest abstraction level; (ii) *Task*: it is a specialization of Step and defines a specific problem that is being solved; (iii) *Method*: it is a methodology (or scientific model) used as a basis for an Algorithm; (iv) *Algorithm*: it is how a Task is performed, and is based on a specific Method; (v) *Software*: it is an implementation of a particular Algorithm. This is an important concept to our proposal, since the SciFlow ontology may be used to guide the user during the composition of abstract workflows and concrete workflow derivation. It complements the computational environment (SWfMS) in which the concrete workflow definition is manipulated and essentially deals with logical sequences of executable programs; (vi) *Measure*: it is a representation of how an Algorithm performance is measured; (vii) *Data*: it is the semantic representation of an input or output of any Software. Each Software consumes and produces a specific type of Data.

Figure 1 shows part of the SciFlow ontology, in the UML class diagram notation. Axioms are defined for the classes *Step, Task, Algorithm, Method, Measure, Software* and *Data*, to represent semantic statements, such as: a step Si always precedes a step Sj if there is a task Ti that precedes a task Tj, Ti follows algorithm Ai, Ai is implemented by software Swi, Swi outputs data Di. Many important properties were defined for the super classes as well. For instance, a property "Available" of Software indicates whether it is available for use or not. Other examples we can cite are: the relationship "Precedes" that indicates which Step or Task precedes another in a logical order. Although SciFlow is a step forward, it does not consider actors - and software is the only kind of (non-human) actor allowed.

3. EXPERIMENT LINES AND GEXPLINE

An Experiment line [Ogasawara et al. 2009] is an approach to represent a scientific experiment. It may be defined as an abstract workflow that is capable to be derived into multiple workflows at concrete (executable) level. Each activity in an Experiment Line behaves like an independent component. Each abstract activity of the flow may be implemented by a list of compatible sequences of concrete activities. Also, a sequence of abstract activities may be grouped together to form another abstract activity. When an abstract activity has more than one sequence of concrete activities to represent its behavior, it is called a variant activity. Actually, this means that an abstract activity is a variant activity if it has more than one program to implement its conceptual component behavior. Also, when

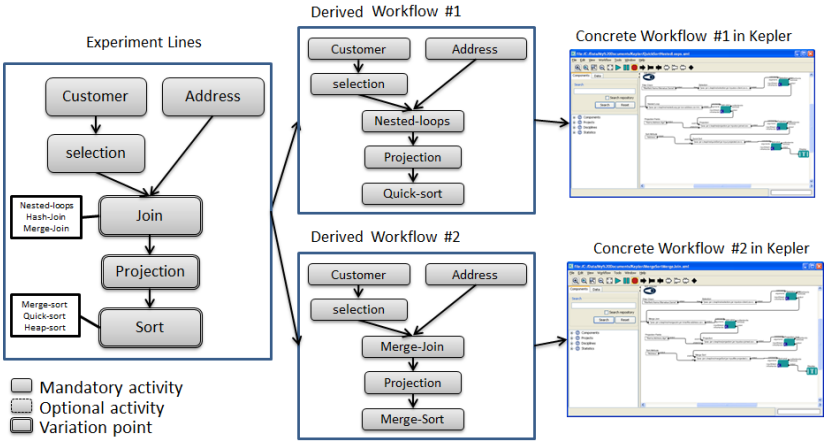

Fig. 2. An example of an experiment line derivation

an abstract activity may be suppressed from the experiment line, it is defined as an optional activity. A mandatory activity is an abstract activity that must be used in all derived concrete workflows.

An example of derivation in experiment lines is presented in Figure 2. This experiment (a common database query represented as a workflow) is composed by six main abstract activities (Customer, Address, Selection, Join, Projection and Sort). In this experiment line, the Sort activity of the flow is optional which means that we can derive a concrete workflow that does not contain this activity. The Join and Sort activities of the flow are variant points, which means that we have options to choose when deriving a concrete workflow. Although an experiment line is designed to derive many concrete workflows through its optional, mandatory and variant activities, it does not provide important metadata, such as algorithms or methods that are associated with each activity of the abstract workflow.

