Adding Ontologies to Scientific Workflow Composition[1]

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

[1]Federal University of Rio de Janeiro – COPPE/UFRJ
[2]Federal Center of Technological Education – CEFET/RJ
[3]Federal University of the State of Rio de Janeiro – NP2Tec/UNIRIO
{danielc, ogasawara, marta}@cos.ufrj.br, Fernanda.baiao@uniriotec.br

Abstract. Scientific workflows are being used as an abstraction for the composition of large scale scientific experiments. As scientific workflows become more complex, these abstractions isolate scientists from infrastructure issues. Although representing a workflow in an abstract level 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 equivalent to each other. However, representing an abstract workflow with semantics is an open problem. Existing approaches address either the representation of abstract workflows or the use of domain ontologies to add semantics to activities, but not both. In the latter case, these approaches focus only on adding semantics to executable workflows, instead of working in different abstract levels. This makes it difficult to group workflows into a common abstract representation in the conceptual level. Common abstract workflow representation allows for a better understating of the experiment by scientists and enables the association of the experiment definition with domain knowledge. This paper proposes coupling workflow ontologies to abstract workflow representations. This provides the required semantic mechanisms to help scientists to identify equivalent activities, group executable activities into one abstract activity with the same semantics, and verify the compatibility between activities. We implemented and evaluated the proposed approach by coupling SciFlow – a workflow ontology - with GExpLine – an abstract workflow management tool. Experiments show the benefits of this approach.

Keywords: scientific experiment, ontologies, semantics.

1. INTRODUCTION

In the last decade, the effective use of scientific workflows to model large scale scientific experiments became a reality (Mattoso et al. 2010). A scientific experiment is characterized by the composition and execution of several variations of workflows (Mattoso et al. 2010). 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 usually managed by Scientific Workflows Management Systems (SWfMS), which enable the specification and execution of a chain of executable activities represented by programs or services, and are responsible for enacting, controlling and monitoring 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 between executions of workflows variations.

When scientists model their experiments using scientific workflows, one important problem they should worry about is the composition process. Composing a scientific workflow is not a simple task. During composition, scientists structure and define the whole experiment, establish the logical sequence of activities, plan variations that have to be explored, and define the types of input and

[1]This work was partially funded by CNPq and CAPES

output data for each activity. There are not many approaches for workflow composition (Mattoso et al. 2010); currently, composition considers workflows in a low (and fixed) 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 or services, which poses several limitations to scientists and may prevent them from planning and controlling alternatives. Composition tasks, however, should ideally be carried out before defining a workflow in the specific SWfMS language. Some initiatives to workflow composition encompass the representation of workflows in hierarchical abstract levels. This facilitates workflow composition, but this abstract representation is still an open, yet important, issue (Shoshani 2009, Ludäscher 2009). 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 (which may be very difficult to achieve with distributed and heterogeneous teams of scientists); (iii) lack of representation of which activities are equivalent to each other; and (iv) lack of representation of dependencies between activities.

These problems can be addressed by adding semantics to the composition process, through the use of ontologies. Some existing approaches propose to add semantics to workflows by associating domain ontologies to the specification, but they associate ontologies to concrete workflows (Wolstencroft et al. 2007). This enables ontology navigating from a concrete activity, but it does not help in grouping workflows that share the same algorithm or method, for example, since the abstract concepts of algorithm or method are not associated to abstract workflow activity roles. Therefore, workflow metadata is needed to represent the abstract workflow and to help workflow resource finding according to these roles 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. It represents 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 a more complete semantics for scientists. Associating the ontology with abstract representations provides: (i) controlled vocabulary that formalizes domain terminology, which is also coupled to abstract representation of activities; (ii) a checking mechanism to verify program and service compatibilities prior to execution time; (iii) formalization of knowledge related to which activities can be linked to each other; (iv) verification of equivalency between activities (if they perform the same conceptual role in the experiment); (v) verification of dependency between activities.

