AN INTERACTIVE COMPOSITION OF WORKFLOW APPLICATIONS BASED ON UML ACTIVITY DIAGRAM

Yousra Bendaly Hlaoui, Leila Jemni Ben Ayed
Research Unit of Technologies of Information and Communication
Tunis, Tunisia
Yousra.bendalyhlaoui@esstt.rnu.tn
Leila.jemni@fsgt.rnu.tn

ABSTRACT

In today's distributed applications, semi-automatic and semantic composition of workflows from Grid services is becoming an important challenge. We focus in this paper on how to model and compose interactively workflow applications from Grid services without considering lower level description of the Grid environment. To reach this objective, we propose a Model-Driven Approach for developing such applications based on semantic and syntactic descriptions of services available on the Grid and abstract description provided by UML activity diagram language as well. As there are particular needs for modeling composed workflows interactively from Grid services, we propose to extend the UML activity diagram notation. These extensions deal with additional information allowing an interactive and semi-automatic composition of workflows. In addition, this specific domain language contains appropriate data to describe matched Grid services that are useful for the execution of the obtained workflows.

Keywords: Grid services, Interactive, semantic, composition, Workflow application, UML activity diagrams.

1 INTRODUCTION

Today’s distributed applications [23] are developed by integrating web or Grid services [13, 14] in a workflow. Due to the very large number of available services and the existence of different possibilities for constructing workflow from matching services, the problem of building such applications is usually a non-trivial task for a developer. This problem requires finding and orchestrating appropriate Grid services in a workflow. Therefore, we propose an approach that allows semi-automatic and interactive composition of workflow applications from Grid services. To describe and model workflow applications we use UML [25] activity diagrams. Recently, several solutions were proposed to compose applications from Grid services such as works presented in [8, 17, 18]. However, the proposed solutions need interaction with user and guidelines or rules in the design of the composed applications. Consequently, the resulting source code is neither re-usable nor it promotes dynamic adaptation facilities as it should. However, for applications composed of Grid services, we need an abstract view not only of the offered services but also of the resulting application [31]. This abstraction allows the reuse of the elaborated application and on the other reduces the complexity of the composed applications. There are several architectural approaches for distributed computing applications [22] which make easy the development process. However, these approaches need rigorous development methods to promote the reuse of components in future Grid development application [16]. It has been proven from past experience that using structured engineering methods makes easy the development process of any computing system and reduces the complexity when building large Grid application [22].

To reduce this complexity and allow the reuse of Grid service applications, we adopt a model-driven approach [24]. Thus we introduce in this paper a new approach to build, interactively, workflow applications by following OMG(s) principals of the MDA in the development process [2, 3, 4].

In this approach [2, 3, 4], our focus is to compose and model workflows from existing Grid services that represent the main aspect in the development of Grid services applications. The workflow modeling identifies the control and data flows from one depicted Grid service's operation to the next to build and compose the whole application. To model and express the composed workflow of Grid services, we use as abstract language the
activity diagrams of UML [25]. The provided model forms the Platform Independent Model (PIM) of the proposed MDA approach. This model is more understandable for the user than an XML [35] based workflow description languages like BPEL4WS [15] which represent the Platform Specific Model (PSM).

This paper is organized as follows. Section 2 presents the related work. Section 3 introduces the different components of the composition system; section 4 specifies our proposed UML profile, composition patterns and different steps of the interactive composition process. Finally, section 5 concludes the paper and proposes areas for further research.

2 RELATED WORK

Many works were carried out in the field of Grid and Web services composition, such as works presented in [8, 17, 18, 19, 20, 28, 29, 30]. In [28] authors were interested in the semi automatic composition of web services and proposed a validation approach based on the semantic descriptions of services and on a logic based language to describe and validate the resulting composite Web services. However, the resulting composed web service is not clear for user who is not familiar with logic based languages. In our contribution, we propose a solution not only to compose workflows from available Grid services, but also to provide graphical and comprehensive models of the resulting workflows. In the same framework, authors in [29] proposed a composition approach of Web services based on Symbolic Transition Systems (STS). They developed a sound and complete logical approach for identifying the existence of available composition. They have emphasized upon the abstract representation of the composition request (the goal of the composition) and the representation of the resulting composite Web service. For the representation, authors have used UML state machine diagrams [25] which are suitable only to describe a sequence of component services without addressing the other forms of matching services in a workflow such as parallel branches or and-branches. On the other hand, UML activity diagrams that we use in our modelling approach support all kind of workflow composition patterns [10] such as parallelism, split and fork. The authors in [19, 20, 30] have proposed a Model Driven Approach for composing manually Web services. They were based on UML activity diagrams to describe the composite Web service and on UML class diagrams to describe each available Web Service. The user depicts the suitable Web service and matches it in the workflow representing the composite Web service using UML activity diagrams. This approach would have been better if the composition were automatically elaborated, since the number of available services is in increase with the existence of several forms and manners to compose such services.