For example, in Figure 2, we have variant point named Join. In this case Nested-loops, Hash-Join and Merge-Join are variations of Join activity. However, using this representation, scientists are not aware of which algorithms or methods are associated to those activities. Although, metadata and free-text description are an option to provide more information about the experiment line, according to [Gomez-Perez et al. 2004], the use of free text limits search capabilities and automatic comparisons and generates a non-uniformity of the terms, since each scientist may describe the same concepts in different ways. Metadata without navigation or inference are difficult to be used. In most experiments, this metadata is fundamental. Another issue observed with the experiment lines is the lack of compatibility between activities. During the derivation process, some types of activities may demand the selection of other activities to execute. Also, the selection of another activity may inhibit the selection of other types of activities. Here, ontologies play an important role, in helping scientists during derivation process through its inference mechanisms.

GExpLine is a tool for supporting experiment lines composition. GExpLine is complementary to existing SWfMS, thus offering extra representation layers to be coupled to the existing workflow systems. Its goal is to provide a high abstraction level representation environment for scientists to model their scientific experiments and to automatically generate corresponding executable workflows in pre-defined SWfMS specification languages. The GExpLine tool is based on five main components: (i) Experiment Line Modeler: designs experiment lines, (ii) Derivation: derives concrete workflows based on abstract ones. The derivation process is based on the concept of cartridges, *i.e.* when scientists are deriving their conceptual workflows (represented in our object model) into concrete ones they have to choose one cartridge in a set of available cartridges, where each cartridge of this set generates concrete workflows for different representation language and also in XPDL which is agnostic

from SWfMS, (iii) Import: imports concrete workflows from Kepler [Altintas et al. 2004], Taverna [Hull et al. 2006], and VisTrails [Callahan et al. 2006]. This import process allows for scientists to create experiment lines based on existing workflows that were modeled on the fly; (v) Query: queries prospective provenance data using abstract/concrete information. The GExpLine is ruled by its conceptual model that defines the experiment, abstract and concrete workflows, workflow components such as ports and relations, version identification and organization, as well as operations for retrieving existing versions and constructing new versions (white and green classes in Figure 3). The GExpLine model is composed by two main parts: (i) workflow classes, in which workflow concepts are represented; and (ii) version classes, which represent the way that versions are organized. Figure 3 presents the GExpLine model as UML class diagram. Classes were colored differently since they represent different perspectives on the model. The white classes, which are Experiment, Workflow, Activity, Relationship, Port, AbstractActivity, ConcreteActivity, Derivation and MetaArtifact represent the workflow classes, which is actually the workflow meta-model. Finally, green classes, which are Version, Transaction, ConfigurationItem, User, and Project, represent the version classes. Workflow classes represent the product space, while version classes are called the version space. The linking between the product space and version space is defined by the VersionedElement class, which acts as an interface between both spaces. All classes that are part of the workflow meta-model just need to inherit from the VersionedElement class to be versioned and managed by the GExpLine configuration mechanism.

In GexpLine, a workflow (class Workflow) is composed of activities (class Activity) and relationships (class Relationship). The class Workflow may be specialized into conceptual abstract workflows (class AbstractWorkflow) and concrete workflows (class ConcreteWorkflow). An activity in a workflow has input and output ports (class Port). The relationship between activities is a directed edge that establishes the dependency between activities and also defines the workflow activity chaining. In addition, the class Activity presents a self-relationship that indicates variability, *i.e.* the choices that scientists make when modeling a workflow. There are two specializations for the Activity class. The AbstractActivity class represents activities modeled in the abstract level while ConcreteActivity represents activities modeled in the executable level. Both AbstractActivity and ConcreteActivity inherit from the class Activity. Each Activity that is part of a Workflow produces and consumes a specific MetaArtifact. A MetaArtifact is a type of artifact in a prospective provenance model. A generated artifact obtained during workflow execution is actually an instance of a MetaArtifact. The derivations performed by scientists are registered in the class Derivation. This way, the activities and workflows derived are registered and provide important information for future backtracking information.

Although an experiment line is designed to derive many concrete workflows, it presents the same problems of any abstract representation (already presented in Section 1), since it does not provide: a controlled vocabulary to be used, a representation of activities that are equivalent to each other and dependencies among activities. For example, when scientists have more than one concrete activity (a program, a service or a script) that performs the same abstract activity, the scientists need to manually investigate to determine which alternative is the most suitable. However, this investigation can be laborious. Scientists need more support on workflows composition and analysis of the experiment. In next section we present the improvements implemented on GExpLine to couple SciFlow to the abstract representation.