This paper proposes an approach to associate ontologies to abstract workflow representation. We propose the use of SciFlow ontology to improve abstract workflows composition before using a SWfMS. SciFlow is a task ontology for workflow composition that may be associated to domain ontologies to provide semantic facilities on scientific workflows. It relates the components of the workflow with domain terminology, associating roles in different levels of abstraction. These levels go down until the level of a SWfMS language workflow. SciFlow was incorporated into GExpline (GExp 2009). GExpline is an abstract workflow composition tool that guides the scientist in composing the abstract workflow and maps to the chosen concrete SWfMS for workflow execution. By using SciFlow in GExpline in a real scenario we observed the five benefits of coupling workflow ontologies to abstract representations, previously mentioned.

This paper is organized as follows. Section 2 presents SciFlow. Section 3 explains the advantages and benefits of coupling ontologies to abstract representation of scientific experiments. Section 4 presents the effective use of ontologies with an abstract representation and discusses the coupling of SciFlow to GExpline. Section 5 discusses the existing semantic support for scientific workflows. Finally, in section 6, we conclude this paper and point to future work.

2. SCIFLOW: A SCIENTIFIC WORKFLOW ONTOLOGY

This section presents a detailed view of SciFlow (Oliveira et al. 2009), an ontology for scientific workflow composition. 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 the specific scientific experiment being addressed. By the time the ontology is specialized, it may be referred to by all subsequent scientific workflow composition tasks of that domain. SciFlow models the template ontology in two levels: a super class level and a domain specific level. The first level has classes that represent general concepts used throughout the scientific domains, while the domain specific level is composed by classes that should be specialized by scientists (or ontology engineers) for each different scientific domain. Scientists need to specialize the super classes with domain terminology, without concerning about scientific workflow concepts 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 the scientific workflows requirements, such as input and output data and chaining of tasks. Its key concepts are: Step, Task, Method, Algorithm, Software, Measure and Data. A Step represents a macro activity of the workflow, and models its highest abstraction level. A Task is a specialization of Step and defines a specific problem that is being solved. A Method is a methodology (or scientific model) used as a basis for an Algorithm. An Algorithm is how a Task is performed, and is based on a specific Method. A Measure is a representation of how an Algorithm performance is measured. Software 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. Data is the semantic representation of an input or output of any Software. Each Software consumes and produces a specific type of Data. The input and output of a Software is modeled in SciFlow. 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 S_i always preceeds a step S_j if there is a task T_i that preceeds a task T_j, T_i follows algorithm A_i, and T_j follows algorithm A_j, A_i is implemented by software Sw_i and A_j is implemented by software Sw_j, Sw_i outputs data D_i and Sw_j outputs data D_j. 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 "Preceeds" that indicates which Step or Task precedes another in a logical order.

3. USING ONTOLOGIES TO SUPPORT ABSTRACT WORKFLOW COMPOSITION

Scientific workflow composition is an open problem. One of the difficulties is to represent the experiment while moving along these different levels of abstraction. Domain ontologies provide semantic support to workflow activities and data. Combining domain ontologies with workflow ontologies associated with abstract workflows provides a flexible representation mechanism and allows for navigating through several hierarchical levels. These characteristics can help the composition process, for example, when looking for a specific abstract level component or when finding components that are (or may be) used in sequence. Scientific methods and algorithms can be represented along the composition of the abstract workflow, independent from technological issues. It also helps concrete workflow definition. By following an abstract definition, corresponding executable resources can be found to derive the workflow at the concrete level, while still being independent from the SWfMS. 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 of the abstract workflow. Only programs or services 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 these several alternatives.

