Journal of Theoretical and Applied Information Technology
20th January 2014. Vol. 59 No.2
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A MULTI-ASPECT RULE BASED MODEL FOR WEB SERVICES OFFLINE COMPOSABILITY

HAJAR OMRANA, FATIMA ZAHRA BELOUADHA, OUNSA ROUDIES
Computer Science Department, Ecole Mohammadia d'Ingénieurs (EMI),
Mohammad V University- Agdal,
Rabat, Morocco.
E-mail: hajaromrana@gmail.com, belouadha@emi.ac.ma, roudies@emi.ac.ma

ABSTRACT

Composition is still one of the most challenging key design goals of Web services technology. While they are designed to be aggregated and work with each other, most of existing Web services are developed independently and uses different standards. Detecting all the incompatibilities of Web services composability before processing the composition would increase the efficiency and correctness of this latter considerably. In this direction, we propose a multi-aspect Web services composability model, aligned with WSDL 2.0, SAWSDL and WS-Policy 1.5 standards, which addresses a set of functional, non-functional, contextual, data structure and technical composability rules to check whether to or more web services operations can concretely interact with each other.

Keywords: *Web Services Composability , Dynamic Composition, WSDL 2.0, SAWSDL, WS-Policy.*

1. INTRODUCTION AND MOTIVATION

Dynamic Web Services composition is a complex process that involves several activities such as discovery, composability, selection, etc. Actually, Web Services composability aims to check whether two or more services or operations can interact with each other to build complex services. It represents a crucial step on which depends the efficiency of the overall Web services composition process.

Despite active research [1, 2, 3, 4], composability paradigm is still not deeply investigated in the literature. Conventionally, composing two Web services S1 and S2 is finding, at least, S1 operation and S2 operation, when the output parameters of the first can cover the input parameters of the second. In large body of studied research, the connection of the two operations is based on the semantic and/or syntax matching of these input and output parameters, and do not address all the non-functional, data structural, contextual or technical background information related to the service description. For example, non-functional properties are frequently used to select the best composition plan, when some of them should be considered in composability phase even before building composition plans. Likewise, other contextual or technical properties such as communication protocols, etc.., should also be handled in order to obtain efficient composition plans.

Additionally, several composition solutions proposed in the literature remain significantly related to the structure of the adopted semantic model. The heterogeneities, which may exist between the structures of two different semantic models, make the matching of Web services properties, as well as their composition very complex. For example, inputs and outputs in OWL-S [5] are respectively considered as pre-conditions and post-conditions in WSMO [6]. In other words, the proposed composition solution stays limited to the services described by the semantic model used.

We think that in spite of the efforts invested and the maturity of the techniques and mechanisms adopted to deal with dynamic Web services composition issues, the major limit relates, on the one hand, to the dependence of the proposed composition solutions of the semantic description models used, and on the other hand, to the considered composability level which remains an abstract level emphasizing the matching of inputs and outputs of services, and not treating the various composability checking facets between two or several services to be connected.

To ensure composition efficiency, Web services composability should actually check the coherence of a set of properties relating to the Web services description aspects as functional, non-functional, technical, data structure and contextual aspects (behavior aspect is out of scope). The present work proposes a multi-aspect composability model for Web services, aligned with the W3C specifications WSDL 2.0 [7], SAWSDL [8] and WS-Policy [9]. This model captures functional, non-functional, data structure, contextual and technical properties of a Web service on syntactic and semantic levels, which are relevant for the offline composability process. The proposed approach is based on a set of composability rules that check the possibility of interconnecting two operations according to each aspect considered, to detect possible heterogeneities and avoid unexpected failure at runtime. This should thereafter contribute to reach efficient and executable composition plans.

The paper is structured in five main sections. In Section 2, we introduce the key concepts' definitions of our composability model. In Section 3, we present the defined multi-aspect offline composability rules. Section 4 presents the overall offline composability phases. Finally, Section 5 discusses and concludes our work.

2. MULTI-ASPECT WEB SERVICE OFFLINE COMPOSABILITY MODEL

Undoubtedly, the composability process effectiveness remains strongly dependant on the richness of the collected information regarding the services to be composed, but also on the coverage of the various descriptive aspects of these services. Web services description W3C standards meet these conditions. Moreover, they constitute formalisms widely adopted by the Web services community and the data-processing industry.

