BIOINFORMATICS

Vol. 00 no. 00 2011
Pages 1–6

Semantic Modeling of Services and Workflows for German Grid Projects

Sudhir Agarwal [1], Martin Junghans [1], René Jäkel [2]

[1]Karlsruhe Institute of Technology (KIT),
Institute of Applied Informatics and Formal Description Methods (AIFB),
Karlsruhe Service Research Institute (KSRI),
Englerstr. 11, 76131 Karlsruhe, Germany
firstname.lastname@kit.edu
[2]Technische Universität Dresden,
Center for Information Services and High Performance Computing (ZIH),
Helmholtzstr. 10, 01069 Dresden, Germany
rene.jaekel@tu-dresden.de

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Many projects in academia and industry nowadays use and provide services electronically in order to be able to select from or target to a larger community across regional boundaries. With increasing demand for and supply of complex services, it has become difficult to find and compose appropriate services. The formal description of services and workflows facilitates users in finding appropriate services for a task at hand, to embed them into dynamic workflows or simply to get informed about interesting new services provided by the system.

In this paper we present an approach for modeling services and workflows semantically. The modeling language captures basic properties of services and workflows including their functionality modeled formally as a process, while the Web browser based graphical editor allows users to model their services and workflows in an easy fashion.

We present an example scenario in which a workflow composed out of services from the logistics domain is modeled. The scenario illustrates the description formalism and basic features of our description language and reveals how it can be applied to further projects linked with other activities within the German Grid Community.

1 INTRODUCTION

Services play an increasingly important role for accomplishing personal and business tasks. The reuse of existing services promises to save development costs if desired services can be easily identified. Likewise, it should be also possible to obtain the required functionality by a composition of existing services.

An increasing number of available service leads to the problem that finding and reusing appropriate services has become more difficult. Service based systems benefit from fast replacements of fallen out services and service compositions; the latter is typically exponential in the number of times service retrieval is invoked. In order to ensure benefits such as flexibility, cost reduction, etc. of the service based systems, there is a need for scalable and efficient techniques for retrieving required services.

Existing service description techniques that are based on syntactical service descriptions and service requests require much manual effort in addition. For instance, the offered functionality has to be examined manually if it was described in a textual representation that is not machine interpretable. Furthermore, syntactical service descriptions do not allow to draw conclusions from their descriptions and it is difficult to deal with different vocabularies automatically. The relationships between parameters like input and outputs of a service are not modeled such that they allow to infer the effects of a service, which is relevant for the composition of services. Lastly, service descriptions are often underspecified and do not allow to automatically execute the described services or workflow.

We present in this paper a semantic description formalism for services and workflows that overcomes the briefly mentioned shortcomings. The language is developed in the WisNetGrid project and aims at its application for services and workflows in the Grid. Here, we use the term workflow for a complex process behavior, which can be assembled by simple atomic service functionalities and existing complex behaviors (workflows). In addition to the description of the behavior, a service is offered with additional properties attached to its description.

In Section 2 we introduce the service description language. We highlight major language features in Section 3. Each feature covers a requirement of a description language for Grid service. A detailed list of language requirements was examined in [4]. Then, in Section 4, we show how the presented service description approach can be applied for the motivated scenarios like retrieval and composition. It explains our conceptual approach that coheres to the architecture implemented in the WisNetGrid project. In Section 5 we introduce a workflow example from a logistics scenario that was developed in the InterLogGrid project, which applies the technologies developed in WisNetGrid. The relation to

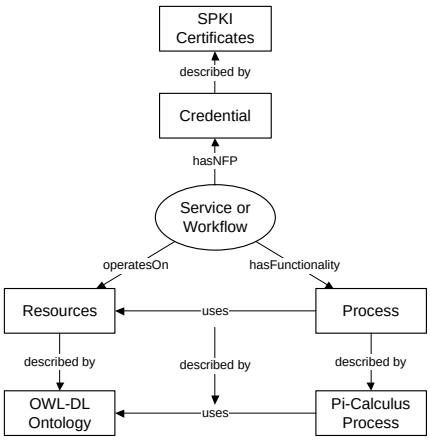

Fig. 1. Abstract Model of Services and Workflows

other approaches is given in Section 6 and we draw conclusions in Section 7.

