WS-UML: A UML PROFILE FOR WEB SERVICE APPLICATIONS

Dhikra Kchaou
Mir@cl Laboratory, Faculté des Sciences Economiques et de Gestion, Sfax University Sfax, Tunisia
dhikrabi@yahoo.fr
Nadia Bouassida
Mir@cl Laboratory, Institut Supérieur d'Informatique et de Multimédia, Sfax University Sfax, Tunisia
Nadia.Bouassida@isimsf.rnu.tn
Hanene Ben-Abdallah
Mir@cl Laboratory, Faculté des Sciences Economiques et de Gestion, Sfax University Sfax, Tunisia
Hanene.BenAbdallah@fsegs.rnu.tn

Web service technologies have specificities that must be accounted for at the design level. The first part of this paper presents a web service design language WS-UML that is a UML profile. WS-UML extends UML with graphical annotations to express the specific concepts of web services, e.g., composition, security, location, trace of execution. The second part of this paper presents a meta-model describing the essential elements of web services. Finally, the paper illustrates the usefulness of the WS-UML notation using a Tourist information service example.

Web service, Profile UML, security, composition.

1. INTRODUCTION

Web services (WS) are emerging as a new development paradigm for distributed applications over the Internet and the Web. Despite their advantages, using WS to develop distributed applications remains a complex task. The complexity stems from the various aspects that a web service comprises and that must be specified at the design level, e.g., service discovery, composition, security, mobility, etc.

To address this complexity, several design notations have been proposed (*cf.*, Provost [13], Ortiz and Hernandez [12], Belouadha and Roudiés [2] [3], Skogan [15]). These notations tried to: 1) fix the essential concepts specific to the WS domain, and 2) express the identified concepts through particular notations.

To compile the list of concepts, these propositions relied on different WS views (organizational, compositional, security ...) and standards (WSDL [4], BPEL [16], OASIS RBAC [8] ...). On the other hand, to define their particular notations, these propositions extended the standard design language UML. The extensions are required since UML does not clearly account for the following WS specific concepts:

- Security, which represents one of the major problems encountered during the access process to any WS. On the one hand, the client needs to certify a WS and be certain of its reliability, on the other hand, the supplier has to be sure that the customer has the permission to reach the WS.
- Composition: the execution of a WS often requires the call of several other web services, this is called web service composition. Several composition languages were proposed like BPML [10], XLANG [14], WSCI [1], BPEL4WS [16], etc.
- Quality of a service: The use of a WS is faced to many problems such as the number of services that evolve very quickly on the web and the reliability of used services. In fact, the growing number of service makes their selection a costly and non trivial operation. To remediate to this problem an automatic selection based on criteria such as the quality of service (QoS) must be foreseen to facilitate this task as the price, execution time, reliability, etc [3].

- Community of a service and functionality: In many cases the collection of services possesses the same functionalities without necessarily having the same quality of service; this concept allows replacing a service whose call poses a problem due to a change of behavior [3].
- Execution trace: the follow-up of service executions is a requirement in the domain of the process management. Companies need to follow their activities to be able to explain failings or breakdowns and to review their practices in order to increase their efficiency and to improve the satisfaction of clients [5].
- Location: a client should be able to locate a service and the company of the service, and find out how to invoke the service. This information facilitates searching the service and trusting the company that offers this service.

Overall, since they tackled the WS from different views, none of the proposed notations expresses all of the above concepts. For instance, none of them considers the WS location and execution trace.

Clearly identifying (both finding and denoting) the security, composition and location aspects is essential in understanding a WS application. This motivated our work in proposing a WS design language, called WS-UML.

In defining the WS-UML language, we took special care about the following through points: 1) the language must be a UML profile that increases the expressiveness of UML with WS specific concepts; 2) it should facilitate the implementation of a WS; and 3) it should guide the client to choose the appropriate and reliable service and the supplier to supervise and to control the service execution. To ensure these properties, the proposed language adds tags and graphical annotations to UML diagrams; the extensions help to distinguish visually between functional and non-functional properties of a web service.

The remainder of this paper is organized as follows. Section 2 first overviews proposed languages and, then, collect the set of WS specific concepts to be treated at the design level. Section 3 presents the WS-UML metamodel. Section 4 details the WS-UML Profile and illustrates it through an example. Section 5 concludes with a summary and outlines future works.

