SGSDesigner, the ODESGS Environment User Interface

Asunción Gómez-Pérez
Ontology Engineering Group
Campus de Montegancedo s/n,
28660 Boadilla del Monte, Madrid. Spain.
(+34) 913367439
asun@fi.upm.es

Rafael González-Cabero
Ontology Engineering Group
Campus de Montegancedo s/n,
28660 Boadilla del Monte, Madrid. Spain
(+34) 913367439
rgonza@fi.upm.es

ABSTRACT

In this demo, we will show SGSDesigner, the ODESGS Environment user interface. ODESGS Environment (the realization of the ODESGS Framework [1]) is an environment for supporting both a) the annotation of pre-existing Grid Services (GSs) and b) the design of new complex Semantic Grid Services (SGSs) in a (semi) automatic way. In the demo we will focus in the annotation of a WSRF GS, using the annotation process proposed by the ODESGS Framework.

Categories and Subject Descriptors

D.2.2 [Software Engineering]: Design Tools and Techniques-*user interfaces*

General Terms

Design, Theory.

Keywords

Semantic Grid Services, Problem-Solving Methods.

1. INTRODUCTION

GSs are defined as a network-enabled entity that offers users some capabilities. In order to make them computer-interpretable, user-apparent and agent-ready, we annotate them with formal and explicit semantics.

Our proposal for annotating GS (and designing new complex SGS from these annotated ones) is the ODESGS Framework [1]. It uses ontologies and Problem-Solving Methods (PSMs) to describe the features of GS operations (a PSM is defined as a domain-independent and knowledge-level specification of the problem solving behavior that can be used to solve a class of problems [2]).

The process of annotating a GS in the ODESGS Framework (what we will show in this demonstration) starts with the creation of the knowledge level description (step 1 in Figure 1). Once this GS knowledge level description is created (the GS can be optionally checked, step 2), it can be automatically translated into different representational instantiations (the last step in Figure 1). The knowledge and representational level descriptions are both attached to the GS by means of Semantic Bindings [3](S-Bindings), allowing thus that a single GS may have multiple descriptions, which could be expressed in different languages and formalisms.

As we have already stated, ODESGS Environment realizes the ODESGS Framework. Therefore the front-end of the ODESGS Enviroment should be a graphical user interface that gives support to all these steps of the annotation process shown in Figure 1. This interface, on the one hand, should allow the easy creation of these knowledge level specifications (i.e. should facilitate knowledge entities creation), and on the other hand, should aid the user to attach pre-existing GSs to these descriptions (i.e. should facilitate S-Bindings creation). Finally, it must be able to communicate with different translators. Therefore, SGSDesigner current main features are:

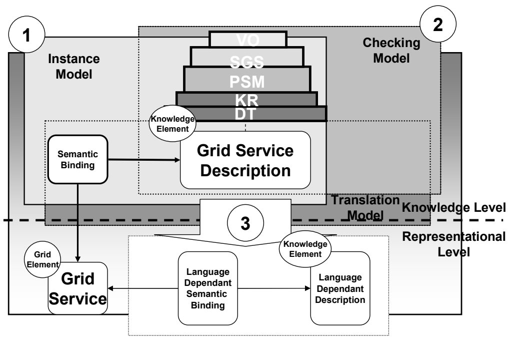

Figure 1 ODESGS Framework Annotation Process.

- ***Knowledge level descriptions are made in a graphical fashion***. The user is not aware of the formalisms used to represent the service; the whole design of the PSMs and other diagrams is carried out just by drawing or dragging and dropping knowledge components.
- ***Semantic markup export capable***. Once the knowledge level description and the S-Bindings are defined, an RDF(S) representation of them can be automatically generated. The service model that is used is being developed in the Ontogrid Project[1].
- ***Multiple and heterogeneous ontologies handling***. The editor can use OWL and RDF(S) ontologies stored in files or ontologies available in any instance of the WebODE ontology workbench.
- ***WSRF Compliant***. Currently, SGSDesigner is able to annotate WSRF[2] GSs, as it will be described later.

2. SGSDesigner

The SGSDesigner design has been inspired in the classical representation of PSMs. It includes hierarchical trees of tasks (abstract domain independent operations) and methods (abstract domain independent reasoning processes); input/output interaction diagrams of the tasks; diagrams about how the sub-tasks that compose a method are orchestrated; and data flows that describe data exchange of the sub-tasks. Let us describe each of the elements of this user interface.

[1] http://www.ontogrid.net

[2] http://www.globus.org/wsrf/specs/ws-wsrf.pdf

Workspaces are the main components of SGSDesigner. They are the concept of projects in software development tools. They contain the definitions of knowledge components and provide all the mechanisms and diagrams for both defining and storing them. Therefore, before we start working with SGSDesigner for creating the description of the service, we will select, at least, one workspace to use.

Workspaces have two general areas: trees and views (they are identified in Figure 2). Let's describe their intended use in detail.