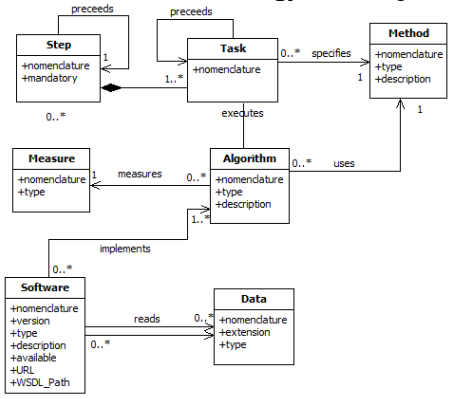

Figure 1 An excerpt of the SciFlow ontology

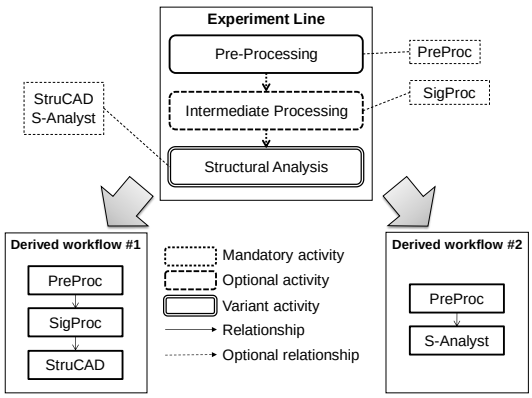

Figure 2 An example of an experiment line derivation

In summary, our approach 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 a curation team (a team responsible for managing provenance in scientific experiments) and/or by scientists. The terms associated may be methods, algorithms, artifacts, measures, depending on the ontology that is imported to the abstract representation; (ii) A checking mechanism to verify program and service 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 run a check type consistency validation of data types in the definition of the scientific workflow, 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 equivalency 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 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 users 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 $\rho 1$, $\rho 2$ and $\rho 3$. 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 $\rho 1$, $\rho 2$ and $\rho 3$. SciFlow coupled to a 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 α perform? What kind of Method is used by Software δ? What types of Data does the Software β handle? What Algorithm does Software Δ implement?

4. MODELING AN ABSTRACT WORKFLOW WITH SEMANTICS ASSOCIATED

Modeling a computational experiment is very similar to designing an in vivo or in vitro scientific experiment (Travassos and Barros 2003). In a scientific experiment in laboratory scientists choose a scientific method to follow, document steps executed and then analyze results. Scientific experiments carried out in a computational environment should also be considered in the same way. Their representation should reflect the scientific method, the chosen algorithms and so on. Experiment lines (Ogasawara et al. 2009) are an option to model abstract workflows.

An experiment line (Ogasawara et al. 2009) may be defined as an abstract workflow capable to derive multiple workflows at concrete (executable) level. It is a flow of activities in an abstract workflow, where each activity behaves like an independent component. Each abstract activity may be implemented by a list of compatible sequences of concrete activities. Also, a sequence of abstract activities may be grouped to form another abstract activity. An abstract activity may be mandatory (if it must be used in all derived concrete workflows), optional (if it may be suppressed from a derived workflow related to the experiment line) or variant (if it has more than one alternative program to implement its conceptual component behavior). An example of derivation is in Figure 2.

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 between 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.

We implemented SciFlow in OWL and coupled to experiment lines to support our proposal. SciFlow facilitates the derivation process and acts as a controlled vocabulary for the attributes of the software model. The ontology terms are used to query on metadata and on provenance data, thus helping scientists during derivation process through its inference mechanisms. As mentioned in Section 3 we must have a unified conceptual model to represent the abstract and concrete definitions along with ontology concepts. We extended the experiment line conceptual model (dark gray classes in Figure 3) to create this unified model, including nine classes to map any workflow ontology to a domain conceptual model. This mapping metamodel allows scientist to associate an ontology concept to a workflow activity. Each one of those classes was then implemented as a table in the physical database schema, following the work of Astrova and Kalja (2008).

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 OtlConcept 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 relations and property values. Although the ontology concepts are present in the conceptual model, inference is still applied using an external reasoner (e.g. Pellet2). Once the inference is completed, the concepts of the ontology are associated to a specific

[2] http://clarkparsia.com/pellet/

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.

