Profiting from Instant Data Integration on the Cloud

Daniel A. S. Carvalho
(Supervised by Chirine Ghedira-Guegan, Genoveva Vargas-Solar and Nadia Bennani, with inputs from Plácido A. Souza Neto)

Université Jean Moulin Lyon 3, Centre de Recherche Magellan, IAE, France
daniel.carvalho@univ-lyon3.fr

Abstract. This PhD project addresses data integration in multi-cloud environments. Current data integration systems imply consuming data from data services deployed in cloud contexts and integrating the results. The project takes a new angle of the problem by considering different actors (Data providers, data consumer, the infrastructure, and the data itself), and their characteristics and constraints, and proposing an SLA-based data integration approach in a multi-cloud environment which considers users' preferences (including quality constraints and data access requirements) to be matched with the data provider' quality constraints extracted from their SLAs while delivering the results. The objective is to enhance the quality and performance on data integration taking into consideration the economic model imposed by the cloud. At this stage, Our first results have shown that quality and performance can be enhanced, and the cost of the integration can be minimized by using our approach.

Keywords: Data integration. Query rewriting. Query rewriting algorithm. Cloud computing. SLA.

1 Introduction

Current data integration systems imply consuming and integrating data from data services which deliver data under different quality conditions related to freshness, trust, cost, reliability, availability, among others. Selecting data services, producing query rewritings and executing query plans are computationally expensive. These data integration tasks can take advantages from multi-cloud architectures considering their elasticity, *pay-per-use* model and parallel processing.

Multi-cloud environments bring new challenges to data integration due to different entities (data provider, data consumer, infrastructure, and the data) that should be taken into account, and their associated constraints and characteristics (such as access policies, access constraints, processing capacity, memory limits, among others). To better understand our problem, let us suppose the following scenario. During Brazilian Olympic games 2016, Lucas is an spectator willing to collect information about the weather forecast near your actual location with a time interval of two days in advance. Lucas may have several preferences such privacy issues, time interval, budget, using free services or not, for instance. To achieve his needs, there are several data provider services distributed on different clouds that can be integrated to produce an answer for his query or part of it. Thus, given a user query, the integration process deals with different matching problems: (i) matching the *query* and *data provider services* - the data provider services have to produce a result for the query; (ii) matching the *user preferences* and the *quality guarantees* provided by the data provider - the user preferences concern the data itself, the data services and the type of subscription the user has with the clouds; (iii) matching the *user preferences* and *user' type of subscriptions* - the user may have several subscriptions with different clouds that should influence the way to choose the services if the underlying cloud offer more resources to the user, and this cloud can be running out of budget for consuming the necessary resources; and (iv) the *data provider services* and *their type of subscriptions* - the data provider services also have subscription with the clouds, and they can also be out of budget and resources according to their subscription.

In cloud computing, the quality conditions that the user can expect from a service are defined in contracts called service level agreements (SLA). Current SLA models are not sufficient to cover the data integration requirements. Usually, SLAs describe only cloud resources, and do not tackled the data integration aspects. Thus, we strongly believe that: (*i*) a new kind of SLA is need to unify all information regarding the constraints and requirements as a meaning for the integration process; and (*ii*) by using the new SLA, the integration history could be reused as much as possible enhancing the quality and performance in the current data integration solutions. Considering the problems and challenges aforementioned, this PhD project contributes designing a SLA model for data integration. As result, a data integration approach adapted to the vision of the economic model of the cloud is proposed. The originality of our approach consists in guiding the entire data integration solution - while selecting, filtering and composing cloud services, and delivering the results - taking into account (*a*) user preferences statements; (*b*) SLA contracts exported by different cloud providers; and (*c*) 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.

The reminder of this paper is organized as follows. Section 2 discusses the related works.

2 Related works

The related works concerning the problem stated in the previous section can be divided in three topics: (*i*) data integration approaches in the cloud or in service-oriented contexts; (*ii*) query rewriting approaches; and (*iii*) service level agreements for cloud computing.

The authors in [5, 7] perform data integration in service-oriented contexts, particularly considering data services. However, they only take into consideration the requirement of computing resources for integrating data focusing on performance aspects. [10] focused on data privacy in order to integrate data obtained from different data services. [9] proposed an inter-cloud data integration system considering privacy requirements and the cost for protecting and processing data. Although [9, 10] tackled in their approaches quality aspects of the integration, we believe there are other crucial elements that should be studied regarding the requirements and constraints of data consumers, data providers, the associated infrastructures and the data itself, and how to filter services and produce the best query plan considering these requirements and constraints.

As traditional databases theory, data integration on cloud and service-oriented context deals with query rewriting issues. Researches [1–3, 6] have refereed 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. In general, these works share the same performance problem depending on the size of the query and on the number of available services. Although [1, 3] have considered preferences and scores to produce rewritings, the multi-cloud context brings new challenges once new requirements and constraints are introduced. Thus, new heuristics should be considered in the rewriting process in order to make it efficient.