2.1 Offline and Online Composability

Generally, dynamic Web services composition of is initiated by a customer request defining the inputs filled and the outputs expected. The offline composability, subject of the present article, is defined as a composability process phase, which is carried out prior to the dynamic composition of services. In order to optimize the response time of the dynamic composition process at runtime, this phase allows identifying previously the services' operations, which can be composed considering functional, non-functional, technical, data structure and contextual aspects.

As regards the online composability, this phase is carried out during the dynamic composition of services (at runtime). It aims at checking if two operations, deemed to be composable during the offline composability process, can always be connected to produce an executable composite service within the context of the user's request (e.g. user's preferences).

Finally, the offline composability releases the runtime dynamic composition process from enormously time-expensive tasks. It allows constituting in advance a composable operations' s register, which can be accessible and re-used to treat any request, since the applied composability conditions relate to the descriptive properties of services remain independent of the dynamic constraints of the users' requests.

2.2 WSDL, SAWSDL and WS-Policy properties: Key Concepts and Definitions

To deal with structures' heterogeneities of various semantic Web services models, our approach is mainly based on the use of W3C standards WSDL 2.0, SAWSDL and WS-Policy 1.5 for the description of Web services.

First of all, WSDL specification is mainly adopted by major industry leaders to describe their developed Web services, soon as it is also used by semantic models (eg OWL-S and WSMO) in the grounding mechanism [5], to make their semantic services consumables. The technical details as transmission protocol, binding information, etc., present in the WSDL files are crucial to the execution of the composition plans.

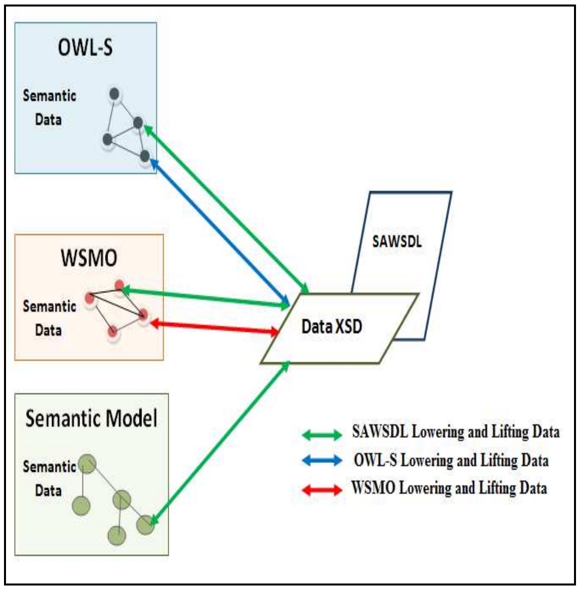
OWL-S
Semantic Data
WSMO
Semantic Data
Semantic Model
Semantic Data
SAWSDL
Data XSD
SAWSDL Lowering and Lifting Data
OWL-S Lowering and Lifting Data
WSMO Lowering and Lifting Data

Figure 1: SAWSDL Annotations Mechanisms

Also, to describe functional capabilities in semantic level, SAWSDL annotations remain independent of the semantic model used and ensure a unified structure to compare semantic properties of services described by two different semantic models. Both attributes *LiftingSchemaMapping* and *LoweringSchemaMapping* provided by SAWSDL make the matching between data structures of XML messages and associated semantic models possible [10] (Figure 1). Its main advantage is its extensibility and its compatibility with WSDL. We have explored the SAWSDL flexibility allowing to annotate the operation's inputs and outputs by ontological instances, in addition to ontological concepts, in order to cover contextual properties [11]. In our model, each operation's parameter is annotated by a semantic concept that describes the parameter's meaning , and by a list of semantic instances specifying the parameter's context.

