On Adaptive Integration of Web Data Sources into Applications*

Roman Khazankin
Distributed Systems Group
Information Systems Institute
Vienna University of Technology
Vienna, Austria
e-mail: khazankin@infosys.tuwien.ac.at

Schahram Dustdar
Distributed Systems Group
Information Systems Institute
Vienna University of Technology
Vienna, Austria
e-mail: dustdar@infosys.tuwien.ac.at

Abstract[1]

Software applications often use data from web data sources, employing different integration techniques. The choice of a technique is made by an analyst or a developer traditionally. As non-functional characteristics of a web source and the business requirements of the application can change over time, the technique, that fits the situation best, might also vary. The manual control of the large number of integrated sources can be cumbersome as their characteristics might change independently. This paper gives an initial idea of (i) how the choice of a technique can be automated and (ii) how to make the integration adaptive to the environment.

1. Introduction

Software applications often use data from web data sources. This data can be used by applications employing different integration techniques. The choice of a technique impacts non-functional properties of utilized sources as from the perspective of the ultimate application. Because of the web's nature, the non-functional characteristics of web sources can change over time, so the choice of the most appropriate technique for data integration might differ. This is shown in motivating scenarios later in the paper.
The selection of data integration technique is driven by the business requirements on the one hand, and by the characteristics of the integrated data sources on the other hand. Such a choice is made by an analyst or a developer traditionally. However, with proliferation of online data sources, applications tend to use more remote sources [5]. If the application integrates a large number of independent web sources, the manual control of integration techniques can be cumbersome.

This paper shows how (i) the choice of technique can be automated and (ii) how the integration can be made adaptive to the environment therefore allowing for seamless switching of the technique to the most appropriate one according to specified criteria.
The approach's scope is applications that continuously use data from web sources. It does not address any scenarios where integrated data can be changed, so these changes need to be propagated back to the origin sources.

The paper is organized as follows: Section 2 describes the main integration techniques, Section 3 explains the factors that influence the choice of an integration technique, Section 4 depicts the motivating scenarios for the proposed approach, Section 5 describes the approach formally, Section 6 describes the basic architecture for the approach, Section 7 discusses the related work, Section 8 concludes the paper.

2. The techniques

The three main techniques used for integrating data are data *consolidation*, data *federation*, and data *propagation* (depicted in Fig. 1). All techniques are aimed to provide data from one or multiple origin sources in a specific form to the data consumer.
Data *consolidation* captures data from multiple sources and stores it into a single persistent target data store.
Data *federation* provides a single virtual view of one or more data sources. When the query is posed against federated view, it is forwarded to the origin sources. The responses are then combined and returned as a result.

---

[1] **Proceedings of the International Workshop "Innovation Information Technologies - Theory and Practice", September 6th-10th, Dresden, Germany, 2010**

*** This work was supported by the Vienna Science and Technology Fund (WWTF), project ICT08-032.**

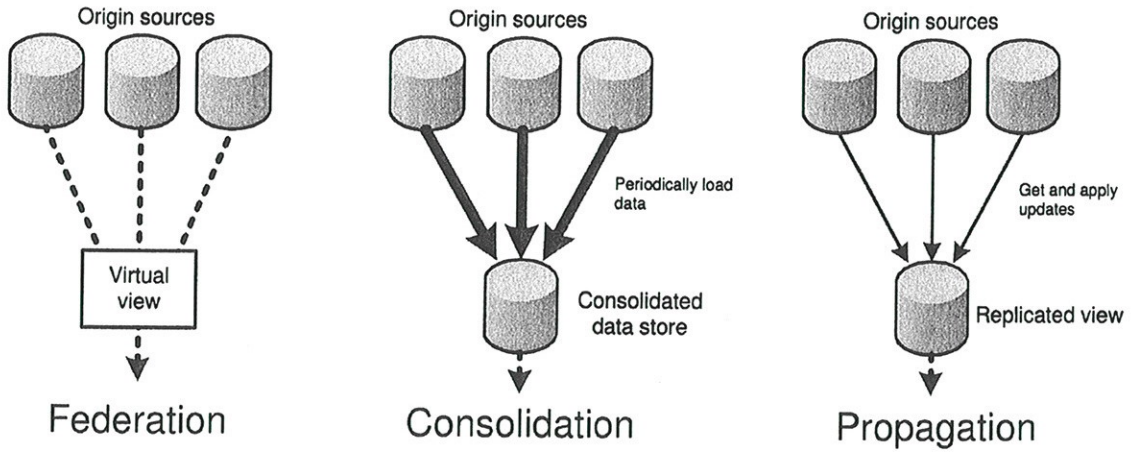

Figure 1: Main data integration techniques

Data *propagation* captures the changes in the source system and applies them to the replicated data store. Federation provides real-time data and requires no additional storage, but makes the data consumer dependent on the origin sources' Quality of Service (QoS) properties and increases workload on them proportionally to the request frequency and requested data volume.

Consolidation guarantees QoS, but provides data with higher timeliness, requires additional storage and periodical workload on both origin and consumer systems proportionally to the requested data volume and the refresh rate.

