Quality-Based Data Integration On Multi-Cloud Environments

Daniel A. S. Carvalho
(Supervised by Chirine Ghedira-Guegan)
Université Jean Moulin Lyon 3
Centre de Recherche Magellan - IAE
Lyon, France
daniel.carvalho@univ-lyon3.fr

ABSTRACT

In the traditional databases theory, data integration is a problem of merging data from data sources in order to provide a unified view of this data to the user. This PhD project address data integration in multi-cloud environments. Current data integration systems implies consuming data from data services deployed in cloud contexts and integrating the results. The project takes a new angle of the problem by proposing an SLA-based approach, which given a user query and her integration preferences, to match the user preferences (such as whether she accepts to pay for the data, its provenance, freshness and how much she is ready to pay for the integration, among others) with the services' SLA while delivering the results. The objective is to enhance the quality on data integration taking into consideration the economic model imposed by the cloud. Our preliminary experiments have shown that quality can be enhanced and the cost of the integration results can be minimized by using our approach.

1. INTRODUCTION

In recent years, the cloud have been the most popular deployment environment for data integration [5]. Data integration has evolved with the emergence of data services that deliver data under different quality conditions related to data freshness, cost, reliability, availability, among others. Data are produced continuously and on demand in huge quantities and sometimes with few associated meta-data, which makes the integration process more challenging. Some approaches express data integration as a service composition problem in which given a query the objective is to lookup and compose data services that can contribute to produce a result. Finding the best service composition that can answer a query can be computationally costly. Furthermore, executing the composition can lead to retrieve and process data collections that can require important memory, storage and computing resources.

Data integration has been addressed in the service-oriented architectures [6, 9, 14, 15]. [6] proposed a rewriting method based on SPARQL for RDF data integration. The work focused on avoiding ineffective data integration by solving the entity co-reference problem. [9] introduced SODIM, a system which combines data integration, service-oriented architecture and MapReduce distributed processing. [15] presented a solution focusing on data privacy in order to integrate data. [14] developed an inter-cloud data integration system that considers a trade-off between users' privacy requirements and the cost for protecting and processing data. According to the users' requirements, the query is created and executed meeting privacy and cost constraints. The novelty of these approaches is that they perform data integration in service oriented contexts, particularly considering data services. They also take into consideration the requirement of computing resources for integrating data. Thus, they exploit parallel settings for implementation costly data integration processes. However, they are manly focused on performance and privacy aspects putting aside other users' integration requirements such as data provenance, data integrity, confidentiality, reliability, availability, whether she wants to use free services, among others. Moreover, even with the cloud on-demand resources provisioning (which implies an associated cost), the user is limited to her cloud subscription and maximum budget she is ready to pay for her desired integration. The economic cost should be taken into consideration.

As highlighted in our previous work [5], we strongly believe that service level agreements (SLA) can be used in order to cover the limitations and enhance the quality in the current data integration solutions. Research contributions SLA in cloud computing mainly concern (i) the negotiation phase (step in which the contracts are established between customers and providers) and (ii) monitoring and allocation of cloud resources to detect and avoid SLA violations. Yet, to the best of our knowledge, we have not find works proposing SLA-based approach for data integration in a multi-cloud environment. Simiraly to our idea, [12] proposed a SLA-based data integration model for grid environments. The approach uses SLAs to define database resources and evaluated them in terms of processing cost, amount of data and price of using the grid. In addition, a matching algorithm is proposed to produce query plans us-

ing the selected resources. The most appropriated solutions based on these QoS are selected as final results. Our work differs from [12] in some aspects:

- Data is delivered as *data services* in a multi-cloud context. *Data services* and *cloud providers* export their SLA defining the quality conditions under which the service is delivered.
- SLAs are not limited only to describe the cost and amount of data, but also data quality aspects such its provenance, privacy, confidentiality, freshness, and service's delivery aspects such as response time, availability, reliability, among others.
- Users are able to express queries associating quality integration requirements to them. Then, the service selection and rewriting process in terms of service compositions are guided by the user's requirements and the SLAs exported by *data services* and *cloud providers*.