While the description of functional properties (such as inputs, outputs, preconditions, effects, etc.) is mandatory to invoke a service, the specification of non-functional properties is necessary to complete the service description. NFP constitute all Web service characteristics other than its functional capabilities [12]. They are related to different domains such as the security, the quality of service (QoS) or general characteristics. WS-Policy is an extensible language based on XML language and used to describe and communicate Web services' strategies (policies). A policy can be an aggregation of policies alternative and be attached to an endpoint, message or operation, etc. To describe a non-functional property, WS-Policy employs assertions embedded in policies alternatives. All assertions related to a non-functional domain is defined using XML schemas. In our previous work [13], we proposed to enrich these XSD elements by SAWSDL tags to enable describing semantic Web services policies using different semantic models.

This section highlights the main concepts and definitions which are used in our composability model [14]. Operation forms the elementary functionality of a Web service involved in the composition process. It represents the main concept of the proposed composability model. As shown in the definitions given below, we formalize them as tuples of data according to W3C standards: WSDL 2.0, SAWSDL and WS-Policy 1.5.

- ***Definition 1 ($Asser_{ikmpq}$)***

An assertion *q* belonging of an alternative policy *p* attached to a policy *m* associated to an endpoint *k* related to an operation *i*, noted $Asser_{ikmpq}$, is a tuple (*assertSemConceptUri*, *assertContextUriList*, *optional*) where :

✓ *assertSemConceptUri* refers to the URI of the semantic concept describing the assertion $Asser_{ikmpq}$.

✓ *assertContextUriList* refers to the list of URIs of semantic instances informing about the assertion's context $Asser_{ikmpq}$.

✓ *optional* is a Boolean attribute. If the value of this attribute is "false" then the assertion is required , else then the assertion is optional.

- ***Definition 2 ($Policy_{ikm}$)***

A policy *m* associated to an endpoint *k* related to an operation *i*, noted $Policy_{ikm}$, represents any policy attached, either to the endpoint *k*, to the operation *i*, to the binding or service associated to the considered endpoint. It is defined by a tuple (*id*, *policyLevel*, *policyAl*) where :

✓ *id* refers to a policy key.

✓ *policyLevel* specifies the policy level. Its value belongs to the enumeration { Endpoint, Binding, Service, Operation}.

✓ *policyAl* refers to the list of alternative policies *p* attached to the policy $Policy_{ikm}$, noted $PolicyAl_{ikmp}$.

- ***Definition 3 ($MessagePolicy_{ijkm}$)***

Message policy *m* associated to a parameter *j* of an operation *i* accessible through an endpoint *k*, noted $\boldsymbol{MessagePolicy_{ijkm}}$, is defined by a tuple (*id*, *MessagePolicyAl*) where :

✓ *id* refers to the message policy key.

✓ *MessagePolicyAl* refers to the list of policies alternative *p* associated to the message policy $\boldsymbol{MessagePolicy_{ijkm}}$.

- ***Definition 4 (E_{ik})***

An endpoint *k* of an operation *i*, noted $\boldsymbol{E_{ik}}$, is the endpoint from where an operation *i* can be accessed. It is defined by a tuple (*uri*, *transportProtocol*, *policy*) where.

✓ *uri* indicates the endpoint's URI

✓ *transportProtocol* designs the supported binding transmission protocol used by the operation *i* (e.g., SOAP and HTTP).

✓ *policy* refers to the list of policies attached to the endpoint *k*.

- ***Definition 5 (P_{ij})***

A parameter *j* of an operation *i*, noted $\boldsymbol{P_{ij}}$, is a tuple (*id*, *messageLabel*, *semConceptUri*, *xsdSchema*, *contextUriList*, *exitsLoweringSchema*, *existsLiftingSchema*) where :

✓ *id* is a unique key identifying the parameter $\boldsymbol{P_{ij}}$.

✓ *messageLabel* can be {"In", "Out"}, where "In" and "Out" respectively indicate that the parameter *j* is either an input or an output of the operation *i*..