Propagation guarantees QoS and provides near real-time data, but requires additional storage and increases workload proportionally to the update ratio. Propagation also requires tighter coupling as the access to the change log is required.

As a simple example, consider that company *First, Inc.* uses products from its partner, *Second GmbH*, to provide some services to their customers. To confirm its ability to provide the service to a customer, *First* needs the confirmation of the appropriate amount of products available from *Second*. Let *Second* have the data source with this information exposed. *First* needs to integrate this source into its system to use this information. As shown in Fig. 2, the choice of integration technique impacts the source's characteristics from the perspective of the ultimate application.

3. The technique choice

The choice of most suitable technique is driven by such concerns as timeliness of data, data volume, network latency, throughput, and availability of DS on the one hand, and by the corresponding requirements of consuming application (like latency or throughput) on the other hand. The examples of concerns and requirements regarded are listed in Table 1.

Table 1. Examples of data source offerings and data integration requirements

Offerings	Requirements
Timeliness	Timeliness
Latency	Latency
Throughput	Throughput (Request ratio)
Availability	Availability
Change ratio	Offline usage capability
Size of the source	
Update log availability	**General criterion**
Data storing permission	Disk space saving
	Workload saving
	Data transfer reduction

Different goals imply different requirements. For example, if the integrated source is frequently used by a few users at a time, e.g., for decision making, then the latency will be crucial; if the source is used to provide some information on company's website, then the throughput might be more important.

Enabling the analyst to set the priorities, we can automate the selection of most suitable technique. Having the priorities and/or restrictions defined, we can calculate the actual value for each criterion for each technique, and, finally, choose the technique which has the most satisfying calculated values according to priorities and/or restrictions.

4. Motivating scenarios

In this section we show the scenarios of applying the adaptive integration with explicit benefits.

4.1. Change of offerings

Extending the scenario from Sec. 2, let *First* receive some of the orders by phone, so it desires to confirm the availability in a short period of time (e.g., 3 seconds) to increase customer satisfaction. The required timeliness of data is 1 hour at maximum. Let *Second* have 10000 products which First uses. Usually *Second* provides data with mean latency=1 sec, so *First* employs federation which satisfies the requirement.
Now consider that network problems arise for two days and from the *First*'s perspective the latency increases to 5-10 sec, which hampers its business. Using the adaptive technique, the system can automatically detect that, for example, every-hour consolidation becomes more appropriate for the current situation, and can switch the technique, while the rest of the system would work as usual. After 2 days the network problems are fixed and adaptive technique switches back to federation to eliminate unnecessary periodical load.

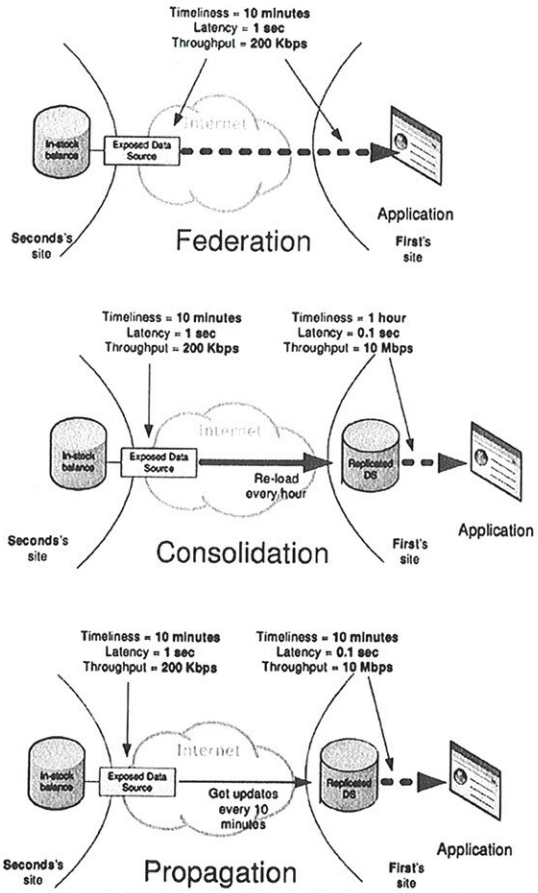

Figure 2. Impact of the technique's choice

While consolidation of one data source does not bring much overhead, it would make an impact when a large number of sources need to be integrated. Thus, in the case when an application integrates numerous external sources whose performance can change over time, the adaptation would optimize the disk space used by consolidated sources, the network load, and the workload caused by loads and transformations by automatically switching the technique used. Therewith, the manual control for a large number of integrated sources will be cumbersome.

4.2. Change of requirements

During the reporting period, the reports are generated much more frequently in a company. If these reports incorporate data from external sources (e.g., from the partners in virtual enterprise), then the throughput requirements become higher, so it might be more appropriate to use consolidation or propagation for integrating these sources, while normally federation is enough. The adaptive integration is able to seamlessly make decisions and automate such switching.