2 SEMANTIC DESCRIPTION OF SERVICES AND WORKFLOWS

In this section, we give an overview of the formalisms we use for describing functional and non-functional properties of services and workflows semantically. We first present the abstract model of the service and workflow properties and then show how the properties can be described semantically.

2.1 Abstract Model of Services and Workflows

As shown in Figure 1, we consider functional and non-functional properties of services and workflows. We have justified and motivated in [10], why it is important to treat functional and non-functional properties uniformly. A description embraces a set of independent properties; one of which describes the functional and the others describe the non-functional properties. A service or a workflow operates on resources that we describe with an ontology to enable interoperability and automated reasoning on the description of the resources. The functionality of a service or a workflow is described with a π-calculus process expression, which allows description of even highly complex process with rather simple syntax. The resources used during the execution of a service or a workflow are referred to by corresponding associations from their use in the process expression to their ontological description. The ontology describing the resources semantically is often referred to as domain ontology of a service or a workflow. In addition to the functionality, we support modeling of the non-functional properties (NFPs) as well. Especially in case of NFPs it is very important that the values of the NFPs of a service or a workflow are trustable. Therefore, we model them as Simple Public Key Infrastructure (SPKI) certificates.

2.2 Description of Functionality

The functionality (Φ, O) of a service comprises the description Φ of its behavior, which is called Process in Figure **??**, and the description O of the static domain knowledge used within the process. The naming of the resources like constants and variables occurring in such a process expression is defined in O by expressions of the $\mathcal{SHIQ}(\mathbf{D})$ description logics.

The description formalism used to model the behavior is based on the π-calculus [15], a process algebra that supports dynamic workflows. Benefits of this approach, details on the description formalism, and its formal semantics were already introduced [1, 3].

DEFINITION 1. *Description Formalism of Process Behavior*

$$\begin{aligned}\Phi(a_1,\ldots,a_n) \quad ::= \quad & \mathbf{0} \mid c(x_1,\ldots,x_n).P' \mid \overline{c}\langle y_1,\ldots,y_n\rangle.P' \\ \mid \quad & l(x_1,\ldots,x_n)(y_1,\ldots,y_m).P' \mid P_1 \parallel P_2 \\ \mid \quad & [\omega_1]P_1 + [\omega_2]P_2 \mid @A\{y_1,\ldots,y_n\}\end{aligned}$$

The process expression Φ receives the arguments $a_1,\ldots,a_n$ upon invocation. They are described by description logics expressions. I.e., the arguments can be described by, for instance, their type and the relationships among them or to other individuals. The *null process* $\mathbf{0}$ denotes a process that does not do anything and is used as termination symbol in a process expression.

The *input process* $c(x_1,\ldots,x_n).P'$ is a process that takes inputs at port c, which is a communication channel, and binds them to the variables $x_1,\ldots,x_n$. These variables are described by description logics expressions. In practice, it is useful to have information on the type of the communication protocol and messages transmitted over a channel. E.g., a book selling process sends the book via "HTTP" as PDF file or via "surface mail" as hard copy. The communication channel c is represented by an individual in O and refers to the information about the communication type and the partners (i.e., the two communication channel end points). The subsequent behavior of this process is defined in the process expression P' that follows.

Analogously, the *output process* $\overline{c}\langle y_1,\ldots,y_n\rangle.P'$ denotes a process that outputs the values $y_1,\ldots,y_n$ at a port c and then behaves like P'. The constants $y_1,\ldots,y_n$ are again specified by description logics expressions.

The *local process* $l(x_1,\ldots,x_n)(y_1,\ldots,y_m).P'$ performs the atomic operation l with the arguments $x_1,\ldots,x_n$ and produces outputs $y_1,\ldots,y_m$. The arguments and the outputs are again described by description logic expressions. The operation is regarded to be atomic in order to allow the modeler to abstract from functionality provided by l. This is necessary if it is not desired or of no additional value to reveal the precise implementation of l. The functionality is characterized by the changes on the individuals introduced by this operation, that is the set of changes between the knowledge before and after the invocation of the local process l. These changes performed by l also represent the dependencies of the outputs on the arguments. The *composition* $P_1 \parallel P_2$ consists of the two processes P_1 and P_2 acting in parallel.

