Towards the Generation of Explanations for Semantic Web Services in OWL-S

Carlos G. Fernandes, Vasco Furtado
University of Fortaleza
Av. Washigton Soares 1321, Fortaleza, CE, Brazil
carlosgustavo@edu.unifor.br, vasco@unifor.br

Alyssa Glass, Deborah L. McGuinness
Stanford University
Stanford, CA 94305 USA
{glass | dlm} @ksl.stanford.edu

ABSTRACT

In this article we define a system, called *XplainS* that automatically generates an infrastructure for providing explanations of semantic web services described in OWL-S. *XplainS* has a strategy for generating production rules from a flow where several levels of distribution of the services exist.

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *Representation languages, Representations (procedural and rule-based), Semantic networks.*

General Terms

Algorithms, Languages.

Keywords

Explanation, Web Services, Semantic Web.

1. INTRODUCTION

Semantic Web Services (SWS) are prominent components in the Semantic Web context. They possess descriptions of service functionalities along with input and output parameters and are being used to provide automatic procedures of discovery, composition and execution. SWS users need to understand how the solution was produced before they can be expected to trust and depend on the solution. They need explanations that describe the flow of the SWS and the steps of inference that were followed to produce a particular result.

In this article, we propose an approach called *XplainS*, which is embedded in OWL-S and permits the automatic generation of an infrastructure to supply SWS explanations. The approach involves a strategy to generate rules used for representing explanations of the execution of Semantic Web Services. Such rules are represented in Proof Markup Language (PML), an ontology used as the basis of the Interlingua for the Inference Web (IW) explanation framework [1]. In this approach, explanations can be passed along with SWS, allowing the construction of rich explanations based on distributed and interoperable services.

2. GENERAL XPLAINS ARCHITECTURE

The representation of knowledge for explanations is an acyclic graph where each node of the graph is represented by a PML structure defined in IW. Each PML node of the explanation graph has three main pieces of information: the inference rule (utilized to guide the search for information), the conclusion (declarative information representing the information concluded by the process execution) and the antecedents of the inference rule. Our intention is to create a declarative representation of the execution process of a SWS in order to utilize the explanation-manipulating tools already developed in IW. The process of generating the information from the process execution flow graph is done in parallel with the OWL-S execution flow. Notice that OWL-S, in addition to supplying the process ontology, has a software interface that makes the ontology "operational" by executing the web services associated with the ontology description.

In OWL-S, there are two types of processes: atomic and composite. The atomic process is executable, and invokes the web service. The composite process is not executable, but rather is a set of other processes. Thus, a clear and recursive execution structure can be seen, whereby the API [2] interprets the composite processes recursively until it finds an atomic process. Notice that upon interpreting the process flow, it is also necessary to identify the conditions (*Repeat* and *IfThenElse*) and the pre-conditions for the process to be executed. Rules of process finalization that result in the activation of such process are generated based on the interpretation of the conditions/pre-conditions. Moreover, *XplainS* creates a process execution rule that represents the element of construction, and the scope of the rule is represented by the condition at the top of the conditions stack. Such procedure is repeated until all of the processes have been executed.

One of the most important aspects of web service explanations is in the context of distributed services. For example, a travel agency, when attempting to reserve a hotel room for a client, can use a service to find hotels that meet the user's location preferences. An explanation based only on the description of the immediate call to the service would not be satisfactory for the user, who may want detailed information produced by the web service invoked by the hotel services. The heterogeneity of the web and the dynamicity supplied by the automatic composition of web services makes an *a priori* preparation of the explanation infrastructure very difficult. Our approach solves this problem by embedding the generation of the explanation infrastructure at the OWL-S level. Therefore, the only necessary requirements are web services that are both described in OWL-S and running *XplainS*.

In order to permit the recording of a distributed web services audit file without requiring a change in the legacy web services structure, we added a new characteristic in the API OWL-S — explanation manager call. It is activated every time a web service described in OWL-S is called.