Based on domain ontology description, we lead the user through to the composition process. Also, we provide for this user a graphical interface based on a domain specific UML language for automatic grid service composition. This UML profile [5] is based on stereotypes, tagged values and workflow patterns [5] that we propose to ensure the automatic composition. In the field of Grid services composition the most related work is the work presented by Gubala et al in [8, 17, 18]. In this work, the authors have developed a tool for semi automatic and assisted composition of scientific Grid application workflows. The tool uses domain specific knowledge and employs several levels of workflow abstractness in order to provide a comprehensive representation of the workflow for the user and to lead him in the process of possible solution construction, dynamic refinement and execution. The originality of our contribution is that firstly we save the effort of the user from the dynamic refinement and execution as we propose a Model Driven Approach which separates the specific model from the independent model.

Secondly, we use UML activity diagrams to deliver the functionality in a more natural way for the human user. The use of UML activity diagrams in the description of workflow application is argued in several works such as works presented in [1, 10, 12, 27]. Thus, the advantage of UML activity diagrams is that they provide an effective visual notation and facilitate the analysis of workflows composition.

In our approach, we propose an UML profile for composing systematically a workflow application from Grid services [5].

3 THE INTERACTIVE WORKFLOW COMPOSITION SYSTEM

The system allows an interactive and semantic composition of workflows from Grid services. As shown in figure 1, the system is composed of three components: a Grid Services workflows composer, an ontological Grid Services registry and a workflows execution system also we call it activity machine.

3.1 The Grid services workflow composer

This system is composed of three components: the composition tool, the transformation tool and the verification tool.

3.1.1 The composition tool

It provides a graphical interface in the form of UML activity diagrams editor allowing to the user an interactive, systematic and semantic workflow
Composition [6]. This composition is based on the composition process which will be detailed in section 4.3. In the Grid registry, services are described in an ontological form with statements regarding the service operation's inputs, outputs, pre-conditions and effects (the IOPE set) [26]. Through these notions, the composition system is able to match different grid service's operations into a workflow following a reverse traversal approach. Thus, and by associating the required data with the produced output, the composer constructs a data flow between Grid service’s operations using our workflow composition patterns and UML profile[5]. The composer may also use a specific notion of effect that may bind two operations together with non-data dependency. If the Grid registry fails to find the right operation, the composition process stops. Otherwise, the composition process will stop when all workflow dependencies are resolved.

The request is sent to the Ontological Grid Registry in the form of SPARQL query [34]. This language provides a higher-level access to the ontology transcribed knowledge for the automatic discovery and semantic matching of services.

Therefore, once the workflow model is built, it should be validated and verified to ensure its reliability before being executed and reused as sub-workflow.

3.1.2 The transformation tool

To support the verification and the execution of workflow models described in UML activity diagrams (UML-AD), the transformation tool translates the activity diagram into a Hyper-Graph (HG). This HG will be translated as well by the transformation tool into a NuSMV format file according to a relative semantic. The details of these semantics may not be relevant to the topic for which the paper is submitted. However these details could be made available.

3.1.3 The verification tool

Checking errors in design models like UML activity diagrams is essential since correcting an error while the system is alive and running is usually very costly [21]. Consequently, workflow activity diagram models should be spotted and corrected as early as possible [6].