The *summation* $[\omega_1]P_1 + [\omega_2]P_2$ denotes a choice of one of the alternatives P_1 or P_2 guarded by conditions ω_1 and ω_2, respectively. Each condition is a Boolean query that is evaluable to either true or false using the knowledge that is available at this stage of the described process. If several conditions can be evaluated to true, then only one subsequent process is chosen non-deterministically from the set of subsequent processes for which the condition holds.

Lastly, $@A\{y_1,\ldots,y_n\}$ denotes the *invocation of an agent identifier*, which represents a named process expression. The

arguments $y_1, \ldots, y_n$ of the agent identifier A correspond to the arguments $a_1, \ldots, a_n$ of a process P if P is described by the process expression $\Phi(a_1, \ldots, a_n)$ and the agent identifier A with arity n is defined as $A(y_1, \ldots, y_n) \stackrel{def}{=} P$. The concept of agent identifiers can be used to embed processes recursively. For example, the invocation of a Web service can be expressed with an agent identifier invocation where the base URI of the service is the agent identifier A and the service parameters correspond to the arguments of A.

2.3 Description of Non-functional Properties

Non-functional properties (NFPs) are part of semantic service descriptions and supplement the behavioral description. In contrast to functional descriptions, NFPs describe manifold quality attributes of services. We model NFPs of a service or a workflow as described above with SPKI certificates, more specifically with a semantic extension of SPKI certificate in order to enable automatic reasoning about certifiable property names despite heterogeneity in their syntax. In a distributed environment without central control, providers or users of a service or a workflow can issue name certificates to the service or the workflow. A name certificate roughly certifies certain value of property to a service or a workflow. Furthermore, users or providers can issue delegation certificates to other users and providers to empower them for issuance of name certificates. Any users or providers can build their trust in other user and providers independently and there is no notion of global trust. Depending on such a trust policy of a user the significance of the NFP values of a service or a workflow is determined. For more details about semantic extension of SPKI certificates, access and trust policies, we refer to [5].

3 PROPERTIES OF THE DESCRIPTION LANGUAGE

Adjacent to the introduction of the semantic modeling language in the previous section, we now want to elaborate on the attributes of the presented service description language and focus on the benefits that apply for the various Grid communities. We already analyzed the requirements on a language for modeling services and workflows in the Grid by an investigation of the applications and usage scenarios of services in different Grid communities [4]. The planned domain independent usage scenarios based on semantic service descriptions can be summarized with retrieval, ranking, composition, and the execution of services and workflows. In the remainder of this section, we discuss the essential properties of the description language for each domain independent usage scenario.

Retrieval denotes the identification of matching services, e.g., atomic Web services or already composed workflows. Therefore a formal specification of the demand (request) is required to express which properties of a service are required and which properties the service must not have. The retrieval component then allows for the reuse of existing service and workflows. *Ranking* deals with ordering the retrieval results according to the user's preferences. That is, if several services match a given request with respect to functional and non-functional requirements, preferences specify which services are better than others. Thus, ranking allows for a reasonable service selection from the retrieval results. *Composition* creates new workflows, i.e., complex services, from existing services and workflows. The composition complements retrieval as it can be possible to compose ad hoc workflows that fulfill the request and is composed out of existing services and workflow. The *Execution* uses the information captured by the service description language and invokes the services and workflows likewise.

3.1 Properties Concerning Retrieval

- The language allows for the specification of complex behavior, which includes types of input and output parameters, the relationships among them as well as the choreography and orchestration. Especially, an explicit modeling of the relation between output and input parameters allows to model and reason about the changes that are introduced by a service execution or a local operation.
- The description language supports mappings between different terms with equivalent meaning because it is likely that different providers will use different vocabularies. Such mappings are also inevitable as the requester typically uses a vocabulary that differs from the vocabularies used in service descriptions.
- User access rights can be explicitly modeled in order to grant or prohibit access to services and workflows for particular users or groups. For instance, not all workflows and services that are available in the Grid should also be read by all users or groups. Thus different different Grid communities can restrict the access to their services.
- The description of the communication protocols (e.g., HTTP, mail) allows a differentiation between similar services operating with different protocols. Protocol wrappers that allow to invoke services with other protocol are also described if applicable. Such wrappers as well as scripts that perform small tasks and typically are not published as Web services are modeled as services in the presented description language, too. Thus, transformation steps that allow to invoke services with altered interface or protocol are also reflected in a workflow description.