4. COUPLING WORKFLOW ONTOLOGIES TO EXPERIMENT LINES

Scientific workflow composition has many difficulties. One of the difficulties is how to represent the experiment while moving along different levels of abstraction (abstract and concrete). Domain ontologies provide semantic support to workflow activities and data. Combining domain and workflow ontologies with experiment lines provides a flexible representation mechanism and allows for navigating through hierarchical levels (concrete and abstract). These characteristics can help the composition process, for example, when looking for a specific abstract level component or when searching for components

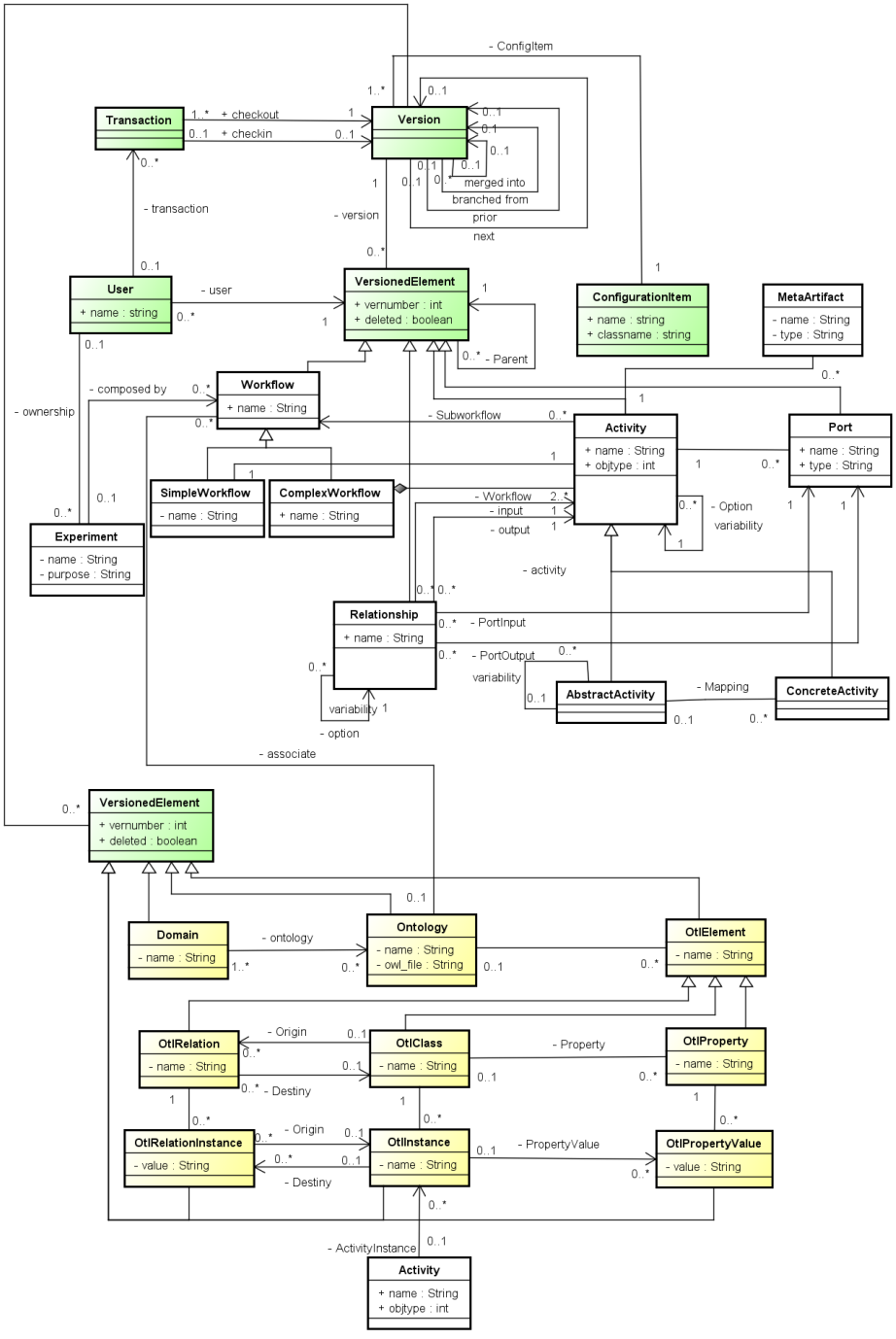

Fig. 3. Conceptual model for experiment lines

that are (or may be) used in sequence. Scientific methods and algorithms can be represented along the composition of the experiment line, independent from technological issues. It also helps concrete workflow derivation. Based on an abstract definition, corresponding executable resources can be found to derive the workflow at the concrete level, while still being independent from the SWfMS. Here, the SciFlow ontology plays a fundamental role during concrete workflow derivation by restricting the set

of possible programs that may be chosen by the workflow designer to implement each corresponding activity in the experiment line. Only programs associated to the ontology classes of the abstract workflow are available to be included into the concrete workflow, avoiding rework and misunderstandings. Since there may be several programs, algorithms and methods available, the ontology also helps in exploring the alternatives. In summary, the proposed combination of SciFlow with experiment lines in GExpLine provides the following facilities to scientists: (i) a controlled vocabulary to be coupled to abstract workflows: activities of an abstract workflow may be associated with ontology terms by scientists. The terms associated may be methods, algorithms, measures, depending on the ontology that is imported to the abstract representation; (ii) a checking mechanism to verify program compatibilities: through inference on the ontology, it is possible to identify which data type is related to each software component input and output. With this resource it is possible to type check workflow definitions, thus avoiding incompatible chaining; (iii) formalization of all knowledge related to which activities can be linked to each other: all the knowledge related to the experiment is structurally represented in the ontology; (iv) verification of similarity between activities (if they perform the same conceptual role in the experiment): in most scientific experiments it is necessary to identify which activity of an abstract workflow is equivalent to another. This is possible by looking for all activities which perform the same step in the workflow. This may be useful when a scientist knows the algorithm he/she wants to execute, but does not know (or does not care of) which software to choose that implements the algorithm. In this case, the ontology should be able