Context driven mediation service in Data-as-a-Service composition

Idir Amine Amarouche[1] and Djamal Benslimane[2]

[1] Université des Sciences et de la Technologie Houari Boumediene
BP 32 El Alia 16111 Bab Ezzouar, Algeirs, Algeria
[2] Université Lyon 1, LIRIS UMR5205
43, bd du 11 novembre 1918, Villeurbanne, F-69622, France
i.a.amarouche@gmail.com, Djamal.Benslimane@liris.cnrs.fr

Abstract. Data as a Service (DaaS) builds on service-oriented technologies to enable fast access to data resources on the Web. Many approaches are proposed to achieve the DaaS composition task which is reduced to query rewriting problem. In this context, DaaS is described as Parametrized-RDF View (PRV) over Domain Ontology (DO). However, the DO is unable to capture the different perspectives or viewpoints for the same domain knowledge. This limitation raises semantic conflicts between pieces of data exchanged during the DaaS composition process. To face this issue, we present a context-driven approach that aims at supporting semantic mediation between composed DaaSs. The semantic reconciliation based on mediation service is performed through the execution of rule mapping which achieves the transformation between contexts.

Keywords: DaaS composition, mediation service, context, semantic conflict.

1 Introduction

Nowadays, modern enterprises are using Web services for data sharing within and across the enterprise's boundaries. This type of Web service is known as Data-as-a-Services (DaaSs) which return collections of Data for a given set of parameters without any side effects. DaaSs composition is a powerful means to answer users' complex queries. Semantic-based approaches are proposed to enable automatic composition by describing the Web services properties over ontology. In fact, many ontology languages (e.g.,OWL-S[3], WSMO [4]) and extension mechanisms (e.g., WSDL-S [5]) provide standard means by which WSDL[6] document can be related to semantic description. However, this means do not provide a way to relate semantically the Web service parameters (i.e., input and

[3] http://www.w3.org/Submission/OWL-S/
[4] http://www.wsmo.org/TR/d2/v2.0
[5] http://www.w3.org/Submission/2005/SUBM-WSDL-S-20051107/
[6] Web Service Description Language

output) which hampers their applicability to DaaS composition. The automation of DaaS composition requires the specification of the semantic relationships between inputs and outputs parameters in a declarative way. This requirement can be achieved by describing DaaS as views over a *DO* following the mediator-based approach [8]. Thereby, the DaaS composition problem is reduced to a query rewriting problem in the data integration field. In this context, several works [2, 9, 7] consider DaaS as Parametrized RDF[7] Views (*PRVs*) with binding patterns over a *DO*, to describe how the input parameters of the DaaS relate to the data it provides. Defined views are then used to annotate DaaSs description files (e.g., WSDL files) and are exploited to automatically compose DaaSs. However, there are several reference ontologies which formalize the same domain knowledge. Thus, the construction of a *DO* unifying all existing representations of real-world entities in the domain is a strong limitation to interoperability between DaaS, this essentially raises semantic conflicts between pieces of data exchanged during DaaS composition. To this end, the applicability of previously cited DaaS composition approaches is not practical. Therefore, considering the semantic conflict detection and resolution during the composition process is crucial as service providers' contexts are practically different. In this regard, the approaches discussed in [4] and [5], have used the context representation for semantic mediation in Web service composition. In fact, they propose an extension of *DO* by a lightweight ontology which needs a small set of generic concepts to capture the context. However, these representations assure only simple mapping between semantically equivalent context parameter (price, unit,etc.). Further, the technical transformation code assuring the conversion from one context to another makes harder the maintainability of the semantic mediation between service composition components.

Motivating example: Let us consider an e-health system where the information needs of health actors are satisfied with DaaS Composition System (DCS), as proposed by [2, 9], which exports a set of DaaSs to query patient data. We assume that a physician submits the following query Q_1: "What are the states indicated by the recent Blood Pressure Readings (BPR) for a given patient". We assume that the DCS will automatically generates DaaS composition, as response to physician query, including respectively S_1, S_2 and S_3 as depicted in figure 1.(a). The DCS invokes automatically in the following order: 1) "S_1" that provides the recent Vital Sign Exam (BPR,etc.) performed on his patient; 2)"S_2" to retrieve the BPR measure[8]; 3)"S_3" to retrieve the "BPR" state from the BPR value returned by S_2. However, the DCS exports DaaSs expressed over *DO* does not take into account the context. By the context we mean the knowledge allowing to compare DaaS parameters values when there is a conflict (i.e, measurement unit, codification system, classification system, BPR value structure,etc.). Indeed, the physician has to manually detect the existing conflict in generated DaaS composition. For that, he has to select and to invoke