- ✓ *semConceptUri* refers to the concept's URI in a semantic model that describes the parameter $\boldsymbol{P_{ij}}$.
- ✓ *xsdSchema* refers to XML data type of the parameter $\boldsymbol{P_{ij}}$
- ✓ *contextUriList* refers to a list of instances' URIs specifying the parameter's context $\boldsymbol{P_{ij}}$.
- ✓ *exitsLoweringSchema* is a Boolean attribute. If true, it means that there is a schema mapping transforming semantic data corresponding to *semConceptUri* into XML structure (*xsdSchema*).
- ✓ *existsLiftingSchema* is a Boolean attribute. If true, it means that there is a schema mapping transforming data from XML structure (*xsdSchema*) into the semantic data corresponding to *semConceptUri*

- ***Definition 6 (Op$_i$)***

An operation $\boldsymbol{Op_i}$ is a tuple (*key, params*) where :

- ✓ *key* is a unique key identifying the operation $\boldsymbol{Op_i}$.
- ✓ *params* identifies the set of $\boldsymbol{Op_i}$ parameters.

3. OFFLINE COMPOSABILITY RULES

Web services composability still a major issue [15]. To interconnect Web services' operations, a set of multi-aspect rules should be applied to check their composability. In this context, we introduce the notion of "Strict" and "Fexible" rule. "Strict" offline composability rule indicates that the interconnection of two operations is impossible if the rule cannot be satisfied. "Flexible" offline composability rule indicates that the interconnection of the two operations can be enabled by applying a manual, semi-automatic or automatic mediation even if the rule is not satisfied. This section presents the rules developed to ensure composability over functional, non-functional, contextual, data structure and technical aspects, and deal with syntactic, semantic and policies composition heterogeneities.

3.1. Functional Rule

We assume that an operation $\boldsymbol{Op_i}$ is functionally composable with an operation $\boldsymbol{Op_j}$ if it exists an $\boldsymbol{Op_i}$'s input $\boldsymbol{P_{im}}$ and an $\boldsymbol{Op_j}$'s output $\boldsymbol{P_{jk}}$ where the semantic concept describing $\boldsymbol{Op_i}$'s input is semantically equivalent to the semantic concept describing $\boldsymbol{Op_j}$'s output, or the semantic concept describing $\boldsymbol{Op_i}$'s input subsumes the semantic concept describing $\boldsymbol{Op_j}$'s output, or the semantic distance between the two concepts are lower than the threshold of the use's semantic relaxation [16, 17] tolerated.

$\boldsymbol{Op_i}$ is functionally composable with an operation $\boldsymbol{Op_j}$ where:

$$\begin{aligned}
&\exists\,(P_{im}, P_{jk})\ /\\
&(P_{im}\,.\,\text{messageLabel="In"} \wedge P_{jk}\,.\,\text{messageLabel="Out"})\\
&\wedge\,(\ (P_{im}\,.\,\text{semConceptUri} \equiv P_{jk}\,.\,\text{semConceptUri})\\
&\vee\,(P_{im}\,.\,\text{semConceptUri} \sqsupset P_{jk}\,.\,\text{semConceptUri})\\
&\vee\,(\text{diSem}\,(P_{im}\,.\,\text{semConceptUri}\ ,\ P_{jk}\,.\,\text{semConceptUri})\\
&\qquad\qquad <= \mathit{Seuil})\ )
\end{aligned}$$

Figure 2: Functional Offline Composability Rule

The functional rule is considered as a "Strict" rule since no connection between $\boldsymbol{Op_i}$'s inputs and $\boldsymbol{Op_j}$'s outputs is possible if this rule is not satisfied. The verification of this rule should precede any further checks. In case of this rule is not satisfied, the composability process has to be completed.

3.2. Contextual Rule

As previously explained, our approach proposes to inform contextual properties. These properties are described using SAWSDL ModelReference mechanism. The context of each operation parameter is specified by a list of semantic instances.

Thus, contextual composability checks for an operation $\boldsymbol{Op_i}$ functionally composable with an operation $\boldsymbol{Op_j}$ through respectively its input and its output, if the input $\boldsymbol{P_{im}}$ ' s context is compatible with the output $\boldsymbol{P_{jk}}$ ' s context. Otherwise, it should verify that each instance among the set of instances describing the context of the parameter $\boldsymbol{P_{im}}$, exists in the list of instances describing the context parameter $\boldsymbol{P_{jk}}$ as explained in Figure 3.

$$\forall\, U \in P_{ik}\,.\,\text{contextUriList},\ U \in P_{jm}\,.\,\text{contextUriList}$$

Figure 3: Contextual Offline Composability Rule

The contextual rule is considered as a "Flexible" rule.