3.2 Properties Concerning Ranking

- The service description formalism also captures non-functional aspects of a service and workflows. Non-functional properties are first class citizens when different services are compared with each other and an ranking of the service's suitability is computed based on user preferences. Ranking can be applied to choose the best service(s) from a set of discovered services that, ensured by the retrieval component, satisfy the functional requirements.
- Non-functional properties describe manifold attributes like the quality of the service, which in turn can be modeled by a vast amount of properties like price, response time, availability, etc. Since the selection of service is based on the ranking of the services, the authenticity of the properties and their value is crucial for the user. Therefore, our description language includes a mechanism to certify service properties in a decentralized way.

3.3 Properties Concerning Composition

- The description language covers atomic services and complex services uniformly. Therefore, a distinction between the behavior of services and workflows as a composition of atomic and complex services can be left out. Both types are semantically described in the presented description language and differ only in their complexity of the described behavior. Therefore, any service described with our language can be considered for a reuse in a new composition. Consequently, the properties of the language apply to both services and workflows likewise.
- The data flow within a workflow is modeled semantically by communication activities like input and output operations. Parameters and their types and relationships as well as the communication channel with the communication endpoints and the communication protocol are contained in the description. The control flow within compositions can be modeled by language elements such as sequence and choice.

3.4 Properties Concerning Execution

- In contrast to most other workflow languages that model a centralized execution, the presented description language allows for a decentralized execution. This property is inherited by the mobility of the π-calculus, i.e., communication channel information can be exchanged as data objects between different services.
- Actors and communications between them are included in the service descriptions. This information is necessary for a correct execution. It enables the description of human tasks in a composition, for instance the input of text in a Web form and the subsequently transmitted information.
- In compositions, communications with external partners may also take place. Therefore, the description language allows to describe services that communicate with other services. E.g., a user provided delivery address can be forwarded to another shipping service without requiring further user interaction.

3.5 Additional Unstructured Properties

- Full-text description elements of service descriptions attach metadata information. It may contain information about context sensitivity, which is exploited when a search request contains the description of the user's context.
- The service can be classified optionally using an taxonomy that is modeled in an ontology. Classifications are often useful as they are intuitive for users and can be exploited in certain settings in order to improve computational performance.

4 USAGE OF SEMANTIC SERVICE & WORKFLOW DESCRIPTIONS

After we introduced the semantic description language that can be used to describe services and workflows likewise, we want to show in this section how the formalism can be applied for practical use, for instance in the German Grid settings. We will introduce a conceptual architecture of tools that build on the presented language and provide the typically aimed functionalities like search, composition, and execution of services and workflows.

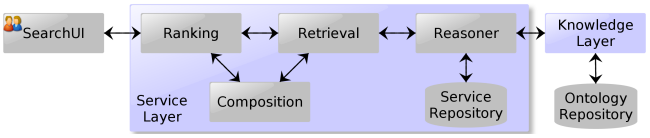

Fig. 2. Conceptual architecture for a usage of service and workflow descriptions, drawing on the example of a search scenario.

We created a Java API for the description language presented in Section 2 that allows us to develop tools that use these descriptions. The conceptual architecture of tools based on the description language API is depicted in Figure 2. In the following we explain the depicted components. The *knowledge layer* manages the knowledge that is necessary to describe services. The knowledge is expressed in ontologies that are stored in the ontology repository within the Grid. We use an integrated Rule Oriented Data System (iRODS [16]) system to organize distributed data and its meta data in the Grid. The layer offers the following functionality: (i) save ontologies, (ii) search ontologies, (iii) extraction of ontologies based on existing information, (iv) mapping between ontologies, and (v) derive new information and new knowledge from existing ontologies.

The *service layer* represents an infrastructure for the management, composition, and execution of services. In the WisNetGrid project we consider complex workflows for information processing and knowledge creation. This layer comprises the *service repository*, i.e., a registry for services that also manages the storage of the service descriptions in the Grid. It allows to create, read, update, delete descriptions. Also, users can lock descriptions or get notifications upon changes by other users.