[7] RDF: Resource Description Framework

[8] BPR is represented by two concatenated values. eg., 120/80 where 120 is BPR Diastolic (BPR.D) value and 80 BPR Systolic (BPR.S) value

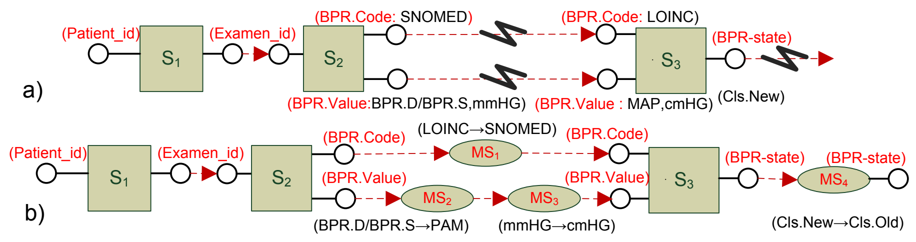


Fig. 1. Physician query scenario: a) DaaS composition generated by the DCS; b) The DaaS composition with the appropriate mediation services.

the appropriate mediation services, in the right order, to make the generated composition executable as depicted in figure 1.(b). The physician has to invoke: 1)"MS_1": to mappe the BPR code returned by S_2(LOINC[9]) to code acceptable by S_3 (SNOMED[10]); 2) The composition of "MS_2" and "MS_3" where :"MS_2" aggregates the two values expressing BPR measure returned by "S_2" to MAP[11] value acceptable by S_3 and "MS_3" converts the MAP value expressed with the measurement unit ("mm/Hg") returned by MS_2 to the MAP value expressed with the measurement unit acceptable by S_3 ("cm/Hg"); "MS_4" : to mappe the BPR state returned by "S_3" represented according to the new classification BPR value table (e.g., stage 1,2,3,4) to the state acceptable by the physician represented according to the old classification (e.g., severe, moderate, mild). This is a rather demanding task for non expert users (e.g.physicians). Thus, automating conflict detection and resolution in DaaS composition is challenging.

Contributions: In this paper we propose a context driven approach for automatically inserting appropriate mediation services in DaaS compositions to carry out data conversion between interconnected DaaS. Specifically, we propose 1) a context model expressed over Conflicting Aspect Ontology(CAO) which is an extension of "DO"; 2) an extension of PRV based DaaS model based on context to express more accurately the DaaS parameters semantic; 3) a mediation service model behaving as a mapping rule to perform the transformation of DaaS parameters from one context to another.

Outline: The rest of this paper is organized as follows. Section 2, presents the overview of our approach. In Section 3, we leverage different proposed models. In Section 4, we present a global view on our conflict detection and resolution algorithm and our implementation. Finally, section 5 provides a conclusion and future works.

[9] LOINC : Logical Observation Identifiers Names and Codes
[10] SNOMED: Systematized Nomenclature of Medicine, Clinical Terms
[11] Mean Arterial Pressure is BPR value, $MAP = \frac{2}{3}(BPR.D) + \frac{1}{3}(BPR.S)$

2 Approach overview

Figure 2 gives an overview of our approach. Our proposal aims to provide a framework for automatic conflict detection and resolution in DaaS composition. Our approach takes into account the context of the service components in DaaS composition and the context of the query. DaaS services are modeled as PRV over a DO and contextualized over Conflicting Aspect Ontology (CAO). The mediation services are modeled as mapping rule over CAO specifying the DaaS parameters transformation from one context to another. The contextualized PRV and the mapping rule are incorporated within correspondent WSDL description files as annotation. The DaaS service registry includes business services while the mediation service are organized in other registry to keep the mediation concerns orthogonal from functionalities of DaaS.