3.3. Data Structure Rule

In general, in order to connect an operation's input $\boldsymbol{P_{im}}$ with an another operation's output $\boldsymbol{P_{jk}}$, the data schemes of the two parameters involved should be compatible.

$$\begin{array}{l}
(P_{ik}.\text{xsdSchema} \equiv P_{jm}.\text{xsdSchema}) \\
\vee\ (\ (P_{ik}.\text{semConceptUri} \equiv P_{jm}.\text{semConceptUri}) \\
\quad \wedge\ (P_{ik}.\text{existsLoweringSchema=true}) \\
\quad \wedge\ (P_{jm}.\text{existsLiftingSchema = true}) \\
)
\end{array}$$

Figure 4: Data Structure Offline Composability Rule

SAWSDL annotations used to semantically describe the inputs and outputs of Web services' operations, can specify mapping schemes (LoweringSchema and LiftingSchema) enabling the transformation of semantic data into XML structure and vice versa. In case of equivalence of two semantic concepts, their schemes (if they exist) can assume the role of mediator to transform XML data into semantic data (using $\boldsymbol{P}_{im}$ LoweringSchema) and transform semantic data into XML data (using LiftingSchema) even if $\boldsymbol{P}_{im}$. *xsdSchema* and $\boldsymbol{P}_{jk}$. *xsdSchema* are not equivalent, as shown in Figure 4.

The data structure rule is considered as a "Flexible" rule.

3.4. Non-Functional Rule

On the non-functional aspect, the composability of two operations $\boldsymbol{Op}_i$ and $\boldsymbol{Op}_j$ depends on the existence of two endpoints $\boldsymbol{E}_{ik}$ and $\boldsymbol{E}_{jk'}$ where all policies attached to these endpoints at *Endpoint*, *Binding*, *Operation* and *Service* Levels are coherent. In addition, the composability of the two operation in non-functional level depends also on the coherence of all message policies associated to $\boldsymbol{E}_{ik}$ and $\boldsymbol{E}_{jk'}$ and attached to the two parameters $\boldsymbol{P}_{im}$ and $\boldsymbol{P}_{jk}$, enabling the functional composability of the two operations $\boldsymbol{Op}_i$ and $\boldsymbol{Op}_j$.

$$\begin{array}{l}
\exists\,(E_{ik,}\ E_{jk'})\,/ \\
\quad (\forall\ Policy_{ikm},\ \exists\ Policy_{jk'm'}\ /(\ Policy_{ikm}.\text{policyLevel} = Policy_{jk'm'}.\text{policyLevel}), \\
\quad \exists\,(PolicyAl_{ikml,}\ \ PolicyAl_{jk'm'l'})\,/ \\
\quad (\forall\ \text{Asser}_{ikmlq}\ /\ (Asser_{ikmlq}.\text{optional = 'false'})\ ,\ \exists\ \ Asser_{jk'm'l'q'}\ \ / \\
\quad (Asser_{ikmlq}\ .\ assertSemConceptUri \equiv Asser_{jk'm'l'q'}\ .\ assertSemConceptUri)\ \boldsymbol{\vee} \\
\quad (Asser_{ikmlq}\ .\ assertSemConceptUri \sqsupset Asser_{jk'm'l'q'}\ .\ assertSemConceptUri)\ \ \boldsymbol{\vee} \\
\quad (\text{diSem}(Asser_{ikmlq}\ .assertSemConceptUri,\ Asser_{jk'm'l'q'}\ .assertSemConceptUri) \leq Seuil))) \\
\wedge \\
\quad (\forall\ \ MessagePolicy_{ixkw}\ \ ,\ \exists\ MessagePolicy_{jyk'w'}\ / \\
\quad \exists\,(MessagePolicyAl_{ixkwt}\ ,\ MessagePolicyAl_{jyk'w't'})\ \ , \\
\quad \forall\ Asser_{ixkwtr}\ \ /\ (Asser_{ixkwtr}\ .\ \text{optional ='false'})\ ,\ \exists\ \ Asser_{jyk'w't'r'}\ / \\
\quad (Asser_{ixkwtr}\ .\ \text{assertSemConceptUri} \equiv Asser_{jyk'w't'r'}\ .\ \text{assertSemConceptUri})\ \boldsymbol{\vee} \\
\quad (Asser_{ixkwtr}\ .\ \text{assertSemConceptUri} \sqsupset Asser_{jyk'w't'r'}\ \ .\ \text{assertSemConceptUri})\ \ \boldsymbol{\vee} \\
\quad (\text{diSem}(Asser_{ixkwtr}\ .\text{assertSemConceptUri},\ Asser_{jyk'w't'r'}\ .\text{assertSemConceptUri}) \leq Seuil)))
\end{array}$$