In this context, current SLA models are not sufficient to cover the data integration requirements and multi-cloud context. Thus, we are current working on new models to tackle these aspects. In summary, the objectives of this PhD project contribute as following: (1) we design a new SLA model for data integration; (2) propose data integration approach adapted to the vision of the economic model of the cloud. The originality of our approach consists in guiding the entire data integration solution taking into account (i) user preferences statements; (ii) SLA contracts exported by different cloud providers; and (iii) several QoS measures associated to data collections properties (for instance, trust, privacy, economic cost); and (3) validation of our approach in a multi-cloud scenario.

2. APPROACH

This section introduces our new vision of data integration, proposes our approach and briefly presents our query rewriting algorithm guided by user preferences and SLAs.

2.1 New vision of Data Integration

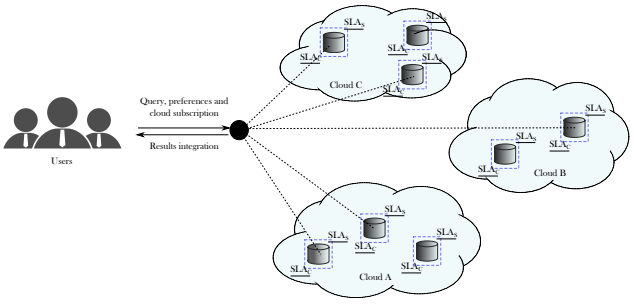

Figure 1: Data integration scenario

In our work, we consider a new vision of data integration in a multi-cloud context (see figure 1). Data is accessed and retrieved as *data services* deployed in *cloud providers* geographically distributed. *Data services* and *cloud providers* export their SLA specifying the level of services they can guarantee to users. Two types of SLA are considered: (1) cloud SLA (SLA_C), agreements between a *data service* and a *cloud provider*; and (2) service SLA (SLA_S), agreements between *users* and *data services*. The figure 2 illustrates a cloud SLA schema.

Therefore, given a user query, her integration quality requirements and her cloud subscription, it is rewritten in terms of cloud services (*data services* and *data processing services*) composition that fulfill the integration requirements and deliver the expected results to the user. This new vision brings challenges to data integration, such as:

- **Performance**. In the multi-cloud, we are dealing with a huge amount of *data services*, *data processing services* and *cloud providers*. Consequently, producing and executing service compositions can require an important quantity of resources and processing time.
- **Economic model**. Even with the possibility of having an unlimited access to resources, the user is limited to the resources she has contracted and to the budget she is ready to pay for. Thus, it is crucial to design new methods in order to produce rewritings which satisfies the user integration requirements and the cloud economic model.
- **Quality issues**. Some rewritings produced and executed to the user query could not satisfy her quality requirements concerning privacy, data provenance, cost, among others. Producing and executing these rewritings implies increasing time processing and integration total cost.
- **SLA heterogeneity**. While producing the rewritings, it is necessary to match the user requirements with the different SLAs exported by the *cloud providers* and *cloud services*. In the multi-cloud, *cloud providers* and *cloud services* export SLAs with different semantics and structure that makes the matching SLA and user requirement challenging. In addition, we are also dealing with incompatibilities of SLAs.
- **Reuse**. Rewriting and executing the user query is computationally costly in terms of processing time and economic cost. Thus, it is necessary to propose a manner of reusing previous integration in order to save time and money, but also meeting the user expectations.

2.2 SLA-based data integration approach

Motivated by the challenges discussed in the previous section, we propose our SLA-based data integration approach to multi-cloud environments. The data integration process includes (i) looking up services that can be used as *data services*, and for services required to process the retrieved data and build an integrated result (called *data processing services*); (ii) processing data retrieval, processing and integration; and (iii) delivering results to the user considering her quality requirements, context and resource consumption. In this sense, our approach is divided in four steps. Given a user *query*, a set of user *preferences* associated to it, *cloud providers* and *cloud services*:

SLA derivation. In this step, we compute what we call an *integrated SLA* that matches (including quality constraints and data requirements) with the SLA's provided by *cloud services*, given a specific user cloud subscription. The user may have general *preferences* depending on the context she