Several techniques are used in the field of behavioural design verification such as theorem proving and model checking [11]. The latter is the most useful because it is fully automatic and gives feedback in case of detected errors. It verifies whether some given finite state machine satisfies some given property specified in temporal logic [9]. For activity diagrams, symbolic model checking has proven to be an efficient verification technique [11]. Thus, our verification tool is based on NuSMV symbolic model checker [9] that supports strong fairness property which is necessary to be verified in a workflow model to obtain realistic results. With the model checker, arbitrary propositional requirements can be checked against the input model. If a requirement fails to hold, an error trace is returned by the model checker. The transformation tool translates systematically the error trace into an activity diagram trace by high-
lighting a corresponding path in the given activity diagram.

3.2 The Grid registry

During the workflow composition process, the Grid registry provides the composer system with the description of services available at the moment and provides reasoning capabilities to enable proper matchmaking of services inputs and outputs.

The Grid registry [6] is an ontological distributed repository of Grid services and sub-workflows. This registry is responsible for storing and managing documents which contains descriptions of syntax and semantics of services and their operations expressed in an RDF file [33]. The semantic Web is making available technologies which support automate knowledge sharing. In particular there are several existing initiatives such as OWL-S [26] which proves that ontologies have a key role in the automating service discovery and composition. That knowledge is based on semantic descriptions of service classes published by the service developers and provided in the Grid environment [16]. Our Grid registry is based on an ontological description of services and workflows.

The service ontology [7] provides concepts and properties that allow description and matchmaking of available services. A part of this ontology is common to all services and it is based on a standard semantic web service description ontology OWL-S [26] which makes interoperability with existing services. A part from the common ontology, there is a domain specific part of the ontology. The domain service ontology [7] allows users to extend the common ontology schema in order to provide a better specification of services as well as their inputs and outputs. For these we define a data ontology [7] which provides concepts and properties for describing services input and outputs.

Ontology alignment [7] is a process for finding semantic relationships among the entities of ontologies. Its main activity is to find similar concept in ontologies being aligned, in order to map them. The measures for similarity computation can be divided into two general groups; namely lexical measures and structural measures. Lexical measures are based on surface similarity such as title or label of entities. In contrary, structural measures try to recognize similarities by considering structures of ontology graphs. The most advanced similarity algorithms use combination of multiple similarity measures to obtain more information about concepts similarity. In our Grid registry, we adopt an approach using a combination of lexical and structural similarity [7].

3.3 The workflow execution system

The reliable workflow model is sent to the workflow execution system [6] which produces implementation code for handling control flow and data flow. The activity diagram describing the workflow model is translated into a specific XML file which will be the input of the execution system. A workflow execution system executes different workflow activities specified in the workflow XML document in the correct order and with their required inputs and outputs data. The execution of an activity corresponds to the invocation of a Grid service’s operation. The workflow execution system monitors these activities using the tagged values information expressed in the activities but does not perform them. An activity of the activity diagram modelling the workflow represents a state of the workflow execution system in which the system waits for an invoked grid service operation to complete its work. Hence, the defined semantics of activity diagrams for the verification describe the behaviour of the execution system. When the system enters a state relative to an invocation grid service node or activity $ai$, it invokes a piece of behaviour that is executed by the service or system environment. While the latter is in $ai$ (activity $ai$ is active), it waits for the termination event of the invoked piece of behaviour. When termination event occurs, the system reacts by executing the outgoing edge $E$: it leaves the $E$’s sources and enters the $E$’s targets and the execution process continues for the other activity nodes until the final node is reached.

4 UML BASED INTERACTIVE COMPOSITION OF WORKFLOWS FROM GRID SERVICES

In order to match and compose different Grid service’s operations, we need to analyze constructs of workflow models at higher abstraction level. Since UML [25] is the core of the MDA [24], we use its activity diagram language to model composed workflows. The composition system provides to the user a graphical interface to compose its request using a UML profile specific for the domain of composing systematically.

4.1 UML Profile for composing workflows

In this section, we present our UML profile which is based on Domain Specific Language (DSL) for customizing UML activity diagrams for the systematic composition of workflows from Grid services [5].

In our DSL (See Figure 2), an activity of an UML activity diagram represents a Grid service's operation, while object flows represent the types of results which flow from one activity to another.
Effects binding two operations are presented with control flows [5].

The name of an activity in the diagram represents the name of the Grid service's operation. This name must be specified as a Grid service could have more than one operation often called interface which are specified in its relative WSDL file [32].