Service level agreements (SLA) have been widely adopted in different domains in order to specify what service consumer can expect from the service delivered by a service provider. Research contributions in cloud computing mainly concern (i) SLA 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. We strongly believe that SLAs could be used in order to cover the limitations discussed in the previous paragraphs and to enhance the quality in the current data integration solutions. In this sense, to the best of our knowledge, we have not identified other works that uses SLA to guide the entire data integration on a multi-cloud context.

3 New vision of data integration

The metamodel in the figure 1 illustrates our new vision of data integration. The *Multi-Cloud* is configured as a set of *Cloud Infrastructures*. *Data producers* and *data consumers* subscribe to *cloud infrastructures*. Their subscription credentials are illustrated thanks to a *SLA* (*Consumer SLA* or *Producer SLA*) defining what the *cloud infrastructure* offers to them through their subscription. *Data* are provided and consumed by *Data Producers* and *Data Consumers*, respectively. *Data consumers* willing to process an *Integration* defines a *Query* according to his/her needs. Then, the *integration process* is triggered selecting *data consumers* deployed on the *multi-cloud* considering the *query* and its *producer SLA*.

Moreover, the process deals also with consumer SLA in order to verify if the consumer has access to the selected producer and if he/she has enough resources for processing the *data producer*'s service. In addition, the *integration process* can also deal with *integration SLAs* verifying whether there is a previous integration request that can be reused for the actual integration. The *integration process* proposes an *integration plan* that retrieves data and integrate the *results*. These results are finally delivered to the consumer according to his/her requirements and constraints. When the integration is done, a new *integration SLA* is created to the last integration including all information about query, consumer requirements and constrains, selected data producers, integration plan, integration time and performance execution.

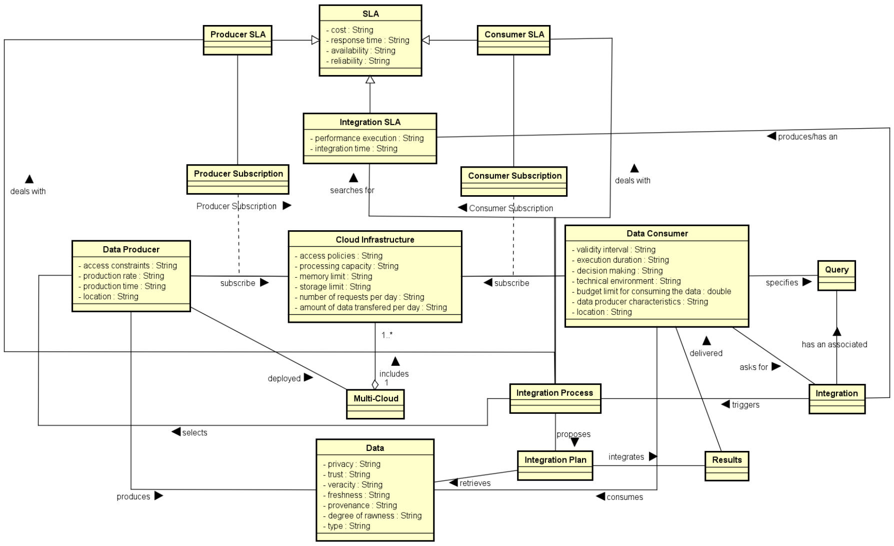


Fig. 1: Data integration metamodel

Therefore, given a *Data Consumer* query, his/her quality requirements and cloud subscription, the query is rewritten in terms of *Data Producers* that fulfill the integration requirements and constraints (associated to *Data Consumers*, *Data Producers* and *Data*), and deliver the expected results. This new vision brings challenges to data integration, such as: (i) Performance issues. Given the huge amount of *Data Providers* in the *Multi-Cloud*, the *Integration* can require a huge amount of cloud resources and processing time; (ii) Economic model. Even with the on-demand and pay-per-use imposed by the cloud, users are limited to the resources they have contracted and to the budget they are ready to pay for while asking for a integration. (iii) Quality issues. Some rewritings produced and executed to the query could not satisfy his/her quality requirements concerning privacy, data provenance, cost, among others. Consequently, users may become dissatisfied with the obtained results; (iv) SLA heterogeneity. In the multi-cloud, *Data Consumers* and *Data Producers* export their *SLAs* with different semantics and structure making the matching of integration requirements and constraints more challenging. Moreover, while integrating services, we can also face incompatibilities of SLAs such as measures with the same meaning but with different names, measures with the same name but with different associated values and units, among others; and (iv) Reuse. Rewriting and executing the *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.