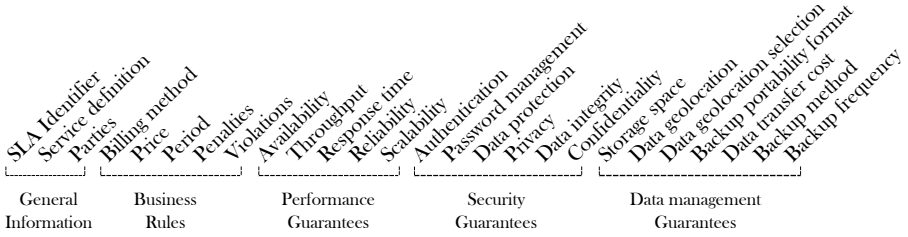

Figure 2: Cloud SLA

wants to integrate her data such as economic cost, bandwidth limit, free services, and storage and processing limits. The *SLA derivation* is the big challenge while dealing with SLAs and particularly for adding quality dimensions to data integration. Furthermore, the *integrated SLA* guides the query evaluation, and the way results are computed and delivered.

Filtering data services. The *integrated SLA* is used (i) to filter previous SLA derived for a similar request in order to reuse previous results; or (ii) to filter possible *cloud services* that can be used for answering the query. The SLA exported by a selected *cloud service* should satisfy the user *preferences*.

Query rewriting. Given a set of *data services* that can potentially provide data for integrating the query result, a set of service compositions is generated according to the *integrated SLA* and the agreed SLA of each *data services*.

Integrating a query result. The service compositions are executed in one or several clouds where the user has a subscription. The execution cost of service compositions must fulfill the *integrated SLA*. The clouds resources needed to execute the composition and how to use them is decided taking in consideration the economic cost determined by the data to be transferred, the number of external calls to services, data storage and delivery cost.

The algorithm 1 describes in general lines our approach which integrate data on multi-cloud environments considering quality aspects. Giver a user query Q, its associated user preferences P, a user integration context (its cloud subscription (C)), a set of previous *integrated SLA* $\mathcal{I}_{\mathcal{SLA}}$, a set of *data services* (DS), a set of *data processing services* (DPS) and a set of *cloud providers* CP, the integration results (IR) is delivered to the user fulfilling her preferences and considering her integration context. The approach begins looking for previous *integrated SLAs* that can be matched with the user query and user' preferences (lines 2 - 6). The set pI includes all matched i (line 4). If previous *integrated SLA* are identified (line 7), the one that perfectly matches with the user preferences is selected (line 8). Then, a new *integrated SLA* is computed to user query based on the previous *integrated SLA*, her preferences and her context (line 9). At this step, the services composition produced to a previous query is reused from its *integrated SLA* (sI). Otherwise, a new *integrated SLA* is created to user request given the query, the preferences and the context (line 11). Then, it is updated by calling the *Rhone* procedure (line 12). The *Rhone* is responsible (i) to select *data services* (in DS) and *data processing services* (in DPS) which satisfy the user preferences based on their SLAs; and (ii) to produce services compositions that completely match with the query and fulfill the user preferences. Finally, the integration results (IR) are produced and delivered to the user considering her context executing the service compositions generated to the user query (included in the *integrated SLA*). Note that while executing compositions, it is necessary to satisfy the user context requirements and also to dynamically update the integrated SLA adding the information of contracts that were established to allow the integration. Finally, the integrated SLA is stored to be used in a next query.

Algorithm 1 - SLA-based data integration

Input: Query (Q), user preferences (P), user cloud subscription (C) and a set of previous *integrated SLA* ($\mathcal{I}_{\mathcal{SLA}}$).
Output: Integration results delivered considering the user context (cloud subscription).