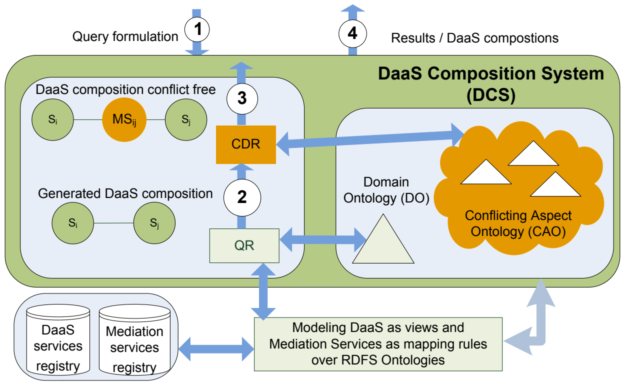

Query formulation
1
4
Results / DaaS compostions
DaaS Composition System (DCS)
DaaS composition conflict free
3
S_i
MS_{ij}
S_j
CDR
Generated DaaS composition
2
S_i
S_j
QR
Domain Ontology (DO)
Conflicting Aspect Ontology (CAO)
DaaS services registry
Mediation services registry
Modeling DaaS as views and Mediation Services as mapping rules over RDFS Ontologies

Fig. 2. Approach overview

The DaaS composition process starts when the user specifies a query over DO and CAO using SPARQL [12] query language (see circle 1 in figure 2). The DCS uses the query rewriting algorithm proposed by [2] and existing PRV to select the DaaS that can be combined, to answer the query (see circle 2 in figure 2). After that, our Conflict Detection and Resolution Algorithm (CDR) takes place for conflict verification in each generated DaaS composition. Then, in case where a conflict is detected between output/input operation (i.e., subsequent services in DaaS compositions, query and DaaS compositions) our algorithm insert automatically calls to appropriate mediation services to resolve semantic conflict (see circle 3 in figure 2). Then, the DCS translates a composite DaaSs conflict free into query execution plan describing data and control flow. The plan will be executed and returns data to the user (see circle 4 in figure2). In this paper, we will focus only on Conflict Detection and Resolution process.

[12] We adopt SPARQL: http://www.w3.org/TR/rdf-sparql-query/, the de facto query language for the Semantic Web, for posing queries.

3 Modeling issues

We leverage in this section different models used through the paper. The definition of the basic concepts such as the Domain Ontology(DO), the Parametrized RDF view (PRV) and the Conjunctive Query (CQ) are presented formally in [1]. Due to space limitations, we will not present their corresponding figures. In the sens of the present work, the DaaS Composition $cs = \{s_i..s_n\}$ represents the set of ordered services into DaaS composition ; $First(cs)$ (e.g, s_i) and $Last(cs)$ (e.g,s_n) denote the first and the last DaaS in "cs". We mean by the "CSs" the set of compositions, generated by the query rewriting algorithm of "DCS", requiring testing and resolution of conflicts.

3.1 Conflicting Aspect Ontology:

Conflicting Aspect Ontology (CAO) is a family of a lightweight ontology, specified in RDFS. CAO extends the DO entities with a taxonomic structure expressing different DaaS parameters semantic conflict[13]. The CAO is a 3 tuple $< AC_g, AC_i, \tau >$, where: 1) "AC_g" is a set of classes which represents the different conflicting aspects of a DO entities. Each ac_g class in AC_g has one super-class and a set of sub-classes. Each ac_g class has a name representing a conflicting aspect, such as, `CAO:Measurement-Unit` as depicted in Figure 3; 2) "AC_i" is a distinct set of instanceable classes having one super-class in AC_g. By definition, ac_i is not allowed to have sub-classes. For instance "mm/HG" and "cm/HG" are two instanceable classes from the `CAO:BPR-Unit` class; 3)"τ" refers to the sibling relationships on AC_i and AC_g. The relationships among elements of AC_g is disjoint. However, elements of AC_i of a given ac_g can be related by the *Peer relationship* which indicates similar data semantics. *Part-Of relationship* which relates ac_i entity and its components (e.g., BPR.D and BPR.S values are Part-Of BPR).

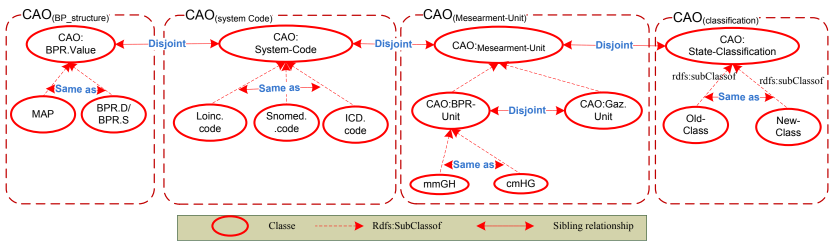


Fig. 3. Conflicting Aspect Ontology

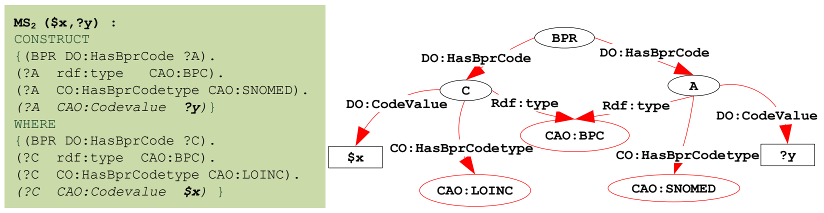
[13] For the classification of the various incompatibility problems in web service composition see [6]