to offer a list of available software that implements the specific algorithm. Additionally, SciFlow supports the derivation process from abstract to concrete workflows. Relationships between software and its method, algorithms and their associated measures, are explicitly represented. When scientists derive a concrete workflow based on an abstract representation, it is possible to discover programs that can be derived from a specific abstract activity through reasoning from a particular method or algorithm. (v) verification of dependency between activities: the ontology provides ways to identify whether an activity depends on another and to guide the scientific workflow specification.

SciFlow also allows for ontology-based semantic provenance registry. Data provenance is the process of storing and sharing information about the origin of each data generated by the workflow [Freire et al. 2008]. SciFlow can help scientists track workflow definitions using domain terminology and semantic relationships between concepts. The ontology data may be stored in a provenance schema along with data of workflow definitions, thus enabling to semantically track composed workflows at several levels of abstraction. Consider that an algorithm follows a significant scientific method and that is implemented by programs P1, P2 and P3. Current provenance support is based on the concrete level only. Thus, when scientists want to know which executed workflows employed some algorithm, they must know there are three available programs and must write queries that search workflows using P1, P2 and P3. SciFlow coupled to GExpLine provenance schema can solve queries at more abstract levels (methods, algorithms or programs) through inference capabilities. Browsing provenance data with high levels of abstraction also helps composition with workflow reuse, answering questions such as: What Task does the Software A perform? What kind of Method is used by Software B? What types of Data does the Software C handle? What Algorithm does Software D implement?

In order to couple SciFlow (modeled in OWL) with GExpLine, some improvements had to be made on GexpLine's structure. The first improvement was related to the GExpLine model. It has been extended to comprise ontological concepts. As mentioned in Section 3 we must have a unified conceptual model to represent the abstract and concrete definitions along with ontological concepts. We extended the experiment line conceptual model (yellow classes in Figure 3) to create this unified model, including nine classes to map workflow ontology to a domain conceptual model. This mapping metamodel allows scientists to associate ontology concepts to workflow activities. Each one of those classes was then implemented as a table in the physical database schema, following the work of [Astrova et al. 2004]. In addition, the Workflow concept was specialized to SimpleWorkflows (which contain just one Activity) and ComplexWorkflows (which are composed of two or more "parts", that is,

Activities), to address the weak supplementation principle for conceptual modelling [Guizzardi 2011].

