Building Dynamic Market Places Using HyperQueries*

Christian Wiesner, Peter Winklhofer, and Alfons Kemper

Universität Passau
Fakultät für Mathematik und Informatik
D-94030 Passau, Germany
⟨*wiesner*|*winklhof*|*kemper*⟩@db.fmi.uni-passau.de
http://www.db.fmi.uni-passau.de/

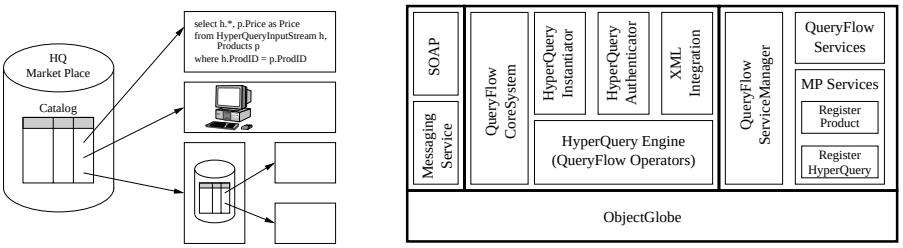
1 Motivation and Introduction

Electronic market places and virtual enterprises have become important applications for query processing [2]. Building a scalable virtual B2B market place with hundreds or thousands participants requires highly flexible, distributed query processing capabilities. Architecting an electronic market place as a data warehouse by integrating *all* the data from *all* participating enterprises in one centralized data repository incurs severe problems:

- *Security and privacy violations:* The participants of the market place have to relinquish the control over their data and entrust sensitive information to the market place host.
- *Coherence problems:* The coherence of highly dynamic data, such as availability and shipping information, may be violated due to outdated materialized data in the market place's data warehouse.
- *Schema integration problems:* Using the warehouse approach all relevant data from all participants have to be converted à priori into the same format. Often, it would be easier to leave the data inside the participant's information systems, e.g., legacy systems, within the local sites, and apply particular local wrapper operations. This way, data is only converted *on demand* and the most recent coherent state of the data is returned.
- *Fixed query operators:* In a data warehouse-like electronic market place, all information is converted into materialized data. This is often not desirable in such complex applications like electronic procurement/bidding.

We propose a reference architecture for building scalable and dynamic market places and a framework for evaluating so-called *HyperQueries* in such an environment. HyperQueries are essentially query evaluation sub-plans "sitting behind" hyperlinks. This way the electronic market place can be built as an intermediary between the client and the providers executing their sub-queries referenced via hyperlinks. The hyperlinks are embedded as attribute values within data objects of the intermediary's database (Figure 1(a)). Retrieving such a virtual object automatically initiates the execution of the referenced HyperQuery in order to materialize the entire object.

* This work was supported by the German National Research Council (DFG) under Contract Ke 401/7-1



(a) Hyperlinks referencing HyperQueries (b) Architecture

Fig. 1. Overview of the QueryFlow System

2 Architecture of the QueryFlow System

We propose a reference architecture for building scalable electronic market places. During the implementation of our prototypical system, we payed special attention to rely on standardized (or proposed) protocols such as XML, XML Schema, XML Signature, X.509 certificates, HTTP and SOAP. Figure 1(b) depicts the basic components of the system, that can be described as follows:

- The query processing capabilities of the QueryFlow system are based on the *ObjectGlobe* system [1], which is a distributed and open query processor for data sources on the Internet.
- The communication between multiple hosts is implemented via a proprietary *Messaging Service*. Additionally, *SOAP* provides an open interface both to integrate other systems and to be integrated by other systems.
- The *HyperQuery Engine* combines all operators that are specific for HyperQuery processing, i.e., for resolving the hyperlinks to HyperQueries and operators for optimization of HyperQuery execution.
- The *QueryFlow CoreSystem* manages the instantiated HyperQueries and the distribution of the execution to multiple physical hosts. Furthermore all additional data structures such as caches and administrative data of the executed HyperQueries are managed within this component.
- The *HyperQuery Instantiator* manages the instantiation of HyperQueries at the remote sites. The HyperQueries are stored in a hierarchical repository.
- The certificate-based *HyperQuery Authenticator* is used for signing and verifying requests and queries when communicating with other sites.
- The *XML Integration* component collects all functionality to access XML data sources and to handle semi-structured data.
- The extensible *ServiceManager* for executing Java-based e-services is required for administrative issues of the market place. The services can be accessed via the SOAP interface of the QueryFlow system.

A full description of these components can be found in [4].

3 Processing HyperQueries in the QueryFlow System

We demonstrate the HyperQuery technique with a scenario of the car manufacturing industry. We assume a hierarchical supply chain of suppliers and subcontractors. A typical process of e-procurement to cover unscheduled demands

of the production is to query a market place for these products and to select the incoming offers by price, terms of delivery, etc. The price of the needed products can vary by customer/supplier-specific sales discounts, duties, etc.

In traditional distributed query processing systems such a query can only be executed if a global schema exists or all local databases are replicated at the market place. Considering an environment, where hundreds of suppliers participate in a market place, one global query which integrates the sub-queries for all participants would be too complex and error-prone.

Following our approach the suppliers have to register their products at the market place and specify by hyperlinks the sub-plans to compute the price information at *their* sites. These hyperlinks to sub-plans are embedded as virtual attributes into the tables of the market place. When evaluating these hyperlinks, our QueryFlow system distinguishes between two modes: In hierarchical mode the initiator of a HyperQuery is in the charge of collecting the processed data. Under broadcast mode data objects are routed directly to the query initiator. The decision, which processing mode is used, relies—with some restrictions—only to the initiator of a HyperQuery. Thus, the initiator determines, if the results should be sent directly to the client, or if the initiator is in charge of collecting the objects processed by the HyperQueries. Both processing modes can be mixed and nested to obtain arbitrary complex scenarios and involve subcontractors, too. A detailed description including security, optimization issues, and implementation details can be found in [3, 4].

4 The Demo

Our techniques are applicable in many different scenarios. In this demo we show the benefits of our approach with an example B2B market place of the car manufacturing industry. This demo shows that the QueryFlow system enables

1. a company to join an existing market place with very little effort. This is done by registering the products with URIs at the market place and providing HyperQueries at the company. Furthermore the company can determine which operators can be used for the evaluation of requests.
2. a client to simply state queries to the market place that evaluates them by executing HyperQueries at the participating companies.

References

1. R. Braumandl, M. Keidl, A. Kemper, D. Kossmann, A. Kreutz, S. Seltzsam, and K. Stocker. ObjectGlobe: Ubiquitous query processing on the Internet. *The VLDB Journal: Special Issue on E-Services*, 10(3):48–71, August 2001.
2. A. Jhingran. Moving up the food chain: Supporting E-Commerce Applications on Databases. *ACM SIGMOD Record*, 29(4):50–54, December 2000.
3. A. Kemper and C. Wiesner. HyperQueries: Dynamic Distributed Query Processing on the Internet. In *Proc. of the Conf. on Very Large Data Bases (VLDB)*, pages 551–560, Rome, Italy, September 2001.
4. A. Kemper and C. Wiesner. HyperQueries: Dynamic Distributed Query Processing on the Internet. Technical report, Universität Passau, Fakultät für Mathematik und Informatik, October 2001. Available at http://www.db.fmi.uni-passau.de/publications/papers/HyperQueries.pdf.