Figure 5:Non-Functional Offline Composability Rule

Practically, we consider that two policies $Policy_{ikm}$ and $Policy_{jk'm'}$ are coherent if for each assertion $Asser_{ikmlq}$ required by the policy alternative $PolicyAl_{ikml}$, exists an assertion

$Asser_{jk'm'tq'}$ corresponding to the policy alternative $PolicyAl_{ikmt}$ where the semantic concept describing $Asser_{ikmlq}$ is semantically equivalent to or subsumes the semantic concept describing $Asser_{jk'm'tq'}$, or the semantic distance between the two concepts are lower than the threshold of the use's semantic relaxation tolerated.

The non-functional rule is considered as a "Flexible" rule.

3.5. Technical Rule

The technical information concerning a Web service is generally embedded in the Binding element of the WSDL file. We consider that ***Op****i* is technically composable with ***Op****j* if it exits two endpoints ***E****ik* and ***E****jk'* where the transport protocol supported by the Binding associated to ***E****ik* is equivalent to the transport protocol supported by the Binding associated to ***E****jk'* .

The technical rule is considered as a "Flexible" rule.

$$\exists\ (E_{ik}, E_{jk'})\ \ 1 \le k \le |E_i|\ \text{et}\ 1 \le k' \le |E_j|\ /$$
$$E_{ik}.\text{transportProtocol} = E_{jk'}.\text{transportProtocol}$$

Figure 6: Technical Offline Composability Rule

4. OFFLINE COMPOSABILITY PHASES

The main approach adopted to check offline composability of Web services operations, includes three phases as shown in Figure 7.

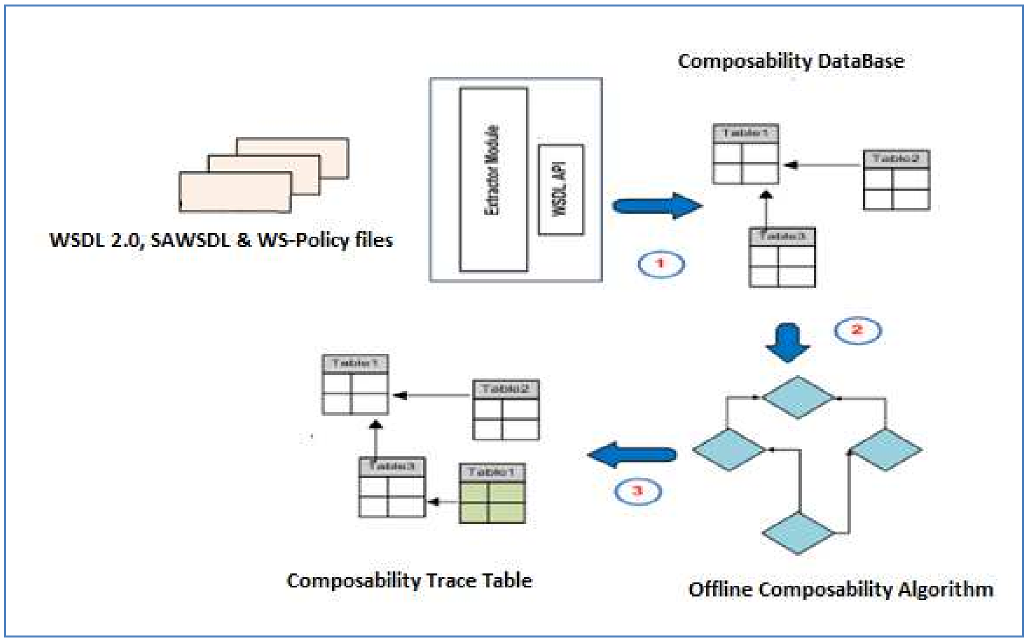
Composability DataBase
Extractor Module
WSDL API
WSDL 2.0, SAWSDL & WS-Policy files
Table1
Table2
Table3
1
2
3
Composability Trace Table
Offline Composability Algorithm