2. RELATED WORK

Currently, WS are described with the Web Service Description Language (WSDL) [4]. However, WSDL documents can be difficult to understand for service developers especially in terms of what the WS really does and how it ought to be composed. These WSDL limits motivated several propositions of UML-based languages to express the interface, behavior and constraints of WS in a more understandable way than WSDL [7]. We next review the most important WS design languages that are based on the *de facto* standard language UML. As summarized in Table I, the proposed languages differ in terms of their WS concept source which reflects their design view/perspective.

2.1 WSDL-centered languages

Provost [13] proposes a UML profile for WS in order to model WS before generating their corresponding WSDL files and implementation. This profile extends the UML class diagram with a subset of WSDL concepts: Service, Port, PortType, Operation, Message, Part and Binding. It uses stereotyped classes to present these concepts and tags to mark, essentially, the URL context of a service and/or its ports.

However, this profile does not take into account all WSDL concepts specific to the WS domain.

2.2 Security-centered languages

Approaching WS design from a security angle, Ortiz and Hernandez [12] considered even fewer WSDL concepts. They propose a simple UML profile based on the Service Component Architecture (SCA) where services are modelled as components. The WS Component can be linked by a given interface where the WS operations could be included; each operation can be later specified in a particular type of interface.

In addition, components show the references they need to complete the behaviour successfully.

These references may be later linked to a required interface.

Furthermore, this work proposes a profile to model extra-functional properties for WS. It defines a package called *EFProperties* that shows extra-functional properties with their necessary additional attributes. The properties defined are: *LogProperty* to record invocation information, *LoginProperty* to control access to operations and *Encryption* to encrypt the WS. They are modelled in terms of stereotyped classes. For instance, to model authentication, the profile defines the stereotyped class

Login and its attributes *username* and *password*, and the stereotyped class *encryption* with its attribute *encryptionKey*. Overall, the modelled security aspects can be difficult to understand; for example, the authentication process is unclear and does not specify who provides the username and the password to access a given WS.

View		Concepts	Language
WSDL		Class UML stereotyped "WebService"	[13] [3]
		Class UML stereotyped "Port"	
		Class UML stereotyped "PortType"	
		Tagged Value relative to a service and the Port {URI="/ "}	
SCA		Component stereotyped "ServiceComponent"	[12]
		Tagged Value relative to an interface {R.uri= "Operationuri"}	
		Class stereotyped "ServiceInterface"	
Security		Note stereotyped "Login"	[12]
		Note stereotyped "Log"	
		Note stereotyped "Encryption"	
Composition	Composition & Orchestration	Class stereotyped "ServiceWebAtomique"	[3] [2]
		Class stereotyped "ServiceWebComposite"	
		Class stereotyped "Mod�leDorchestration"	
		Activity "Processus"	
		"ProcessusAtomique"	
		"ProcessusComposite"	
	Composition & Orchestration	Activity stereotyped "WebServiceCall"	[6]
		Activity stereotyped "AlternativeServices"	
		Activity stereotyped "ImmediateStep"	
		Note stereotyped "DataTransformation"	
		Activity Stereotyped "DataMapper"	
		Tagged Value relative to the activity "ImmediateStep" {DomainObject=}	
	QoS	Class stereotyped "ServiceQuality"	[3][2]
	Community & function	Class stereotyped "Community" Class stereotyped "Function"	[3] [2]

TABLE 1: overview of web service related work

2.3 Composition-centered languages

The work of Gronmo *et al.* [6][7][15] focuses on how to model web services by composing already existing services. In addition to WS composition, it models two aspects: the service and the workflow. The service model identifies services to be exposed with their interfaces and operations; the workflow model (through a UML activity diagram) identifies the control and data flows from one service to another. In addition, a WS activity has a set of tagged values: the WS provider, the URL to the WSDL file where the exact service operation to invoke is defined by the tagged values *Provider*, *WSDL*, *Service*, *PortType* and *Operation*.

The proposed notation proposes an activity with four possible stereotypes: "WebServiceCall" to present the web service operation; "ImmediateStep" to present non-web-service activities that invoke automatic operations with a tagged value called *DomainObject* indicating the internal operation to call; "AlternativeServices" to indicate that its task may be fulfilled by a set of alternative services; and "DataMapper" to present required data transformation.