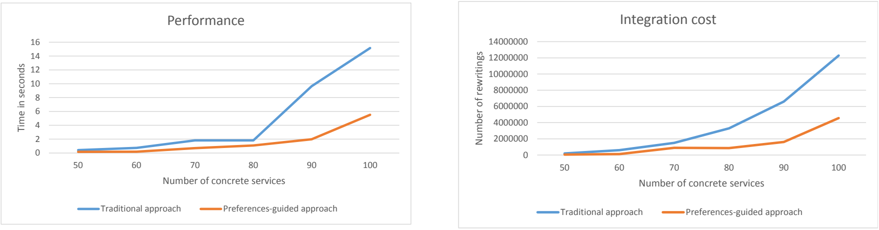
Motivated by these challenges, we propose a SLA-based data integration approach adapted to the multi-cloud which includes: (i) looking for *Data Producers*; (ii) *Data* retrieval and integration; and (iii) delivering results to the user considering their characteristics and constraints. In this sense, our approach is divided in four steps:

Given a user *query*, a set of associated user *preferences* and *Data Consumers*:
SLA derivation. In this step, we compute what we call an *integration SLA* that matches user' integration requirements (including quality constraints and data requirements) with the *SLA*'s provided by *data producers*, given a specific user cloud subscription. The user may have general requirements depending on the context he/she wants to integrate his/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 *integration SLA* guides the query evaluation, and the way results are computed and delivered.
Filtering data services. The *integration SLA* is used (i) to filter previous *integration SLA* derived for a similar request in order to reuse results; or (ii) to filter possible *data producers* that can be used for answering the query.
Query rewriting. Given a set of *data producers* that can potentially provide data for integrating the query result, a set of service compositions is generated according to the *integration SLA* and the agreed *Producer SLA* of each *data producer*.
Integrating a query result. The service compositions are executed with services from one or several clouds where the user has a subscription. The execution cost of service compositions must fulfill the *integration SLA*. The clouds resources needed by the user 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.

4 Preliminary results

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

Currently, our approach runs in a local controlled environment simulating a single-cloud 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 2a and 2b summarize our first results.



(a) Performance evaluation.

(b) Integration cost.

Fig. 2: *Rhone* execution evaluation.

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

The results P-GA 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 (Figure 2a). Moreover, using the P-GA to meet the user preferences, the quality of the rewritings produced has been enhanced and the integration economic cost has considerable reduced while delivering the expected results (Figure 2b). However, the *Rhone* still need to be tested in a large scale case and in a context of parallel multi-tenant to test efficacy.

5 Conclusions and Research plan

This paper introduces a new vision of data integration adapted to the cloud context and to the end-user pay-per-use economic model. It also proposes a new approach for data integration based on user requirements and SLA. In addition, a query rewriting algorithm called *Rhone* serves as proof for the feasibility of data integration process guided by cloud constraints and user preferences . Our first results are promisingly. 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.

SLA incompatibilities are not treated in this paper. Currently we are working on this issue, and improving our SLA model and schema for data integration adapted to the multi-cloud context. Another important part is how to make efficient the rewriting process by reducing the composition search space. Finally, how should be a parallel execution of the query plan to let the execution efficient in the multi-cloud. In addition, we are focusing on evaluating and validating the entire quality-based data integration approach on a multi-cloud environment.

References

1. Ba, C., Costa, U., H. Ferrari, M., Ferre, R., A. Musicante, M., Peralta, V., Robert, S.: Preference-driven refinement of service compositions. In: Int. Conf. on Cloud Computing and Services Science. Proceedings of CLOSER 2014 (2014)
2. Barhamgi, M., Benslimane, D., Medjahed, B.: A query rewriting approach for web service composition. Services Computing, IEEE Transactions on Services Computing (2010)
3. Benouaret, K., Benslimane, D., Hadjali, A., Barhamgi, M.: FuDoCS: A Web Service Composition System Based on Fuzzy Dominance for Preference Query Answering (2011), 37th International Conference on Very Large Data Bases (VLDB 2011)
4. Carvalho, D.A.S., Souza Neto, P.A., Vargas-Solar, G., Bennani, N., Ghedira, C.: Can Data Integration Quality Be Enhanced on Multi-cloud Using SLA?, pp. 145–152. Springer International Publishing (2015)
5. Correndo, G., Salvadores, M., Millard, I., Glaser, H., Shadbolt, N.: SPARQL query rewriting for implementing data integration over linked data. In: Proceedings of the 1st International Workshop on Data Semantics - DataSem '10. ACM Press, New York, New York, USA (2010)
6. Costa, U., Ferrari, M., Musicante, M., Robert, S.: Automatic refinement of service compositions. In: Web Engineering, Lecture Notes in Computer Science, vol. 7977. Springer Berlin Heidelberg (2013)
7. ElSheikh, G., ElNainay, M.Y., ElShehaby, S., Abougabal, M.S.: SODIM: Service Oriented Data Integration based on MapReduce. Alexandria Engineering Journal (2013)
8. Nie, T., Wang, G., Shen, D., Li, M., Yu, G.: Sla-based data integration on database grids. In: Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International. vol. 2, pp. 613–618 (July 2007)
9. Tian, Y., Song, B., Park, J., nam Huh, E.: Inter-cloud data integration system considering privacy and cost. In: ICCCI. Lecture Notes in Computer Science, vol. 6421, pp. 195–204. Springer (2010)
10. Yau, S.S., Yin, Y.: A privacy preserving repository for data integration across data sharing services. IEEE T. Services Computing 1 (2008)