Figure 7: Offline Composability Process phases

a) **Extraction and Storage of Web service information**: this phase aims to extract, from SAWSDL, WSDL 2.0 and WS-Policy files, the web services properties needed to verify the offline composability of Web services, and store these properties in database tables to be computed using the composability algorithm developed. Actually, it should be noted that only the information involved in the treatment of offline composability are concerned by the extraction mechanism.

b) **Checking Offline Composability**: at this stage, the offline composability algorithm developed using java platform is performed. This algorithm incorporates, in fact, all functional, non-functional, contextual, data structure and technical composability rules.

c) **Storage of Offline Composability Results**: after execution of the offline composability algorithm, the functionally composable operations are stored in the table called *ComposabilityTrace*. For each tuplet (***Op****i* ,

Op_i, P_{im}, P_{jk}, E_{ik} , $E_{ik'}$) where Op_i is functionally composable with Op_j through respectively P_{im} and P_{jk}, the algorithm traces the composability result (true or false) for each non-functional, contextual, data structure and technical aspect related to each E_{ik} and $E_{ik'}$.

5. DISCUSSION AND OUTLOOK

In the present article, we propose a multi-aspect Web services composability model aligned with WSDL2.0, SAWSDL, and WS-Policy standards, which addresses a set of functional, non-functional, data structure and technical composability rules to check whether to or more web services operations can concretely interact with each other.

The notions of "Strict" and "Flexible" rules have been introduced to differentiate between a rule whose dissatisfaction prevents any possibility to interconnect two operations (case of functional rule), and a rule whose dissatisfaction does not preclude the possibility of interconnecting the two operations if a manual, semi-automatic or automatic mediation solution can be exploited (case of non-functional , contextual, data structure and technical rules). It is for this reason that the verification of the functional verification rule should precede all other rules.

The work of Medjahed et al. [18] can be considered as a close related work. Authors present an ontology-based framework to describe Web services and propose a set of composability rules to determine whether two services are composable. However, we note that the proposed work is founded on the use of a specific ontology that has been defined by authors. Besides, it proposes to check parameters composability, considering only their data types, units and business roles without covering other essential information such as the applied security protocols (signed or encrypted parameter) or other technical constraints. This information is encapsulated in semantic web services policies in our model.

Regarding the operations composability, authors affirm that the operations' categories (domains) should be semantically compatible. In our proposal, we believe that two operations with compatible parameters may be connected even if their categories are semantically disjoint. They also argue that two operations A and B are composable, if for each input of the operation A there is a composable output in the operation B. We think that operations A and B can be connected if just one operation A's input is composable with one operation B's output.

Also, non-functional properties, often considered in the selection of the best composition plan [19, 20], are checked in the composability phase to intercept any non-functional heterogeneity.

In conclusion, the key characteristics of our model are, on one hand, the existing alignment between our service composability model and WSDL standard to enable composition of real public services. On the other hand, the proposed rules deal with functional, non-functional and technical composability constraints in syntactic and semantic levels to identify possible heterogeneities when interconnecting Web services.

As a perspective of the current work, we propose to address the behavioral aspect based on the OASIS standard BPEL4WS [21] (service orchestration) and the W3C standard WS-CDL [22] (choreography of services). In other words, we intend to check the behavioral composability of two composite services using behavioral description specified by the two standards.

REFRENCES:

[1] Beauche, S. et Poizat, P. , "Automated service composition with adaptive planning". ICSOC, volume 5364 of Lecture Notes in Computer Science, 2008, pp. 530–537.

[2] Fujii, K., et Suda., T., "Semantics-based context-aware dynamic service composition". TAAS, 4(2), 2009.