The Domain class represents the domain in which the ontology is inserted, the Ontology class represents the ontology itself (and its general properties like name, file name, and so on). The OtlElement class represents all of the concepts that exist in the ontology. These concepts may be classes (modeled by OtlClass), properties (modeled by OtlProperty) and relations (modeled by OtlRelation). The other classes (OtlInstance, OtlPropertyValue and OtlRelationInstance) represent the individuals (instances of the ontology), its relationships and property values. Although the ontology concepts are present in the conceptual model, inference is still applied using an external reasoner (*e.g.* Pellet). Once the inference is completed, the concepts of the ontology are associated to a specific experiment line or abstract activity to be persisted in the repository (that follows the classes of the conceptual model), thus allowing future queries on experiment definitions.

Other improvements were made on the GExpLine tool to associate ontology concepts to the activities of the experiment lines, support semantic derivation and facilitated composition. They include importing OWL ontology to the GExpLine environment, thus making GExpLine aware of the ontology structure in order to control inference and reasoning and store the ontology concepts in the GExpLine model. A plug-in to parse the entire ontology and populate the model with ontology concepts was developed. This plug-in is based on the Protégé OWL API that implements a layer over JENA to handle OWL files. Also, a specific user interface was developed to import the OWL ontology to the model. In addition, in the GExpLine tool, the imported ontology is associated to the modeled experiment line, by associating an instance of the Ontology class to one instance of the Workflow class. In our current implementation we limited the association to a single ontology *per* experiment line. Ontology concepts are then associated to the activities. GExpLine enables reasoning on the ontology to find resources if any concept associated with an activity. For example, suppose that an abstract activity may be derived in many concrete activities. Ontology reasoning guides the scientist to find options and choose a variation to derive the concrete representation by adding "filters" using ontology concepts as criteria. Once the concrete activities are chosen and compatibilities are verified, a plug-in is invoked on GExpLine tool to export concrete scientific workflows modeled in many formats, including to Kepler [Altintas et al. 2004].

5. EVALUATING GEXPLINE WITH SEMANTICS ASSOCIATED

GExpLine was extended to support the experiment modeling with semantics associated, thus improving scientific workflow composition using the experiment lines concept. In this Section we evaluate if the scientists can take advantage of GExpLine with semantics to enhance the composition process. With these objectives in mind, we planned a study to obtain the scientist evaluation over three scientific workflow composition methods: From Scratch (FS) approach, in which users start composing a workflow from scratch using a SWfMS, Pure GExpLine (PG) approach, in which scientists model their workflows without semantics associated; and the GExpLine with Semantics (GS) approach, where scientists benefit from the advantages of SciFlow coupled to GexpLine. The main goal of this study is to evaluate how the experiment lines approach with semantics implemented by GExpLine can compose scientific workflows compared to traditional methods. We planned a study involving scientific workflows composition in the database domain. The experiment line to be modeled was the same presented in Figure 2. The case study was performed with graduation students and there were enough students able to fully understand the database workflow examples.

Our experimental study focuses on analyzing the scientist experience with different scientific workflow composition methods, in order to compare the methods considering their ease of use and time from the scientist point of view. Our primary hypothesis (H1) is that the composition with GExpLine and semantics is easier to use than PG approach and FS approach. The null hypothesis (H0) is that there is no difference in ease of use between the three studied approaches. Our study has also a secondary hypothesis that evaluate the time the scientists spend with the methods (H2) with its null

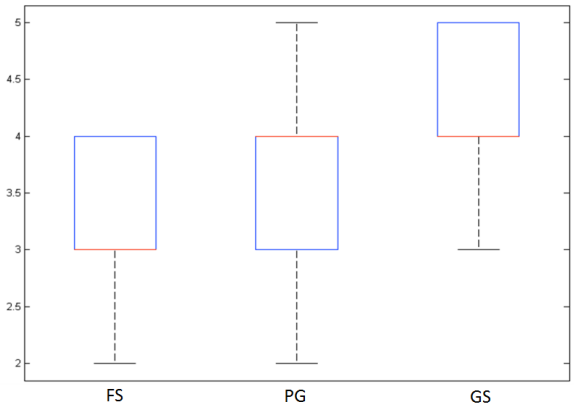

Fig. 4. The collected answers on the box plot form

hypothesis that is there is no difference in time between FS, PG and GS. Since our objective is to compare the PG and the FS approach with the GExpLine with semantics approach, our experiment is a one factor, two paired treatments study [Freedman et al. 2007].