There are two different types of activities: yet-unresolved activities and established activities of the composed workflow. The former represent the need for a Grid Service's operation to be inserted in order to complete the workflow. However, the latter represent abstract operations that are already included into the workflow.

As there are two different activity types in a Grid service workflow model, an activity needs to be typed and specified. To fulfil this, we propose to use the DSL modelling element invoke to stereotype an established activity which is used to invoke an external Grid service's operation and yet-unresolved to stereotype activities which are not yet resolved. Object nodes of an established activity are data stereotyped. Unknown input and output for a yet-unresolved activity are unknown stereotyped.

In our UML profile, an object node could be related to a final node as composed workflow of Grid application should always deliver a result.

4.2 UML-AD composition patterns

We identify, in this section, how UML activity diagrams support some of basic Grid service composition patterns [5]. These patterns are essential in the systematic building of workflow applications from Grid services. The use of these patterns depends on the number of the depicted Grid Service's operations and their inputs and outputs [5]. These operations are results of the semantic research elaborated by the ontological Grid services registry. This research is invoked by a request given by the composition system in order to complete an unresolved activity in the workflow. The Grid service registry provides zero, one or more operations producing the intended output. Operations are depicted to be inserted in the workflow interactively with the user.

4.2.1 Sequence Pattern

When the Grid registry provides one Grid service's operation that is able to produce the required result or the user selects one operation from the provided operation set; the composition system uses the sequence pattern to insert the operation in the workflow. In this case and as is illustrated by the figure 3, a single abstract operation or activity (e.g. GridService1Operation1) will be inserted in the workflow model described by the UML-AD language. This operation may also require some data for itself (e.g. GridService1Operation1Input) and thus it may introduce a new unresolved dependency (e.g. the yet-unresolved stereotyped activity). So, we use a follow-up method to build a simple pipeline-type sequential workflow: a sequence pattern.

![Figure 2: Meta-model of Grid service workflow composition specific UML activity diagram language](image-url)
A Sequence pattern is composed with sequential activities which are related with control flow (non data operations dependency) or object flow (data operation dependency).

4.2.2 And-branches pattern

The and-branches pattern is introduced when the introduced operation represented by an abstract UML activity has more than one input. This pattern is based on the Synchronization pattern presented in [9].

This pattern starts with object nodes, representing alternative operation inputs, which flow to a join node. The latter is linked to the abstract grid service's operation. This operation introduces some unresolved dependencies in the workflow.

Semantically, several services instances are invoked in parallel threads and the join will wait for all flows to finish. As illustrated in Figure 4, the operation of the Grid service GridService1Operation1 needs two inputs data GridService1Operation1Input1 and GridService1Operation1Input2. The relative pattern produces two parallel threads in the workflow.

4.2.3 Alternative branches pattern

When the Grid registry provides more than one operation able to produce the required result, and the user do not select one of them, the composition requires a specific pattern: the alternative branches pattern.

This pattern combines the Exclusive Choice and Simple Merge patterns presented in [9]. In this pattern, each alternative service's operation is linked to object node representing the required input which both of them flow to a merge construct.

Semantically, several services instances are invoked in parallel threads and the merge will only wait for the first flow to finish. We distinguish, in Figure 5, two different Grid service's operations, GridService1Operation1 and GridService2Operation1 providing the same output data DataOutput.

4.2.4 Alternative services pattern

When composing workflows from Grid services, a specific matching based on semantic comparison could provide two or more different Grid services performing each of them the required operation. In such case and when the user do not choose one of the depicted Grid service’s operations, the composition system uses the alternative services pattern to involve the operations in the workflow model.

In this pattern, the Grid service’s operation to insert is modelled by a composed super-activity with a specified input data object and specified output data object (Figure 6). The super-activity is stereotyped as AlternativeServiceInstance to indicate that its task may be accomplished by a set of alternative service's instances. These alternative service instances are described with sub-activities. The sub-activities shall be grid service instances and thus stereotyped as invoke. It was up to decision mechanism of the workflow execution engine to choose which service instance in such given workflow node is to be invoked and executed. In Figure 6, the data DataOutput is provided from GridServiceOperation service operation which could be GridService1Operation1 provider or GridService2Operation2 provider.
4.3 The composition process

Figure 7 illustrates the scenario of the composition process of workflows from available Grid services. This composition is based on the domain specific UML activity diagram language presented in section 4.1. In the following, we comment the different process steps of the scenario presented in the figure 7.