The *reasoner* allows to access and conclude from the semantic descriptions in conjunction with the ontologically described knowledge managed by the knowledge layer. The ontology repository also manages the ontologies that accompany the service descriptions, e.g., describing parameter types and relationships. The *composition* uses *retrieval* and *ranking* in order to retrieve existing services and workflows that are available in the repository and combines them into new workflows that can be encapsulated, described, and published as services again.

The *search user interface* serves as the interface for users to access services and workflows. Here, we abstract from the concrete subtasks (retrieval, ranking, composition) and subsume them under the term search. A Web browser based interface that was built on top of the Oryx process modeling infrastructure[1]. This open source tool excels at high extensibility. We integrated the process editor with the WisNetGrid service layer, i.e., services and workflows available in the repository can be accessed in this tool. Users can explore the repository, formulate retrieval queries and ranking preferences within this tool. As we introduced semantic service descriptions that model their behavior, a request formalism based on temporal logics can be applied to express constraints on the service behavior. E.g., the μ-calculus can be used to express requests on the process behavior as we have shown in [2].

[1] Oryx is available at http://bpt.hpi.uni-potsdam.de/Oryx

Compositions can be easily create semi-automatically, that is, the composition assists the user during a manual workflow modeling task in the editor. Therefore, we defined a stencil set for the description language we presented in Section 2. By this, a user can manually create workflows using data and control flow constructs of the description language in a visual manner. Activities can be added, rearranged, and interconnected with easy drag and drop actions. Further properties for selected objects and the entire service are added in an additional widget.

As the process modeling tool is designated for editing process description without additional semantic information, e.g., BPMN, Petri Nets, among others, we also added the ability to add semantic information to workflows. OWL Ontologies can be imported and explored visually in a tree-shaped visualization based on concepts and relations besides the process model[2]. The process can be enriched with semantic information by dragging concepts and relations from the ontology view into the process model and drop it over a particular element, e.g., an input parameter. Finally, changed service and workflow description are directly saved into the repository.

5 INTERLOGGRID SCENARIO

In this section, we show an example workflow that is composed of electronic transport assisting. We have applied the presented workflow modeling approach to services and workflows of the logistics domain. The InterLogGrid project[3] investigates the application of the D-Grid infrastructure to the logistics domain. Electronic and real world transport services are typically composed into workflows that represent the freightage of goods including multi modality.

Atomic logistic services like accounting, route optimization, and the actual transport of freight from one location to another are described using OWL-S Profile [14]. That means, their functionalities are described by the four elements: sets of input and output parameters, a pre-condition, and an effect description. OWL-S Profile suffices the description of atomic activities, i.e., activities that do not interact with the user during the execution. The only communication of an atomic activity takes place within the provision of the input parameters at the invocation and the return of the output values to at the end of the execution.

Example. The workflow models a tracking process U, which can be embedded in further workflows by an process invocation $@U\{expr, log\}$. In our example, U receives the two arguments *expr* and *log* that represent the user's choice for tracing non standard freight and is composed out of different services that are invoked based on the user input. The process expression P_{std} is given below to foster readability.

$$U(expr, log) \stackrel{def}{=} [\neg(expr \vee log)]P_{std} + [expr]@U_{expr}\{\} + [log]@U_{log}\{\}$$

Standard freight tracing is started if the user does not select express or logistic tracing, i.e., the condition $\neg(expr \vee log)$ is true. Then, the user enters at port p a shipping number n. The local operation L_{getID} maps n to the freight identification code *id*. The tracing is invoked with *id* and the communication channel q to be used to return results as arguments. It retrieves the status *stat* of the freight. The process U_{std} waits for a return of the status at port q. Finally, the status is returned to the user by an output operation at port c.

$$P_{std} = p(n).@L_{getID}(n)(id).\overline{c}\langle id\rangle. @U_{trace}\{id, q\} \parallel q(stat).\overline{c}\langle stat\rangle.\mathbf{0}$$

Depending on the user selection, U_{expr} and U_{log} are invoked as subprocesses in U. For tracing an express freight, the user has to provide the freight type *type* and the number *n* first. Then, based on the freight type, the local operation L_{lookup} determines in which system the shipping number *n* can be mapped to the unique ID *id*. Here, the result *x* of this local operation is the communication port that is used to query the freight id. The process U_{expr} sends *n* to the external provider and waits for the reply *id* on the same channel. The freight is traced and its status *stat* is returned to the user in the subsequent steps of U_{expr}.

$$U_{expr} = p(type, n).@L_{lookup}(type)(x).\overline{x}\langle n\rangle.x(id). @U_{trace}\{id, q\} \parallel q(stat).\overline{c}\langle stat\rangle.\mathbf{0}$$

Dynamically determining the communication channel to be used as shown in this example is an advantage of the underlying π-calculus. The description of the third option of tracing logistic freight is omitted due to the space constraints.