3.2 Context model:

The context has the form: $C = \{(D_i, V_i) | i \in [1, n]\}$ where D_i, represents an ac_g class whose values are from a value-set (Vi) where $V_i \in AC_i$. For instance, the context $C_{MU} = \{BPR - Unit : mm/HG\}$ indicates that the BPR measurement unit is "mm/HG". The proposed context model is used to express more precisely the query formulated by the user, the DaaS published by the provider and the semantic conflict occurring in each O/I[14] operation in given $cs_K \in CS$. **1)"Contextualized Conjunctive Query model"** is $CCQ(X) : - < CQ(X) | CCQ_{(X,CO)} >$ where $CQ(X)$ is the conjunctive query expressed over DO, and $CCQ_{(X,CO)}$ is the context of the distinguished variable X and the query constraint CO expressed over CAO; **2)"Contextualized DaaS model":** The C-DaaS is $S_j(\$X_j, ?Y_j) : - < V_{DO} > | < Ext_{CAO} >$ where V_{DO} is the PRV of S_j and Ext_{CAO} is a tuple $< C_{X_j}, C_{Y_j} >$ where C_{X_j} and C_{Y_j} are respectively the input and the output DaaS parameter contexts. C_{X_j} and C_{Y_j} are described by a set of RDF triples over CAO in form of 2-tuple $< AC_g, AC_i >$; **3)"Context and semantic conflict":** In the sense of the present work, semantic conflict occurs in O_n/I_m operation having respectively O_n and I_m as an output and an input parameter which refer to the same DO entity. However, their contexts represented respectively by C_{O_n} and C_{I_m} refer to different "ac_i" entities from the same "ac_g" . Then we say that a parameter semantic conflict "ac_g" exists in O_n/I_m.

3.3 Mediation service model

Mediation Services MS assures the semantic reconciliation in the case where the O/I operation causes a conflict. The MS model consists of mapping rule having the form $MS(\$O_J, ?I_J) : G_O \rightarrow G_I$, where $\$O_J$ and $?I_J$ are the sets of input and output variables of MS_j respectively. G_O and G_I represent the set of RDF triples representing contextualized DaaS /query parameters. We deem appropriate to use the SPARQL's construct statement (i.e., CONSTRUCT G_I WHERE G_O) as a rules language to define rule mapping as proposed by [3]. For



```
MS2 ($x,?y) :
CONSTRUCT
{(BPR DO:HasBprCode ?A).
(?A  rdf:type   CAO:BPC).
(?A  CO:HasBprCodetype CAO:SNOMED).
(?A  CAO:Codevalue  ?y)}
WHERE
{(BPR DO:HasBprCode ?C).
(?C  rdf:type  CAO:BPC).
(?C  CO:HasBprCodetype CAO:LOINC).
(?C  CAO:Codevalue  $x) }
```

Fig. 4. Mediation service model

[14] i.e, two subsequent DaaSs "S_n" and "S_m" in "cs", First(cs) and $CCQ_{(CO)}$, Last(cs) and $CCQ_{(X)}$.

each conflicting aspect AC_g we define a mapping rule template. For instance, the mediation service MS_2 assuring the same-as mapping one-to-one of BP code value from "LOINC" code to "SNOMED" code is presented in figure 4. In the same manner, we define the mapping many-to-one, one-to-many and many to many. To the best of our knowledge, this work is the first to use SPARQL construct statement to model mediation services.