The experiment was conducted in the context of a distributed and parallel database discipline at Federal University of Rio de Janeiro. There were 31 students participating in the experiment which give us a statistical significance. Five of them are undergraduate students and all others are master students. They answered a questionnaire about their experience with query processing and scientific workflow subjects. About 13 % of them answered that have low knowledge on query processing, 57 % answered that have intermediate knowledge and 30 % answered that have high or very high knowledge. All of them answered that have low or very low expertise in scientific workflows. The participants also received a document with instructions to compose scientific workflows using the three composition approaches (PG, FS, and GS). They received a detailed oral explanation about the documentation and had their doubts solved. They were instructed to execute each approach separately, registering time spent to complete the task of executing each approach. They should, then, proceed answering the questions regarding each approach. The collected answers were, then, analyzed to evaluate our hypothesis.

The questions regarding the easiness of use of each method were used to evaluate our primary hypothesis (H1). The participants worked on the three previously described approaches and rated their easiness to use according to a 5-point Likert scale [Freedman et al. 2007] set as: (1) very difficult; (2) difficult; (3) medium; (4) easy; and (5) very easy. The registered time was used to evaluate secondary hypothesis H2. Since all the questions were answered through Likert scales, we were able to do a quantitative statistic analysis of the samples [Freedman et al. 2007]. To evaluate our main hypothesis (H1), we needed to analyze the answers from three questions regarding the perception about the easiness of use of each usability method (PG, FS, and GS). Our goal is to compare the perception about FS method with GS and the PG method with GS to verify which one the user perceived as the easiest to use. Since we used the mean of the answers to conclude the easiest method according to its usability, we needed first to compare our paired samples with statistical hypothesis tests. We stipulated our significance level (a) at 5 %. The Shapiro-Wilk [Freedman et al. 2007] normality tests showed that we could not state that our samples are normal with the given a of 5 %. Thus, we are assuming our samples are non-parametric. We applied a paired two-sided Wilcoxon rank sum [Freedman et al. 2007] non-parametric test that statistically showed that the FS and GS have different medians with p-value å= 3.03390 X 10-6 and that the PG and GS approaches have different medians with p-value å= 1.7887 X 10-4. Figure 4 shows our collected data in the box plot form.

Our results reject the null hypothesis with great significance, which means that our paired samples

have different medians. Since the median of GS sample is higher than the other methods, it means that the users perceived the experiment lines with semantics method as the easiest one to handle scientific workflow composition. 17 % of the students felt that it was difficult to compose a workflow using only a SWfMS (Kepler), while the remaining 83 % rated the task between medium and easy. Modeling an experiment line without semantic support was rated between difficult and medium by 47 % of the students while 53 % rated it between easy and very easy. To evaluate the secondary hypothesis H2, we used the time values registered by each participant that informs how much time they spent to compose the asked workflow with each method. Again, the Shapiro-Wilk test showed that our samples are non-parametric. We applied the two-sided Wilcoxon signed rank test that statistically showed that the FS and GS samples have different medians with p-value å= 3.8210 X 10-5 and the PG and GS samples have different medians with p-value å= 4.8691 X 10-5. These results reject the null hypothesis with great significance, which means that the paired samples have different means. It is clear that participants built workflows faster using GExpLine with semantics. Note that students were used in the experimental study as subjects. However, according to [Svahnberg et al. 2008], using students as subjects in experimental studies is acceptable since the subjects execute specific tasks determined by the evaluator.