6 RELATED WORK

Quite a few approaches for describing composite Web services semantically have been proposed in recent years, the most known of them is perhaps OWL-S [17]. However, the lack of formal semantics of the OWL-S Process Model makes it hard to develop automatic verification (required for search) of service behavior described with OWL-S. In the following we investigate the relationship of different existing approaches to our service modeling approach.

The Web Service Description Language (WSDL [6]) allows for a description of exchanged messages of services with XML Schema. Further, the interface element relates inputs, outputs, and failures to the operations of the services. The bindings element describes how it can be invoked. However, the syntactical description model prevents a formal description of the service functionality. Semantic Annotations for WSDL and XML Schema (SAWSDL [9]) allows to attach ontologically described concepts to the elements of a WSDL service description. This is similar to our approach as we also link service descriptions to ontologies in order to gain machine interpretable service descriptions, which in turn can improve the task of automated search and composition. Universal Description Discovery and Integration (UDDI [7]) is an XML based registry and has been applied to WSDL [8] and SAWSDL [11] service descriptions. UDDI does allow to attach any kind of meta data to practically any elements of the data model (tModel). Although this could be utilized to add semantic information, the meta data is not considered during the discovery that UDDI provides. Also, SAWSDL lacks the ability to explicitly model the behavior, which does not allow to formulate temporal constraints on the behavior which is relevant for compositions.

[2] We allow to perform simple changes on ontologies with the tool directly, but for a change of complex axioms we rely on external OWL editors.

[3] InterLogGrid is a research project funded by the German BMBF. More information is available at http://interloggrid.org.

OWL-S, the Web Ontology Language for Web Services, is an ontology describing the vocabulary used to describe Web services formally. It is similar to the presented description language to some extent. For example, OWL-S provides the concept Process and descriptions comprise input, output parameters, Precondition and Result for the description of the functionality, Participants among others. Also, control constructs of the process like Sequence, Split, and Choice are defined in OWL-S.

WSDL, SAWSDL, and OWL-S cannot model access rights as well as the trustworthiness of service properties within the service description. Both language properties were identified as important for the Grid scenarios. OWL-S provides the possibility to administer service ratings, but service consumers still have to verify them manually. The certified properties that can be included in the presented description language trustworthy, their validity is proven, i.e., can be evaluated, by the certificate.

Finally, the expressivity of our description language is higher than the one of WSDL and SAWSDL and not less expressive than compared to OWL-S. In the latter case we can not judge precisely as the semantics of the OWL-S process model was not fixed in the specification [14]. But since the π-calculus is Turing-complete, we can conclude that OWL-S services can be also completely modeled in our description language.

The need for a semantic description of non-functional properties in addition to the semantic description of the service functionalities was already motivated in [12]. The authors motivated their approach with an increased potential for automation of tasks like discovery as shown in [13, 10].

7 CONCLUSIONS AND FUTURE WORK

We presented a semantic description formalism for services and workflows that can be applied in the settings of the German Grid projects. We introduced a formal model of services, which captures functional as well as non-functional properties of services. This information can be modeled in description logics and allows for an application of semantic technologies to reason about service properties and allows us, for example, to remove the burden of different vocabularies used among different service providers.

Furthermore, we elaborated on the description of the behavior of services. We combined the π-calculus process algebra for expressing the process structure with a semantic description of the objects occurring in the process. Due to the combination of both, the presented service description approach is able to automatically reason about functional and non-functional aspects, which increases the potential of the targeted service reuse. As we motivated in the beginning of this work, the reuse of services in conjunction with a large amount of available service descriptions can be fostered if we develop automatic retrieval and composition techniques.