Also from a composition perspective, Belouadha and Roudis [2] [3] focussed on concepts related to composition: Quality of Service (QoS) and Service Community which are important elements to help the client choose the suitable service. In fact, the QoS helps in the selection of a service when several services offer similar features but with different QoS. On the other hand, the service community deals with the problem of service changes: a collection of WS possesses the same features without inevitably having the same QoS. The proposed notation presents important

elements relative to a WS. However it lacks the WS security aspect. Moreover, it does not indicate localisation and execution trace aspects of a WS.

3. THE WS-UML META-MODEL

Our main objective in defining a UML-based meta-model is to cover the three above perspectives: WSDL for the WS interface, WS composition, and WS security. In addition, in defining each perspective, we made sure to provide for pertinent standard. The resulting meta-model illustrated in Figure 1 shows the five perspectives each represented through a package.

A WS is modeled as a UML class called *WebService*. The package *WSDL* represents WSDL elements in a way similar to the proposition of Provost [13]: a web service is defined as a collection of *endpoints* or *ports*. A port specifies an address for a binding, thus defining a single communication endpoint. A *PortType* is an abstract set of operations supported by one or more endpoints. The association class called *Binding* represents a concrete protocol and data format specification for a particular *PortType*. The class *Message* represents an abstract definition of the data being transmitted. A message consists of logical parts, each of which is associated with a definition within some system type.

The package called *Security* presents two security aspects: Authentication and Role-Based authorization. In fact, in a situation where the client and service do not share a direct trust relationship, we can use a broker to perform authentication. The broker authenticates the client and then generates a security token that the service can use to authenticate the client. The security token is always verified, but typically, the service does not need to interact with the broker to perform the verification. This is because the token itself can contain proof of a relationship with the broker, which can be used by the service to verify the token [11].

On the other hand, Role-Based Access Control (RBAC) standard [8] is used by many organizations to protect their information resources from unauthorized access. RBAC policies are defined in terms of *permissions* that are associated with roles assigned to users. A *permission* determines what operations a user assigned a given role can perform on particular information resources. Thus, the RBAC model protects activities of web services.

The package *Quality of Service* specifies a set of quality requirements on the behaviour of a WS. The aspects specified in this package help clients in the selection of the appropriate WS. To model these aspects, we use three concepts: First, an UML class called QoS that offers the cost, execution Time, Bandwidth and the reliability's degree of a service. Secondly, UML classes *Community* and *Function* are modelled to group services that possess the same functionalities. Finally, the class called *Trace* allows to follow-up the execution trace of a WS where its attributes: *ExecId* defines execution identity, *Location* defines the location of this execution, and *Status* enumerates the types to represent different status of a service: *Running*, *Suspended*, *Completed* or *Annulled*.

The package *Directory Information* is modelled to know from where the service comes and how it can be invoked. This information can be presented with the UDDI (Universal Description, Discovery and Integration) data model which includes the following main elements: *businessEntity* to represent a physical company, *businessService* to represent a service offered by a company, *bindingTemplate* to offer *Instructions* on how to invoke a service, and *TModel* to describe invocation details [9].

The package *Process* provides for WS composition modelling. It uses concepts inspired from the composition language BPEL4WS (*Business Process Execution Language for Web Services*) [16]. Each element in the BPEL4WS process is called an *activity*. An activity is either *primitive* or *structured*. The primitive activity types are: *invoke* to invoke an operation of a WS described in WSDL; *receive* to wait for a message from an external source; *reply* to reply to an external source; wait to remain idle for some time; *assign* to copy data from one data container to another; *throw* to indicate errors in the execution; *terminate* to terminate the entire permission service instance; and *empty* to do nothing.

In addition, to enable the representation of complex structures, the following structured activities can be used: *sequence* for defining an execution order; *switch* for conditional routing; *while* for looping; *pick* for race conditions based on timing or external triggers; *flow* for parallel routing; and *scope* for grouping activities to be treated by the same fault-handler [16]. The composition tie indicates that a structured activity can be composed of one or several activities. This tie is annotated to indicate the composition type. The primitive activity is also annotated to define the activity type and the service name.

FIGURE 1: Web service Meta-model

4. THE WS-UML PROFILE

A WS design expressed in the WS-UML language is composed of the following UML extended diagrams:

- A class diagram that describes several aspects relative to WS: WSDL, Security, QoS, Directory Information and Process or WS composition.
- A set of sequence diagrams that describe the security aspects such as authentication.
- An activity diagram to describe WS composition.

The extensions to the UML diagrams are described in Table 2. Note that, being a UML profile, the WS-UML language could be adopted easily by the UML community.