4 Algorithm and implementation

In the following, we present the details of our Conflict Detection and Resolution Algorithm (CDR) depicted in figure 5. The inputs to the CDR is a set of

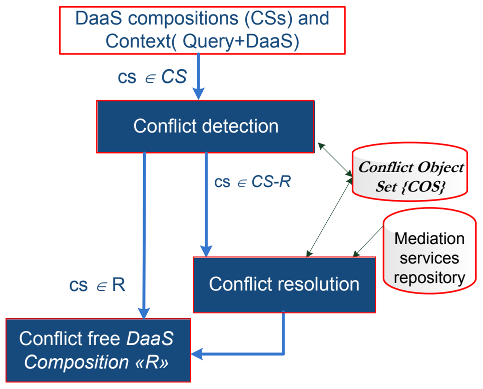

DaaS compositions (CSs) and Context(Query+DaaS)
cs ∊ CS
Conflict detection
cs ∊ CS-R
Conflict Object Set {COS}
Mediation services repository
cs ∊ R
Conflict resolution
Conflict free DaaS Composition «R»

Fig. 5. CDRM architecture

"CSs" generated by the QR algorithm as explained in section 2. The outputs of CDR are "CSs" conflict free. The desired mediation service is found and called automatically using the CDR algorithm which is two phases : Detection and Resolution. In the first phase each composition "cs" is examined to detect potential conflicts. Thus, if "cs" is without conflicts then it is inserted into the set of compositions without conflicts R; else the conflicts of "cs" are added into the conflict object set "COS". Finally, the set of composition without conflict R is removed from CS. Thus CS consists of the composition with conflicts. In the second phase, each detected conflict is resolved by performing the matching between the required context transformation to the mapping rules defining the mediation services. The matching is obtained, the automatic calls to the correspondent mediation services are inserted in "cs" to resolve conflicts. Then, the new set of composition CS (i.e, composition without conflict) are added into R and returned to DCS for query plan execution. In order to test test our proposal, we have implemented a Java Based application and test it with multiple examples, including the motivating example [15]. Each Web services is deployed

[15] The implementation test are available in http://sites.google.com/site/ehrdaas/home

on top of a GlassFish web server. Each DaaS is annotated by the contextualize PRV and each Mediation service is annotated by SPARQL construct statement. In the evaluation phase we have considered a set of queries through which we identify the following : 1) During the detection phase, we can detect the set of conflict aspect identified in "AC_g". 2) During the resolution phase, according to the number of conflict detected in each O/I operation: when there is a conflict including one aspect ac_g (e.g., BPR-code) or a conflict including several aspects ac_g (e.g., BPR-value), our solution provides automatically the appropriate mediation service. When we have a several mediation services allowing to resolve the same conflict, our algorithm returns randomly one of them as long as they achieve the same functionality.

5 Conclusion and future work

In this paper, we propose an extension of PRV based DaaS model based on context. The proposed context model expressed over Conflicting Aspect Ontology aims to handle semantic conflict in DaaS composition. Our model allows to specify the mediation service as mapping rule performing the simple or complex transformation of DaaS parameters from one context to another. Our future perspective will to deal with the performance issues of our algorithm and how to resolve a given conflict for which there is no appropriate mediation service.

References

1. Amarouche, I.A, Benslimane, D., Barhmagi, M., Mrissa, M., Alimazighi, Z. : Electronic Health Record DaaS services Composition based on Query Rewriting. Transactions on Large-Scale Data and Knowledge-Centered Systems. 4, 95–123 (2011)
2. Barhamgi, M., Benslimane, D., Medjahed, B. : A Query Rewriting Approach for Web Service Composition. IEEE Transactions Services Computing. 3, 206–222 (2010)
3. Euzenat, J., Polleres, A., Scharffe, F. : Processing Ontology Alignments with SPARQL. International Conference on Complex, Intelligent and Software Intensive Systems. 913–917 (2008)
4. Li, X., Madnick, S., Zhu, H., Fan, Y. : Reconciling Semantic Heterogeneity in Web Services Composition. ICIS 2009 Proceedings. 20 (2009)
5. Mrissa, M., Ghedira, C., Benslimane, D., Maamar, Z. : A Context Model for Semantic Mediation in Web Services Composition. ER. 12–25 (2006)
6. Nagarajan, M., Verma, K., Sheth, A.P., Miller,J.A. : Ontology Driven Data Mediation in Web Services. Int. J. Web Service Res. 104–126 (2007)
7. Vaculín, R., Chen, H., Neruda, R., Sycara, K. : Modeling and Discovery of Data Providing Services. ICWS. 54–61 (2008)
8. Wiederhold, G. : Mediators in the Architecture of Future Information Systems. Computer. 25, 38–49 (1992)
9. Zhou, L., Chen, H., Wang, H., Zhang,Y. : Semantic Web-Based Data Service Discovery and Composition. SKG. 213–219 (2008)