[3] Zeng, L., Ngu, A. H. H. , Benatallah, B., Podorozhny, R. M., et Lei., H., "Dynamic composition and optimization of web services". Distributed and Parallel Databases, 24(1-3), 2008, pp: 45–72

[4] Ying, L., "A Method of Automatic Web Services Composition Based on Directed Graph", vol. 1, 2010 International Conference on Communications and Mobile Computing, 2010, pp: 527-531

[5] The OWL Services Coalition. OWL-S 1.1 Release. Available at http://www.daml.org/services/owl-s/1.1/, November 2004.

[6] Roman D., Scicluna J., Fensel D., Polleres A., and Bruijn J,. "Ontology-based Choreography of WSMO Services". Wsmo d14 final draft v0.3, DERI, 2006. Available at: http://www.wsmo.org/TR/d14/v0.3/.

[7] Christensen E., Curbera F., Meredith G., and Weerawarana S., "Web Services Description Language (WSDL) 1.1". http://www.w3.org/TR/wsdl, March 2001.

[8] Farrell, J. and H. Lausen (2007). "Semantic Annotations for WSDL and XML Schema", W3C recommendation, http://www.w3.org/TR/sawsdl/.

[9] Vedamuthu, A., D. Orchard, F. Hirsch, M. Hondo, P. Yendluri, T. Boubez and Ü. Yalçinalp (2007), "Web Services Policy 1.5 – Framework", W3C recommendation, http://www.w3.org/TR/ws-policy/

[10] Omrana, H., El Bitar, I., Belouadha, F.Z. and Roudies, O., "A Comparative Evaluation of Web Services Description Approaches". ITNG '13 Proceedings of the IEEE 10th International Conference on Information Technology: New Generations, Las Vegas, Nevada, USA, April 2013, pp: 60-64

[11]Akkiraju, R., Sapkota, B., "Semantic Annotations for WSDL and XML Schema — Usage Guide", 2007. Available at: http://www.w3.org/2002/ws/sawsdl/spec/examples/

[12] O'Sullivan, J., Edmond., D. et Hofstede., "A., Formal description of non functional service properties". Technical Report FIT-TR-2005-01, Centre for Information Technology Innovation, Queensland University of Technology, 2005

[13] Omrana, H., Belouadha, F-Z. and Roudiès, O. (2011) 'UML-based profiles for policy-aware web services', Int. J. Int. J. Reasoning based Intelligent Systems, Vol. 3, Nos. 3/4, pp.217–225.

[14] Omrana, H., Belouadha, F. Z., & Roudies, O. "A Composability Model for Efficient Web Service's Connectivity". In Intelligent Networking and Collaborative Systems (INCoS), 2012 4th International Conference on (pp. 483-484). IEEE.

[15] Medjahed, B., Benatallah, B., Bouguettaya, A., et Elmagarmid., A. K., Webbis : "An infrastructure for agile integration of web services". Int. J. Cooperative Inf. Syst., 13(2), 2004, pages:121–158

[16] Hatzi, O., Vrakas, D., Nikolaidou, M., Bassiliades, N., "An Integrated Approach to Automated Semantic Web Service Composition through Planning". IEEE TRANSACTIONS ON SERVICES COMPUTING, 2011

[17] Omrana, H., Belouadha, F.Z. et Roudiès, O., "Template–based matching algorithm for dynamic web services discovery", International Journal of Information and Communication Technology, Volume 4, Issue 2/3/4, Inderscience Publishers, 2012, pp: 198-209

[18] Medjahed, B., Bouguettaya, A., "A multilevel composability model for semantic Web services", Knowledge and Data Engineering, IEEE Transactions on , vol.17, no.7, pp: 954-968, July 2005

[19] Jiang, W., Zhang, C., Huang, Z., Chen, M., Hu, S., Liu, Zh., Qsynth: "A Tool for QoS-Aware Automatic Service Composition". In Int. Conf. on Web Services, 2010, pp: 42–49

[20] Li, J., Chen, S., Li, Y., Zhang, Q., "Semantic Web Service Automatic Composition Based on Service Parameter Relationship Graph", 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, 2011, pp:1773-1778

[21] Alves et al., "Web Services Business Process Execution Language Version 2.0". OASIS Standard, April 2007. Available at http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[22] Barros A., Dumas M., Oaks P., " A Critical Overview of the Web Services Choreography Description Language (WS-CDL) ", Proc. of the Business Process Trends (BPTrends), 2005