1: $pI \leftarrow \emptyset$
2: **for all** SLA_i in $\mathcal{I}_{\mathcal{SLA}}$ **do**
3: **if** $macthes(Q, P, SLA_i)$ **then**
4: $pI \leftarrow pI \cup \{SLA_i\}$
5: **end if**
6: **end for**
7: **if** $pI \neq \emptyset$ **then**
8: $sI \leftarrow chooseBest(P, pI)$
9: $newSLA_i \leftarrow deriveSLA(sI, P, C)$
10: **else**
11: $newSLA_i \leftarrow deriveSLA(Q, P, C)$
12: $newSLA_i \leftarrow Rhone(newSLA_i)$
13: **end if**
14: $IR \leftarrow execute(newSLA_i)$
15: **return** IR
16: **end function**

2.3 Query rewriting algorithm

To serve as a proof of concept to our approach, we intend to develop a query rewriting algorithm which is guided by users' integration requirements and service level agreements exported by different data services and cloud providers. The query rewriting is an important issue in data integration. In cloud computing, researches have refereed to it as a service composition problem in which given a query the objective is to lookup and compose data services that can contribute to produce a result. [2] proposed a query rewriting approach which processes queries on data provider services. [4] introduced a service composition framework to answer preference queries. Two algorithms inspired on [2] are presented to rank the best rewritings based on previously computed scores. [1] extended [7] and presented an refinement algorithm that produces and order rewritings according to user preferences and scores. In general, these works share the same performance problem depending on the size of the query and

on the number of available services. Furthermore, they do not take into consideration user's integration requirements what can lead to produce rewritings that are not satisfactory to the user in terms of quality requirements and cost. Currently, we have formalized and developed a rewriting algorithm that considers user preferences and services' quality aspects while selecting services and producing rewritings called *Rhone* (Invoked in the algorithm 1, line 12).

3. PRELIMINARY RESULTS

We have developed a prototype of our query rewriting algorithm (called *Rhone*) which takes into consideration users' requirements and services' quality aspects extracted from SLAs. The *Rhone* prototype is implemented in Java.

Currently, our approach runs in a local controlled environment including a service registry of 100 concrete services. Experiments were produced to analyze the algorithm's behavior concerning performance, and quality and cost of the integration. The figures 3 and 4 summarize our first results.

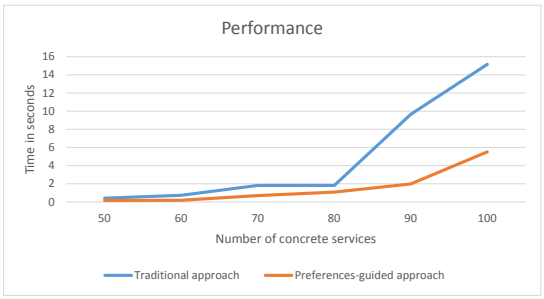

Figure 3: Performance evaluation.

The experiments include two different approaches: (i) the *tradition approach* in which user preferences and SLAs are not considered; and (ii) the *preference-guided approach* which considers the users' integration requirements and SLAs.

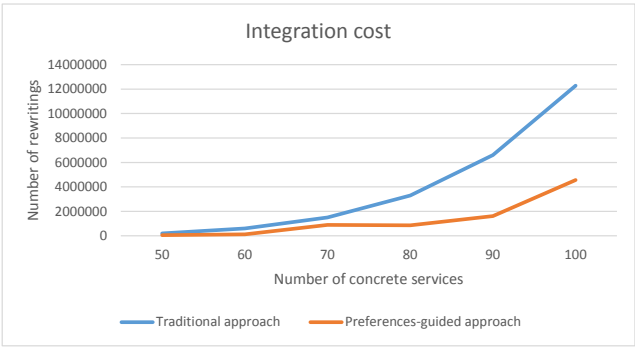

Figure 4: Performance evaluation.

The results while considering user preferences and SLAs are promisingly. The *Rhone* increases performance reducing rewriting number which allows to go straightforward to the rewriting solutions that are satisfactory avoiding any further backtrack and thus reducing successful integration time 3. Moreover, using the *preferences-guided approach* to meet the user preferences, the quality of the rewritings produced has enhanced and the integration economic cost has considerable reduced while delivering the expected results 4.