6. RELATED WORK

We analyzed existing SWfMS support for workflow representation in different levels of abstractions and the semantic support for the process of composing scientific workflows. Available SWfMS are restricted to concrete level. In this way, the knowledge of which activities can be linked to each other is still tacit. Currently, projects in many domains force scientists to redefine, almost from scratch, scientific workflows previously developed by other scientists, incurring in the same composition trial and error process. This occurs due to the absence of a systematic approach for composition and lack of abstract workflow representation. Some approaches use ontologies to help composing scientific workflows. One example is the TAMBIS [Stevens et al. 2000] project that aims to provide transparent access to biological databases and scientific analysis tools. It provides a knowledge driven interface where the scientist is able to compose experiments in terms of biological terminology. It focuses on a specific domain (biology) and does not represent generic workflow metadata. In the same direction, the approach proposed by [Wolstencroft et al. 2007] proposes an ontology based on a specific project.

[Altintas and Ludascher 2003] present an approach to relieve the scientists from designing directly executable workflows. It represents an abstract workflow based on directed acyclic graphs. It uses database mediation techniques that automatically map abstract workflow activities into executable ones. This mapping is powerful and independent of SWfMS. However, this approach does not add semantics to the abstract activity, thus identifying similar workflows that share a common ancestor is not possible. OWL-WS [Beco et al. 2005] is a semantic workflow representation model with a related language that is workflow ontology. The objective is to define a workflow language that enables a specification of dynamic workflows, which are composed of Grid Services and are used as evaluation and binding mechanisms in a Grid Service enactment engine. Although OWL-WS offers a meta-model for scientific workflows, it focuses on concrete workflow representation to model services, ports and infrastructure issues, lacking support for modeling a conceptual representation of a workflow in different levels of abstraction.

myGrid is a semantically rich approach [Wolstencroft et al. 2007] coupled to the Taverna SWfMS. It has an OWL ontology developed for service discovery through service annotation. This ontology is composed by two sub-ontologies: domain ontology and services ontology. The domain ontology models the bioinformatics domain and the services ontology models the function of Web services and their parameters inside the Taverna. Reasoning can be used to find common ancestors of activities between workflow definitions. However, since activity roles are not explicit it is not straightforward to find workflows that share the same method or algorithm. Also, it does not represent which activities

can precede one activity or data dependencies as in SciFlow. The authors did not find approaches that represent and comprise semantic support for different levels of abstraction in a generic way, decoupled from specific SWfMS or domains.

In the context of annotation of data sources in scientific experiments, scientific workflows are used to help the propagation of annotations [Bowers and Ludascher 2006]. In this article the ontology are not only used to generate an annotation, but to model and to execute the workflow. [Bowers and Ludascher 2006] take advantage of existing workflows to propagate annotations considering them as a set of logic rules. A careful discussion [Gil et al. 2007] about workflow challenges reinforces the lack on semantic support for such distributed systems.

7. CONCLUSIONS

In this article we propose an ontology-based approach to add semantics to abstract representations of workflows represented as Experiment Lines. We combined the SciFlow ontology to the GExpLine tool. This association enables scientists to add semantics of different levels of abstraction to this generic representation of the experiment line, so that they can focus on the concepts related to the experiment, instead of having to deal with infrastructure issues during the composition phase. Once the experiment is represented with all its desired variations, concrete workflows are generated to be executed by the SWfMS of scientists' choice. The ontology specialization helps scientists in understanding the scientific workflow domain and the processes that are being specialized in the SciFlow ontology.

Experimental results reinforce that experiment lines implemented in GExpLine with semantics provide benefits during the composition phase of scientific experiments, when compared to non-semantics ones. These benefits include explicit representation of variation points and optional elements with metadata associated, easiness of use and faster during workflow composition. Future work includes the development of additional structural verification mechanisms to enhance composition of experiments in GExpLine. GExpLine is part of the GExp project that aims at providing and developing Databases and Software Engineering techniques in the context of large scale science. More details and videos can be obtained at http://gexp.nacad.ufrj.br.

REFERENCES

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., and Mock, S. Kepler: an extensible system for design and execution of scientific workflows. In *Proceedings of the International Conference on Scientific and Statistical Databases Management*. Santorini, Greece, pp. 423–424, 2004.

Altintas, I. and Ludascher, B. Compiling abstract scientific workflows into web service workflows. In *Proceedings of the International Conference on Scientific and Statistical Databases Management*. Cambridge, USA, pp. 251–254, 2003.