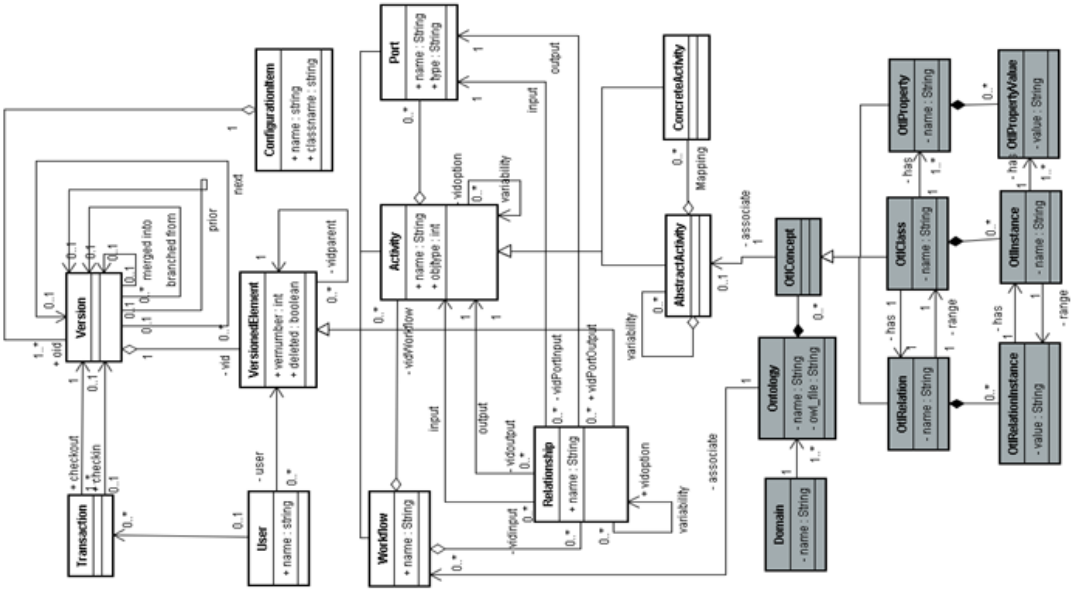

Figure 3 Conceptual model for experiment lines

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 whole 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 JENA3 to handle with OWL files. Also, a specific user interface was developed to import the OWL ontology to the model. The next step was to model the experiment line. We modeled an experiment line, initially represented as a XPDL file without concepts from the experiment. In the GExpLine tool, the imported ontology was 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, as illustrated in Figures 4a and 4b. 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 derive concrete scientific workflows modeled to many formats, including to Kepler (Altintas et al. 2004).

5. 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,

[3] http://jena.sourceforge.net/

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. (Wolstencroft et al. 2007) proposes an ontology based on a specific project. Ludäscher et al. (Ludascher et al. 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. Berkley et al (2005) and Ludäscher et al. (2003) highlighted the need for conceptual-level features in scientific workflows. They show how to couple domain ontologies to the Kepler SWfMS, through semantic annotations. However, they do not provide different levels of abstraction for scientific workflows, thus it cannot be used to represent abstract workflows in general. The work of Majithia et al. (2004) distinguishes between different levels of abstraction of loosely coupled experiment workflows to facilitate reuse and sharing of experiments. It has an approach for semantic composition to design scientific workflows in semantic grids. However, they focus on the semantic grid structure, thus navigation along abstract levels is not supported. myGrid is a semantically rich approach (Wolstencroft et al. 2007) coupled to Taverna SWfMS. It is 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 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. A careful survey (Yu and Buyya 2005) for Grid SWfMS reinforces the lack on semantic support for such distributed systems.

6. CONCLUSIONS

In this paper we proposed an ontology-based approach to add semantics to abstract representations of workflows. 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 his choice. Our approach was used in a real experiment in the deep water oil exploitation for oil platforms fixation. The ontology specialization helps scientists in understanding the scientific workflow domain and the processes that are being specialized in the SciFlow ontology.