Example

In this section, we illustrate the WS-UML profile using the case study *touristInformation* service. This service offers two operations: *CarRenting* which offers renting a particular type of car and *DestinationInformation* which provides information about the destination city.

CarRenting is an atomic service and *DestinationInformation* is a composite WS composed of the atomic WS *WeatherInformation* that returns weather information in a destination city and the composite WS *HotelInformation* that provides the possibility of getting information about different hotels available in a given city. Finally, the *HotelInformation* service is composed of two atomic services: *HotelPrice* and *hotelLocation* service.

In Figure 2, the WS *TouristInformation* is described with the WSDL essential elements: Port, PortType (*TouristInformationInterface*), Operations (*DestinationInformation* and *CarRenting*) and parameters of these operations (*CityName*). These elements are represented as UML classes stereotyped. Service relative URI is expressed as a tagged value bound to the class stereotyped "Port".

The client claimant of the service must possess a certificate X.509, as the security token, signed with the private Key of a certificate authority, the authentication broker that is paired with the public Key of the client.

UML classes stereotyped "Session", "Role" and "Permission" express the RBAC model. In fact, the client can not access to *TouristInformationActivity* if he has not the permission.

Our web service is composed of several activities. The primitive activity consists in calling the web services "CarRenting" and "DestinationInformation". These services are executed in sequence indicated by the tagged value {COMPTYPE=Sequence}. Moreover, the WS *HotelInformation* and *WeatherInformation* which belong to the web service *DestinationInformation*, are executed in parallel as indicated by the tagged value {COMPTYPE=Flow}. Finally, the *HotelPrice* and *HotelLocation* WS constituting the *HotelInformation* service are executed in parallel.

The company that offers the WS *TouristInformation* is indicated here as a class stereotyped "BusinessEntity" called *TouristInformationService* Company. This company has a descriptive URL that gives information about it, and a list of its contacts. All instructions on how to invoke a service are indicated with the class stereotyped "BindingTemplate". The invocation details are presented in a class stereotyped "Tmodel".

Moreover, notice that the *TouristInformation* service quality is indicated by the class *TouristInformationServiceQuality*. *TouristInformation* service belongs to a service community with the function *TouristInformationFunction*. Finally, the *TouristInformationTrace* indicates that the *TouristInformation* service is performed in the *ExecutionID* *e1*, at the location "www.location.xml" with the status *running*.

5. CONCLUSION

This paper first presented the WS-UML language, a UML-based language for the specification of web services. It then presented a case study to illustrate that the UML extensions can handle the specificities of WS designs. The case study showed that the WS-UML notation facilitates the comprehension of a WS especially the security and composition aspects. In fact, none of the proposed works explicitly provides for the modeling of, for example, authentication and RBAC-based security support; in addition, only WS-UML explicitly offers composition modeling through BPEL4WS concepts. Further more, according to the presented case study, the design with WS-UML facilitates the implementation of web service especially with stereotypes and tagged values added. These preliminary advantages need to be further demonstrated through additional, industrial case studies.

Aspects	Notation	Explanation
WSDL	UML Class stereotyped "WebService"	Represents the web service as a collection of port.
	UML Class stereotyped "Port"	Encapsulates a WSDL Port that implements the PortType.
	Tagged Value relative to the Port {URI="/ "}	Annotates the service-relative URI.
	UML Class stereotyped "PortType"	Expresses a WSDL PortType component and supporting Operations, messages and parts by the usual notation for "interface" types.
Security	UML class stereotyped "AuthenticationBroker"	Represents the authentication Broker that authenticates clients by issuing to them a security token.
	UML class stereotyped "SecurityToken"	The security token is used by the client to authenticate with the service. The security token can be used by the client for a period of time that is defined by the authentication broker.
	UML class stereotyped "Client"	Represents the client claimant to the service.
	UML class stereotyped "Session"	Represents sessions created by the client to access to the service.
	UML class stereotyped "Role"	Represents responsibilities well identified in an organization.
	UML class stereotyped "Permission"	Indicates a particular access right on one or several activities of the web service.
Composition	UML class stereotyped "Activity"	Represents the small element of the BPEL4WS Process.
	UML class stereotyped "PrimitiveActivity"	Represents primitive activities of the BPEL4WS process.
	UML class stereotyped "StructuredActivity"	Represents structured activities of the BPEL4WS process.
	Tagged Value {ServiceName=...}	Used to annotate primitive or structured activity to indicate the service name called.
	Tagged Value {ActivityType=...}	Indicates the primitive activities types.
	Tagged Value {CompType=...}	Annotate the composition tie to indicate the composition type.
	UML class stereotyped "enumeration"	Used to define primitive and structured activities type.
Directory Information	UML class stereotyped "BusinessEntity"	Represents a physical company.
	UML class stereotyped "BindingTemplate"	Indicates instructions on how to invoke a service.
	UML class stereotyped "DiscoveryURL"	Represents a descriptive URL to tell as more about the company.
	UML class stereotyped "Contact"	A list of contact at that company.
	UML class stereotyped "TModel"	Represents invocation details.
QOS	Class stereotyped "ServiceQuality"	Presents service quality such as cost, execution time, etc.
	Class stereotyped "ServiceCommunity"	Regroups services that have same functionalities.
	Class stereotyped "Function"	Indicates the functionality of a service group.
	Class stereotyped "Trace"	Indicates the execution trace of a service.