Given an application that executes an OWL-S flow via the OWL-S API, for which the explanation logs are recorded locally[1], the explanation file is represented in PML where each step of execution of the web service is represented in a data structure in the form of a graph. Each OWL-S proof manager communicates with a web service, called *WSProofRecorder* which, in turn, has the task of maintaining a persistent explanation log. When the API OWL-S invokes a particular web service, the explanation manager automatically sends a requisition to the *WSProofRecorder*, informing that, from now on, the generation of the explanation file is the responsibility of the web service that has been invoked. The communication between the proof manager and the *WSProofRecorder* is done immediately after an execution step made in the web service. The web service invoked does not need to complete its entire task before transmitting the explanation file. Thus, *XplainS* supports real-time explanation interactions with the user. Notice that during the communication between the explanation manager and the *WSProofRecorder,* information on file order control is exchanged in order to permit the correct representation of the sequence of the process in execution.

3. VISUALIZING THE PROCESSES EXECUTION FLOW

3.1 Overview

XplainS collects information on the execution of the process in order to generate explanations. The supply of explanations uses an explanation ontology that includes possible question types that the user may ask, along with several ways of answering such questions, requiring no prior knowledge of the problem domain. These questions are divided into three levels: explanation at the process flow level; explanation with abstractions of the processes; and explanations using concepts from the process definition. In each of these levels there is another division that refers to the moment in which the explanation system is utilized. That is, certain questions/answers are only acceptable during the execution of the process, just as others are only utilized once the finalization of the process has been concluded. There are also questions/answers that are independent of the moment in which they are requested.

For first level, only the flow that was executed is described. At this level, the user can understand the order of execution, and visualize why a particular process was executed and another was not. Though the user is able to visualize the process flow, however, he may not necessarily be able to identify the meaningful steps among the many process details. Thus, a second level provides abstractions at determined points of the process execution, allowing a better understanding of what has occurred. The third level refers to the use of information from ontologies, describing the inputs and outputs of the processes. In this article, we will concentrate on the first two levels.

[1] This architecture uses local explanation logs but there is nothing about PML, Inference Web or our architecture that requires this, and in fact, PML and Inference Web expect distributed explanation logs.

3.2 Explaining the Process Flow

Based on the grouping of execution process rules, *XplainS* supplies a service of dialogue that permits user interaction during the execution of OWL-S. One of the objectives of *XplainS* is to answer questions related to the reasons, order, format and time in which the OWL-S processes were executed. The strategy of interaction is to supply an interface with basic questions and, based on answers to such questions, to supply additional follow-up questions that permit an understanding of the process execution inputs.

Two types of dialogue are supplied: a dialogue about the current execution of the process, and a dialogue about the final result produced by the process. Certain types of questions are related and their explanation may be requested during execution or when the process is completed. Thus, whenever there is a question about the process hierarchy (grouping1) or how the process is occurring (grouping2) or why the process has not been finalized (grouping3), the algorithm identifies the group that is most capable of answering the user's question.

XPlainS features a structure for manipulating the explanations of the execution flow on two levels. Each one of the levels presents the same information to the user with different levels of abstraction, i.e., with possible patterns that are identified and shown to the user.

On the first level of explanation—explanation of the process execution flow—the information collected and grouped through rules are utilized to supply explanations to the user. With those groups of rules, we can present the data in different ways: order of execution (sequence or simultaneousness), preconditions for a process to be executed, whether a process is comprised of other processes, etc. On this level, we utilize the rules formation structure (*ParentOf*, *PreconditionsMet*, *Support*) to present the information. With this type of information, the user has the possibility of understanding how the problem was (or is being) resolved and which actions are being taken, even though certain types of patterns occurring during execution are not easily perceived.

On the second level of explanation, the process abstraction level, we seek to reduce the quantity of information in order to draw the user's attention to particular portions of the explanation graph, by grouping portions of the graph and providing summaries. For example: in navigating the graph, we can identify that a process was executed n consecutive times. Thus, instead of listing the process information n separate times, we can summarize by simply noting the multiple executions of the same process.

Thus, for this explanation level, *XPlainS* captures patterns in the execution flows and, instead of showing the information "raw," creates new information abstractions of the data.

4. REFERENCES

[1] McGuinness, D. and Pinheiro da Silva, P. Explaining Answers from the Semantic Web: The Inference Web Approach. Journal of Web Semantics. Vol. 1 No. 4., pages 397-413, October 2004.

[2] OWL-S JAVA API. At http://www.mindswap.org/2004/owl-s/api/