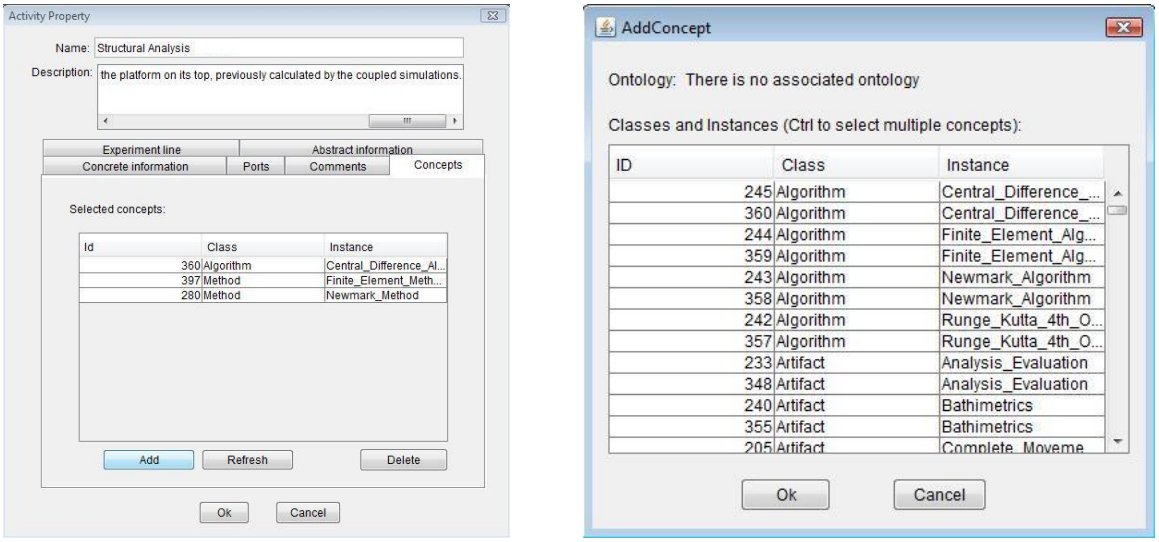

Figure 4. The association of ontology concepts to an abstract activity (a), and the list of concepts imported from the ontology (b)

7. REFERENCES

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., Mock, S., (2004), "Kepler: an extensible system for design and execution of scientific workflows". In: *Scientific and Statistical Database Management*, p. 423-424, Greece.

Astrova, I., Kalja, A., (2008), "Storing OWL ontologies in SQL3 object-relational databases". In: *Proceedings of the 8th conference on Applied informatics and communications*, p. 99-103, Rhodes, Greece.

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

Cannataro, M., Comito, C., (2003), "A Data Mining Ontology for Grid Programming", *PROC. 1ST INT. WORKSHOP ON SEMANTICS IN PEER-TO-PEER AND GRID COMPUTING, IN CONJUNCTION WITH WWW2003*, p. 113--134.

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

GExp, (2009), *Brazilian project for supporting large scale management of scientific experiments*, http://gexp.nacad.ufrj.br/.

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

Ludäscher, B., (2009), "What Makes Scientific Workflows Scientific?", *Scientific and Statistical Database Management*, , p. 217.

Ludascher, B., Altintas, I., Gupta, A., (2003), "Compiling abstract scientific workflows into web service workflows". In: *Scientific and Statistical Database Management*, p. 251-254, Cambridge, MA.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Murta, L., Ogasawara, E., Oliveira, D., Cruz, S. M. S. da, Martinho, W., (2010), "Towards Supporting the Life Cycle of Large-scale Scientific Experiments", *Int Journal of Business Process Integration and Management*, v. 5, n. 1, p. 79–92.

Ogasawara, E., Paulino, C., Murta, L., Werner, C., Mattoso, M., (2009), "Experiment Line: Software Reuse in Scientific Workflows". In: *Scientific and Statistical Database Management*, p. 264–272, New Orleans, LA.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., (2009), "Using Ontologies to Provide Different Levels of Abstraction in Scientific Workflows". In: *5th IEEE International Conference on e-Science*, Oxford, UK.

Shoshani, A., (2009), "The Scientific Data Management Center: Providing Technologies for Large Scale Scientific Exploration", *Scientific and Statistical Database Management*, , p. 1-2.

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

Taylor, I. J., Deelman, E., Gannon, D. B., Shields, M., (2007), *Workflows for e-Science: Scientific Workflows for Grids*. 1 ed. Springer.

Travassos, G. H., Barros, M. O., (2003), "Contributions of In Virtuo and In Silico Experiments for the Future of Empirical Studies in Software Engineering". In: *Proc. of 2nd Workshop on Empirical Software Engineering the Future of Empirical Studies in Software Engineering, Roma*, p. 117-130

Wolstencroft, K., Alper, P., Hull, D., Wroe, C., Lord, P. W., Stevens, R. D., Goble, C. A., (2007), "The myGrid ontology: bioinformatics service discovery", *Int. J. Bioinformatics Res. Appl.*, v. 3, n. 3, p. 303-325.

Yu, J., Buyya, R., (2005), "A Taxonomy of Workflow Management Systems for Grid Computing", *Journal of Grid Computing*, v. 34, n. 3-4, p. 171-200.