5. Formal specifications

In this section we formally outline the adaptive switching logic.
Let's assume that we regard N characteristics (or offerings) of a data source. Let $L(O) \in R$ - loss function that indicates how much penalty the characteristics $O = \{o_1,...,o_N\}$ result in being relevant for a short space of time Δt. L should be defined by the analyst and reflect the business requirements for the integrated source.
Let $C(T,P,O) = \{o'_1,...,o'_N\}$ define the influence of technique T applied with parameters P on the given set of offerings O. C should be defined by the system engineer and reflect the impact of the current technique implementation on data source characteristics as from the perspective of the ultimate application.
Let's assume that the offerings of the origin source are logged such as $o_i(l)$ - mean value of i_{th} offering in a l_{th} space of time, $i = \{1,..,N\}$. The length of each period is Δt. Let $R(l) = \{o_1(l),...,o_N(l)\}$.
Let $\Phi = \{l_1,l_2,....,l_M\}$ - a set of periods that form a basis for the decision making regarding the technique choice, such as last hour or last day.
Let $L_{fact}(l)$ - factual losses for l_{th} period, Ω - switching threshold.
Now, if there exist such $\{T,P\}$ that satisfy the following inequality, then the decision should be made in favour of technique T with parameters P:

$$\sum_{l \in \Phi} L_{fact}(l) - \min \sum_{l \in \Phi} L(C(T, P, R(l))) > \Omega$$

The rationale of this inequality is the following: it selects the technique which would have resulted in the least losses for the time period Φ, but the difference should be big enough to prevent from switching the technique too often, which is ineffective as the switching itself causes the loss. Ω reflects the cost of technique switch and depends on the particular environment and behavioral characteristics of the data sources and data integration tools used.

6. Adaptive integration architecture

This section describes the basic architecture of adaptive integration. As shown in the Fig. 3, the data from a web data source is provided to the end application via the ``proxy'' source that acts as a wrapper using one of integration techniques underlying. The technique swapping is controlled by the adaptation module which makes decisions based on Loss function, Impact function, and the offerings log (see previous Section). The log is populated with the values either from web data source description via specification parser or from factual measurements via QoS measuring module.

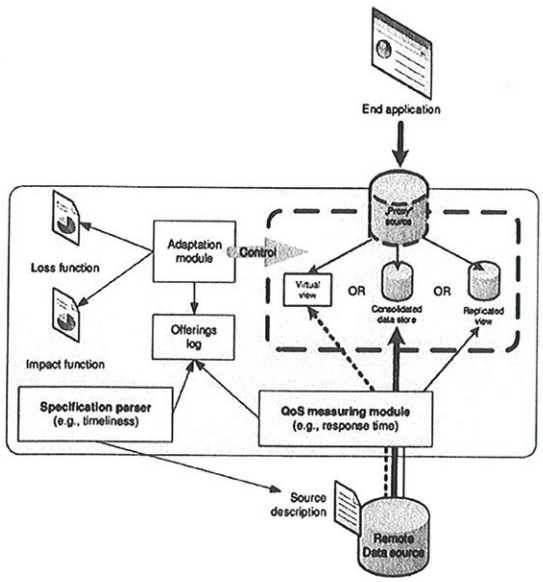

Figure 3. Adaptive integration architecture

The control module should check periodically if the currently used technique is still the best. When the decision about technique change is made, the tool implementing the new technique should be first prepared to serve the requests from the application, and only then the proxy can be re-bound to it. This ensures that the technique switch does not interrupt the application's work.

7. Related work

To the best of the authors' knowledge, no similar work has been done. The closest related area is adaptive databases or web pages caching [1,2,3,4].
The main difference is that the adaptation in those approaches aims to optimize the source system's performance, but not to fulfill the data consumer's individual business requirements.

8. Conclusion and future work

The initial ideas of adaptive data integration of web sources were presented in this paper. The problem was depicted and motivating scenarios were shown. Future work includes the development of real-world templates of impact and loss functions, the proper study of integration technique parameters, the analysis of applicable integration tools, prototype implementation.

References

1. C. Bornhoevd, M. Altinel, C. Mohan, H. Pirahesh, and B. Reinwald. Adaptive database caching with DBCache. *Data Engineering*, 27(2):11–18, 2004.
2. L.W.F. Chaves, E. Buchmann, F. Hueske, and K. Boehm. Towards materialized view selection for distributed databases. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 1088–1099. ACM, 2009.
3. S. Guirguis, M.A. Sharaf, P.K. Chrysanthis, A. Labrinidis, and K. Pruhs. Adaptive scheduling of web transactions. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, pages 357–368, 2009.
4. A. Labrinidis, Q. Luo, J. Xu, and W. Xue. Caching and Materialization for Web Databases. *Databases*, 2(3):169–266, 2009.
5. Xiaofang Zhou, Shazia Sadiq, and Ke Deng. Data quality in web information systems. In *WISE '08: Proceedings of the 9th international conference on Web Information Systems Engineering*, Berlin, Heidelberg, 2008. Springer-Verlag.