4. CONCLUSIONS

This paper introduces a new vision of data integration adapted to the cloud economic model. It also proposes a new approach for data integration based on user integration requirements and SLA. In addition, an query rewriting algorithm called *Rhone* servers as proof of concept. Our preliminary results have shown that the *Rhone* reduces the rewriting number and processing time while considering user preferences and services' quality aspects extracted from SLAs to guide the service selection and rewriting. Furthermore, the integration quality is enhanced, and it is adapted to cloud economic model reducing the total cost of the integration.

5. ADDITIONAL AUTHORS

This PhD project is co-supervised by Nadia Bennani (LIRIS - INSA Lyon, France, `nadia.bennani@insa-lyon.fr`), and Genoveva Vargas-Solar (CNRS-LIG, Grenoble, France, `genoveva.vargas@imag.fr`), with inputs from Plácido A. Souza Neto (IFRN, Natal, Brazil, `placido.neto@ifrn.edu.br`).

6. REFERENCES

[1] C. Ba, U. Costa, M. H. Ferrari, R. Ferre, M. A. Musicante, V. Peralta, and S. Robert. Preference-driven refinement of service compositions. In *Int. Conf. on Cloud Computing and Services Science, 2014*, Proceedings of CLOSER 2014, 2014.

[2] M. Barhamgi, D. Benslimane, and B. Medjahed. A query rewriting approach for web service composition. *Services Computing, IEEE Transactions on*, 3(3):206–222, July 2010.

[3] N. Bennani, C. Ghedira-Guegan, M. Musicante, and G. Vargas-Solar. Sla-guided data integration on cloud environments. In *2014 IEEE 7th International Conference on Cloud Computing (CLOUD)*, pages 934–935, June 2014.

[4] K. Benouaret, D. Benslimane, A. Hadjali, and M. Barhamgi. FuDoCS: A Web Service Composition System Based on Fuzzy Dominance for Preference Query Answering, Sept. 2011. VLDB - 37th International Conference on Very Large Data Bases - Demo Paper.

[5] D. A. S. Carvalho, P. A. Souza Neto, G. Vargas-Solar, N. Bennani, and C. Ghedira. *Database and Expert Systems Applications: 26th International Conference, DEXA 2015, Valencia, Spain, September 1-4, 2015, Proceedings, Part II*, chapter Can Data Integration Quality Be Enhanced on Multi-cloud Using SLA?, pages 145–152. Springer International Publishing, 2015.

[6] G. Correndo, M. Salvadores, I. Millard, H. Glaser, and N. Shadbolt. SPARQL query rewriting for implementing data integration over linked data. In *Proceedings of the 1st International Workshop on Data Semantics - DataSem '10*, page 1, New York, New York, USA, Mar. 2010. ACM Press.

[7] U. Costa, M. Ferrari, M. Musicante, and S. Robert. Automatic refinement of service compositions. In F. Daniel, P. Dolog, and Q. Li, editors, *Web Engineering*, volume 7977 of *Lecture Notes in Computer Science*, pages 400–407. Springer Berlin Heidelberg, 2013.

[8] O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '97, pages 109–116, New York, NY, USA, 1997. ACM.

[9] G. ElSheikh, M. Y. ElNainay, S. ElShehaby, and M. S. Abougabal. SODIM: Service Oriented Data Integration based on MapReduce. *Alexandria Engineering Journal*, 52(3):313–318, Sept. 2013.

[10] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, Dec. 2001.

[11] A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 251–262, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[12] T. Nie, G. Wang, D. Shen, M. Li, and G. Yu. Sla-based data integration on database grids. In *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International*, volume 2, pages 613–618, July 2007.

[13] R. Pottinger and A. Halevy. Minicon: A scalable algorithm for answering queries using views. *The VLDB Journal*, 10(2-3):182–198, Sept. 2001.

[14] Y. Tian, B. Song, J. Park, and E. nam Huh. Inter-cloud data integration system considering privacy and cost. In J.-S. Pan, S.-M. Chen, and N. T. Nguyen, editors, *ICCCI (1)*, volume 6421 of *Lecture Notes in Computer Science*, pages 195–204. Springer, 2010.

[15] S. S. Yau and Y. Yin. A privacy preserving repository for data integration across data sharing services. *IEEE T. Services Computing*, 1(3):130–140, 2008.