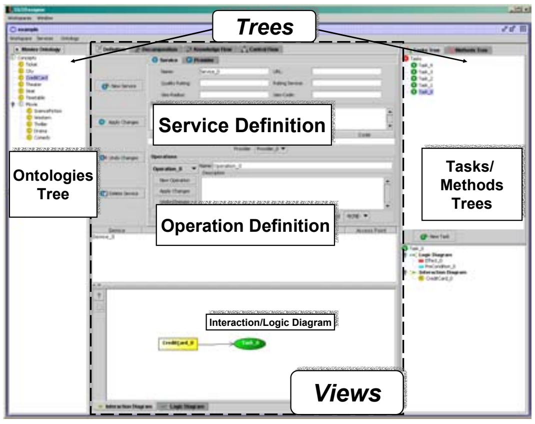

Figure 2 SGSDesigner Workspace.

Trees show the hierarchy of the knowledge components, such as tasks, methods, and ontologies elements.

- ***Ontologies trees.*** These trees show the concepts and attributes of the ontology (or ontologies) used to specify all the knowledge components. They can import the concepts and attributes that will be shown in the ontology tree. Then, the user could drag the icons representing a concept/attribute and drop them into the diagrams that enable the specification of the input/output roles of the tasks and methods.
- ***Tasks(Methods) tree.*** This tree allows users to create the tasks (methods) associated to the functional features of the operations. Once the tasks (methods) have been created, the user can drag, from this tree, the node representing a task (method) and drop it into the diagrams.

Views allow users to specify all the features of the knowledge components that describe a service, and they are represented as tabs:

- ***Definition View*** is used to specify the non-functional features of a service of the service (e.g., name, description, URL, providers, etc.) and the functional properties of each of its operation. Each of these operations is described by associating it to a task (what the operation does) and to a method (how the method performs this task). The operation task is defined by means of two diagrams: the ***Interaction Diagram,*** which specifies the type of the inputs/outputs of the task; and the ***Logic Diagram***, which contains the pre/post-conditions, assumptions and effects of the task. They are shown as colored rectangles.

  Methods are described by these diagrams plus the Knowledge/Control Flow Views.
- ***Decomposition View.*** This tree-like view allows users to specify the hierarchy of tasks and methods. Composite methods are decomposed into sub-tasks, which will be solved by other methods, and so forth. The leaves of a decomposition diagram are atomic methods.
- ***Knowledge Flow View.*** This view defines the data flow and the relationship between the inputs/outputs of the sub-tasks of a composite method.
- ***Control Flow View.*** This view describes the control flow of composite methods. The elements of this view are the sub-tasks of the method plus some workflow constructions (e.g. if-then, while-until, split and join, etc.)

In the demo, as we will annotate a pre-existing WSRF GS, we will create a task and its describing atomic method. Nevertheless, we will also briefly show how to use all these views to create a complex SGS, creating thus several task and several composite methods.

2.1 WSRF Annotation

Once the knowledge level description of the SGS has been created with the aforementioned views and trees, we will use the WSRF Import Wizard. It will guide us through several steps, which will finally associate each of the operations (and its inputs and outputs) of the GS to the knowledge level description of these operations (and the input and output roles of its associated task as Figure 3).

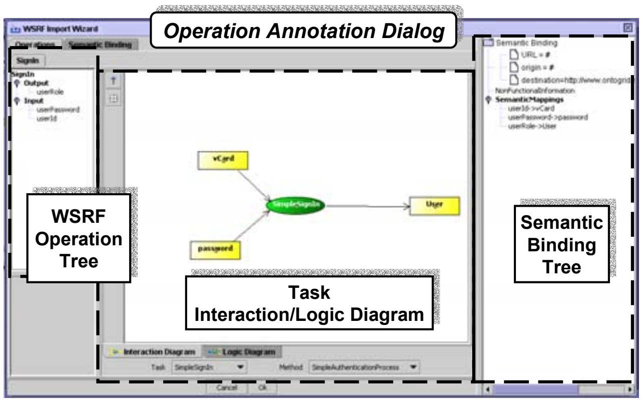

Figure 3 WSRF Service Annotation.

Finally, we will obtain the RDF instance of both the GS description and/or the S-Binding.

3. ACKNOWLEDGMENTS

This work has been partially financed by the Ontogrid Project (FP6-511513) and by a grant provided by the Comunidad Autónoma de Madrid (Autonomous Community of Madrid).

4. REFERENCES

[1] C. Goble, A. Gómez-Pérez, R. González-Cabero, M. Pérez-Hernández (2005) ODESGS Framework, Knowledge-based annotation and design of Grid Services, ICSOC 2005 341-352

[2] E.Motta (1999): Reusable Components for Knowledge Modeling, IOS Press

[3] I. Kotsiopoulos, P. Alper, O. Corcho, S. Bechhofer, C. Goble, D. Kuo, P. Missier Ontogrid Deliverable D1.2 Specification of a Semantic Grid Architecture October 7th, 2005