As the modeling language was the focus of the current work, we highlighted the advantages of our description language and showed how it can be applied in a scenario that enables the reuse of available services. The example workflow taken from the InterLogGrid use case showed a practical and realistic scenario for the usage of service and workflow descriptions. Based on the current status, we plan to refine our existing matchmaking techniques in order to efficiently deal with composition and retrieval tasks for a larger amount of services. The classification of services and the application of indexing techniques are promising techniques to gain efficiency. Also, the Oryx based user interface is continuously developed in order to support users in modeling services, create service requests and compositions as much as possible.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the funding for the project WisNetGrid (http://wisnetgrid.org) by the Federal Ministry of Education and Research BMBF, Germany.

REFERENCES

[1]S. Agarwal, S. Rudolph, and A. Abecker. Semantic Description of Distributed Business Processes . In *In Knut Hinkelmann, Andreas Abecker, Harold Boley, John Hall, Martin Hepp, Amit Sheth, Barbara Thönssen, AAAI Spring Symposium - AI Meets Business Rules and Process Management*, 2008.

[2]Sudhir Agarwal. A Goal Specification Language for Automated Discovery and Composition of Web Services. In *International Conference on Web Intelligence (WI '07)*, pages 528–534, Silicon Valley, California, USA, November 2007.

[3]Sudhir Agarwal. *Formal Description of Web Services for Expressive Matchmaking*. PhD thesis, Universität Karlsruhe (TH), Institut AIFB, D-76128 Karlsruhe, 2007.

[4]Sudhir Agarwal, Patrick Harms, Carolin Michels, Silke Molch, and Eva Radermacher. D 3.2.1 Anforderungen an die semantische Beschreibungssprache fr Web-Dienste, Dezember 2009.

[5]Sudhir Agarwal and Barbara Sprick. Specification of access control and certification policies for semantic web services. In Hannes Werthner Kurt Bauknecht, Birgit Pröll, editor, *6th International Conference on Electronic Commerce and Web Technologies*, volume 3590 of *Lecture Notes in Computer Science*, pages 348–357, Copenhagen, Denmark, August 2005. Springer.

[6]Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web Services Description Language 1.1, 2001.

[7]Luc Clement, Andrew Hately, Claus von Riegen, and Tony Rogers. UDDI Spec Technical Committee Draft 3.0.2. OASIS Committee Draft, OASIS, 2004.

[8]John Colgrave and Karsten Januszewski. Using WSDL in a UDDI Registry. Technical report, OASIS, 2004.

[9]Joel Farrell and Holger Lausen. Semantic Annotations for WSDL and XML Schema, 2007.

[10]Martin Junghans and Sudhir Agarwal. Web Service Discovery Based on Unified View on Functional and Non-Functional Properties. In *ICSC*. IEEE Computer Society, 2010.

[11]Dimitrios Kourtesis and Iraklis Paraskakis. Combining SAWSDL, OWL-DL and UDDI for semantically enhanced web service discovery. In *Proceedings of the 5th European Semantic Web Conference on The Semantic Web: Research and Applications*, ESWC'08, pages 614–628, Berlin, Heidelberg, 2008. Springer-Verlag.

[12]Kyriakos Kritikos and Dimitris Plexousakis. Requirements for QoS-based Web Service Description and Discovery. *Computer Software and Applications Conference, Annual International*, 2:467–472, 2007.

[13]Kyriakos Kritikos and Dimitris Plexousakis. Mixed-Integer Programming for QoS-Based Web Service Matchmaking. *IEEE Transactions on Services Computing*, 2:122–139, 2009.

[14]David Martin, Mark Burstein, Erry Hobbs, Ora Lassila, Drew Mcdermott, Sheila Mcilraith, Srini Narayanan, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, and Katia Sycara. OWL-S: Semantic Markup for Web Services. Technical report, November 2004.

[15]Robin Milner, Joachim Parrow, and David Walker. A Calculus of Mobile Processes, Part I + II. *Journal of Information and Computation*, 100:1–77, 1992.

[16]Arcot Rajasekar, Reagan Moore, Chien-Yi Hou, Christopher A. Lee, Richard Marciano, Antoine de Torcy, Michael Wan, Wayne Schroeder, Sheau-Yen Chen, Lucas Gilbert, Paul Tooby, and Bing Zhu. *iRODS Primer: Integrated Rule-Oriented Data System*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2010.

[17]Katia P. Sycara. Automated discovery, interaction and composition of Semantic Web services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(1):27–46, 2003.