Astrova, I., Kalja, A., Jaeger, E., Jones, M., Ludascher, B., and Mock, S. Storing OWL Ontologies in SQL3 Object-Relational Databases. In *8th WSEAS International Conference on Applied Informatics and Communications (AIC08)*. Rhodes, Greece, pp. 99–103, 2004.

Beco, S., Cantalupo, B., Giammarino, L., Matskanis, N., and Surridge, M. OWL-WS: A Workflow Ontology for Dynamic Grid Service Composition. In *Proceedings of the First International Conference on e-Science and Grid Computing (e-Science05)*. Melbourne, Australia, pp. 148–155, 2005.

Bowers, S. and Ludascher, B. A Calculus for Propagating Semantic Annotations Through Scientific Workflow Queries. In *Proceedings of the International Conference on Extending Database Technology*. Munich, Germany, pp. 712–723, 2006.

Callahan, S., Freire, J., Santos, E., Silva, C., Scheidegger, C., and Vo, H. T. VisTrails: visualization meets data management. In *Proceedings of the ACM SIGMOD International Conference on Management of Data Conference*. Chicago, USA, pp. 745–747, 2006.

Cannataro, M. and Comito, C. A Data Mining Ontology for Grid Programming. In *Workshop on semantics in Peer-to-Peer and Grid Computing*. Budapest, Hungary, pp. 113–134, 2003.

Freedman, D., Pisani, R., and Purves, R. *Statistics, 4th Edition*. Norton, 2007.

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

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., and Myers, J. Examining the Challenges of Scientific Workflows. *IEEE Computer* 10 (12): 34–32, 2007.

Gomez-Perez, A., Corcho, O., and Fernandez-Lopez, M. *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Verlag, 2004.

Guizzardi, G. Ontological Foundations for Conceptual Part-Whole Relations: The Case of Collectives and Their Parts. In *Procedings of CAiSE 2011*. London, United Kingdom, pp. 138–153, 2011.

Hull, D., Wolstencroft, K., Alper, P., Wroe, C., Lord, P., Stevens, R., and Goble, C. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research* 34 (1): 729–732, 2006.

Ludascher, B. What Makes Scientific Workflows Scientific? *Scientific and Statistical Database Management* 10 (3): 10–25, 2009.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Murta, L., Ogasawara, E., de Oliveira, D., Cruz, S. M. S., and Martinho, W. Towards Supporting the Life Cycle of Large-scale Scientific Experiments. *International Journal of of Business Process Integration and Management* 5 (1): 79–92, 2009.

Ogasawara, E., Paulino, C. E., Murta, L., Werner, C., and Mattoso, M. Experiment Line: Software Reuse in Scientific Workflows. In *Proceedings of the International Conference on Scientific and Statistical Databases Management*. New Orleans, USA, pp. 264–272, 2009.

Oliveira, D., Ogasawara, E., Baiao, F., and Mattoso, M. Using Ontologies to Provide Different Levels of Abstraction in Scientific Workflows. In *5th IEEE International Conference on e-Science*. Oxford, United Kingdom, pp. 1–10, 2009.

Oliveira, D., Ogasawara, E., Chirigati, F., Silva, V., Murta, L., and Mattoso, M. GExpLine: A Tool for Supporting Experiment Composition. In *Provenance and Annotation of Data and Processes*. Troy, USA, pp. 251–259, 2010.

Shoshani, A. The Scientific Data Management Center: Providing Technologies for Large Scale Scientific Exploration. In *Proceedings of the International Conference on Scientific and Statistical Databases Management*. New Orleans, USA, pp. 1–2, 2009.

Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N. W., Goble, C. A., and Brass, A. TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. *Bioinformatics* 16 (2): 184–186, 2000.

Svahnberg, M., Aurum, A., and Wohlin, C. Using students as subjects - an empirical evaluation. In *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. Kaiserslautern, Germany, pp. 288–298, 2008.

Taylor, I., Deelman, E., Gannon, D., and Shields, M. *Workflows for e-Science: Scientific Workflows for Grids*. Springer Verlag, 2007.

Wolstencroft, K., Alper, P., Hull, D., Wroe, C., Lord, P., Stevens, R., and Goble, C. The myGrid ontology: bioinformatics service discovery. *Bioinformatics* 3 (3): 303–325, 2007.