TABLE 2: Extensions presented in WS-UML

FIGURE 2: TouristInformation Service profile

In addition to developing a CASE toolset supporting the proposed profile, we are currently working on three research axes. The first consists of modeling web services with more expressiveness and comprehensibility. The second is to assign Design patterns for WS. The third research axe examines other security aspects like encryption.

REFERENCES

- [1] A. Arkin, S. Askary, S. Fordin, W. Jekeli, K. Kawaguchi, D. Orchard, S. Pogliani, K. Riemer, S. Struble, P. Takacs-Nagy, I. Trickovic, S. Zimek, "Web Service Choreography Interface (WSCI) 1.0", W3C Note 8 August 2002.
- [2] F. Belouadha and O. Roudis, "Un profil UML de spcification de services web composites smantiques", CARI 2008-MAROC, pp. 537-544.
- [3] F. Belouadha and O. Roudis, « Vers un modle de spcification pour la composition de services web », Proceedings of SIIE'08, Tunisie, Fvrier 2008.
- [4] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, "Web Services Description Language (WSDL) 1.1", W3C Note 15 March 2001.
- [5] M.-C. Fauvet, M. Dumas, B. Benatallah, "Suivi d'excution de services accessibles par l'Internet", IEEE Computer Society, editor, Proceedings of ICDE'02 Conference, 2002.
- [6] R. Gronmo, I. Solheim, "Towards Modeling Web Service Composition in UML", INSTICC Press, 2nd International Workshop on web services: Modeling, Architecture and Infrastructure, Porto, Portugal, April 2004.
- [7] R. Gronmo, D. Skogan, I. Solheim, J. Oldevik, "Model-driven Web Service Development", IEEE International Conference on e-Technology, e-Commerce and e-Service, 2004.
- [8] D.K. Kim, I. Ray, R. France, N. Li, "Modeling Role-Based Access Control Using Parameterized UML Models", M. Wermelinger and T. Margaria-Steffen (Eds.): FASE 2004, LNCS 2984, pp. 180–193, 2004.
- [9] S. Komatineni, "Understanding UDDI and JAXR", Java.com, 02/27/2002.
- [10] B. List, B. Korherr, "UML 2 Profile for Business Process Modelling", Austrian Federal Ministry for Education, Science, and Culture, and the European Social Fund (ESF) under grant 31.963/46-VII/9/2002.
- [11] Microsoft, "Web Service Security Scenarios, Patterns, and Implementation Guidance for Web Services Enhancements (WSE) 3.0", <http://msdn.microsoft.com/en-us/library/aa480545.aspx>, December 2005.
- [12] G. Ortiz and J. Hernandez, "Toward UML Profiles for web services and their Extra-functional Properties", IEEE International Conference on Web services (ICWS'06), 2006.
- [13] W. Provost, "UML for Web services", XML.com, August 5, 2003.
- [14] S. RAMPACEK, « Smantique, interactions et langages de description des services web complexes », Thse de Doctorat, pp. 35-38, 10 novembre 2006.
- [15] D. Skogan, R. Gronmo and I. Solheim, "Web service Composition in UML", Proceeding of the 8th IEEE Intl enterprise Distributed Object Computing Conf, EDOC 2004.
- [16] P. Wohed, W. M.P. van der Aalst, M. Dumas, A. Hofstede, "Analysis of Web Services Composition Languages: The Case of BPEL4WS", I.-Y. Song et al. (Eds.): ER 2003, LNCS 2813, pp. 200–215, 2003.