Step 1: The user builds its composition request by specifying what kind of outcome or result that it expects from the workflow application execution.

Step 2: The composition system analyses the desired output and sends a SPARQL query to request the ontologies of the Grid registry describing the available Grid services. The composer requests the Grid registry for a Grid service’s operation having the specified result as output.

Step 3: If the required operation is not found and all unknown results are resolved then the composition process stops.

Step 4: If the required operation is found then the system displays its characteristics to the user to confirm the choice. The register may provide more than one operation. In such case the user could choose the operation to insert in the workflow model from the given list. If it does not specify its operation, then the system inserts all the given operations using one of the composition patterns presented in section 4.2. Relatively to the number of depicted operations and their inputs and outputs, the composer chooses the right composition pattern.

Step 5: For each input of inserted operation, the system defines one unresolved dependency as a workflow activity which is not yet established. This activity depends on some Grid service’s operation. For each unresolved dependency the composer asks the user if it wants to continue the composition process or not. If the response is positive the composer re-executes the process from the step 2 to resolve the current unresolved dependency.

Figure 7: Scenario of the interactive composition of Grid service workflows based on UML activity diagrams
5 Illustration of the interactive workflow composition

In the following, we illustrate the composition process through the example of the domain of the city traffic pollution analysis. This application, as presented in [23], targets the computation of traffic air pollutant emission in an urban area.

Step 1: Figure 8 shows an example of initial workflow that represents a composition request for the results of the pollutant emission due to the city traffic. The desired result, PollutionEmission, is described by the rectangle representing the object node in the relative activity diagram.

![Figure 8: Initial workflow as a composition request](image)

Step 2: Figure 9 represents the workflow of the computation of traffic air pollution analysis after one step of composition. The service’s operation, delivering the PollutionEmission result, is AirPollutionEmissionCalculator. This operation is the result of the composer query asked to the ontological Grid registry. The operation requires two inputs TrafficFlowFile and PathsLengthFile, thus it infers two unresolved dependencies in the activity diagram modelling the composed workflow.

![Figure 9: An example of workflow after one step of composition](image)

Step 3: For every dependency that needs to be resolved i.e. a yet-unresolved activity, the composer contacts the ontological registry in order to find suitable service’s operations that may produce the required result. The services are described in an ontological form with statements regarding the service operation’s inputs, outputs, preconditions and effects (the IOPE set) [26]. Through these notions, the composer system is able to match different operations into a workflow following a reverse traversal approach. Thus, and by associating the required data with the produced output, the composer constructs a data flow between operation using workflow patterns and our UML profile [5]. The composer may also use a specific notion of effect that may bind two operations together with non-data dependency. In [10], five basic control patterns were defined to be supported by all workflow languages and workflow products. These patterns are Sequence pattern, Parallel split pattern, Synchronization pattern, Exclusive Choice and Simple Merge patterns. Figure 10 represents the example of city traffic analysis Workflow after the full composition activity. It involves several Grid service operations, sequence branches, parallel split branches, simple merge branches and a loop [5]. The loop is involved in the workflow diagram as the application iterates in order to analyze the possible traffic. The Figure shows also how UML activity diagrams support the five basic patterns in the composition specific domain of Grid services workflows [5]. In the example, some of object node or input data, such as VehicleType and StartZoneId, are given by the user of the application; they do not have an operation provider. This illustrates the interaction between our composition system and the user.

![Figure 10: The workflow application after the full composition](image)
6 CONCLUSION

In this paper, we have presented an approach for composing interactively workflows from Grid services [2, 3, 4, 6]. This composition is based on an UML profile for customizing UML activity diagrams to compose and model workflows [5] and on composition patterns [5] as well. The interactive composition process was illustrated through the example of city traffic pollution analysis domain [23]. We have developed and implemented most of the presented components of the composition system.

Actually, we are working on the implementation of the workflow execution system that invokes and executes the depicted Grid service instances and manages the control and data flows in a run time environment relatively to our proposed activity diagram semantic.

7 REFERENCES


