CIGMA: aCtive Inventory service in Global e-MArket for ordinary online customers based on a real time update scheme

Seungwoo Kang , Su Myeon Kim, Kyungmin Cho,
Inseok Hwang and Junehwa Song[a]

Department of Electrical Engineering and Computer Science, KAIST
373-1 Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea
Tel. +82-42-869-5474, Fax. +82-42-869-3410, Email: {swkang,smkim,phocion,saber}@nclab.kaist.ac.kr

[a] *IBM T.J. Watson Research Center*
P.O. Box 218, Yorktown Heights, N.Y. 10598
Tel. +1-914-784-6431, Email: junesong@us.ibm.com

Abstract

In a highly dynamic e-commerce environment, ordinary online customers may feel that online shopping is not comfortable. Information on various items may undergo constant changes due to frequent changes in business decision. In addition, there could be a flood of similar shopping sites. Unfortunately, existing service models or systems are not effective in supporting ordinary online customers in such a dynamic environment. The CIGMA service is proposed to address the problems in on-line shopping in such an environment. It provides catalog comparison and purchase mediation services over multiple shopping sites for ordinary online customers. Equipped with a high performance caching system, the CIGMA responds very quickly upon customers' requests. A key characteristic of the CIGMA is that the service is based on up-to-date information by reflecting the frequent changes in catalog information in real-time. This paper presents the CIGMA system focusing on its real-time update scheme.

Keywords: Electronic Commerce, Service Model, Service System, Real time Update

1. Introduction

Rapid expansion of the Internet introduces a high level of dynamics to business environment. Such an environment may open up an opportunity for merchants to increase their benefits. However, to ordinary customers, it may introduce a high degree of inconvenience due to the increased complexity in their shopping.

The source of such dynamics and complexity and thus potential inconvenience in on-line shopping can be observed in two aspects. First, information on sales items may undergo constant changes. This is so since, in an Internet connected environment, business processes can be promptly summarized and reported, and decisions can instantaneously be reflected. Such a situation may result in very frequent changes in sales conditions. Suppose that the sales conditions for popular goods are changed several times a day. Customers may not be sure whether the conditions they saw at the merchant site ten minutes ago are correct. Thus, they can fall in a situation in which making shopping decision becomes very difficult. Second, there could be a number of similar shopping sites on the Internet. For example, there are 7,513 registered sites in the Yahoo booksellers' directory[1]. Thus, it is difficult for online customers to be sure if the sites they chose are the best ones. Thus, they may wander around to look for other shopping sites with better sales conditions. To make matters worse, on-line merchants compete to provide better sales conditions than others. Such competition may cause a chain reaction among similar shopping sites within a short time.

To overcome such inconvenience in online shopping, customers need proper support. However, existing service models or systems cannot effectively reflect such a highly dynamic environment and support ordinary online customers. To address the problem, we have proposed a new EC service called "the aCtive Inventory service in Global e-MArket" (CIGMA)[24, 25]. The CIGMA provides catalog comparison services over lots of shopping sites for ordinary online customers. In [25], we described the CIGMA along with merchant-side service deployment and maintenance. In this paper, we describe the CIGMA focusing on the issues related to its real-time update scheme.

Price comparison services [1, 2, 3, 4] can partly address the problems arising from the situation where too many similar shopping sites co-exist. It gathers price information for an item from many shopping sites. Then, it provides customers with the comparison information. However, it cannot guarantee the correctness of the comparison information because it generally updates the gathered price information periodically. So, the comparison information may not be up-to-date at the moment of access. In the end, customers need to visit the original sites to check the validity of the given prices. There also exists an approach based on instant gathering of price data [5]. This approach may provide up-to-date price comparison information. However, it may incur a long delay, which most online customers cannot endure.

With the real-time update scheme, the CIGMA effectively supports customers to shop online even under a

[1] From Yahoo Web site(www.yahoo.com) on Mar. 14th 2002.

highly dynamic e-commerce environment. Customers can easily choose the best sales conditions and do not have to undergo exhaustive surfing over shopping sites. The CIGMA also provides purchase transaction mediation. By using this mechanism, online customers can buy goods from different shopping sites at the CIGMA site without visiting original shopping sites. We believe that customers can save time as well as money.

For the merchants, the CIGMA provides them with a chance to increase their business profit. The CIGMA can be considered as a kind of a sales agency that sells merchants' items on behalf of them. The CIGMA is a system that will have high visibility to many customers, and thus provides merchants with an opportunity to be exposed to a large number of customers.

This paper is organized as follows. In Section 2, the design of the CIGMA is briefly sketched. Section 3 describes the real-time update scheme. Section 4 discusses related work in two aspects: existing e-commerce services and cache consistency mechanisms in dynamic data caching. We conclude this paper in Section 5. Other technical aspects of the CIGMA can be found in [24, 25] along with a detailed design and an example service.

2. System Design

The CIGMA consists of five server components and one remote component. The remote one, called Merchant Wrapper (MW), runs on each merchant server. The server components are Client Manager (CM), Merchant Manager (MM), Transaction Manager (TM), System & DB Manager (SDBM) and Catalog Cache DB (CCDB). Figure 1 shows the CIGMA system along with a merchant and a customer.

There are three kinds of external service in the CIGMA: catalog browsing, item purchase, and catalog update. The first two are for customers and the last, for merchants with each requests being handled differently. The internal service flows for these requests are shown in Figure 1.

The catalog browse request is the most frequent one. The Web server that is a part of the CM first receives it. The CM then parses the request and constructs an equivalent DB query string. It sends the query string to the CCDB to retrieve the requested information such as a catalog comparison table, an entire catalog, an item category list, etc. Lastly, it replies to the request with an HTML page that is dynamically generated with the query result.

To handle a purchase request, a safety mechanism is required because it generally includes important information such as a credit card number, address, phone number, etc. To secure the purchase transaction processing, the CM communicates with customers using the secure HTTP protocol. Then, it forwards the received data to the TM. The TM also forwards the data to a proper merchant server at once. The result of the transaction is delivered from the merchant server to the customer in the opposite direction.

In case of an update request, immediate processing is important. The MW module detects each modification of source data in the merchant's DB. Then, the module constructs and sends an update request message to the MM at once. The MM forwards the message to the SDBM after verifying the integrity of the message. Then, the SDBM actually updates the cached data of the CCDB by composing a DB query string based on the message. After the CCDB update, the return value of the update operation (i.e., OK or NOT_OK) is forwarded to the MW in the reverse order.

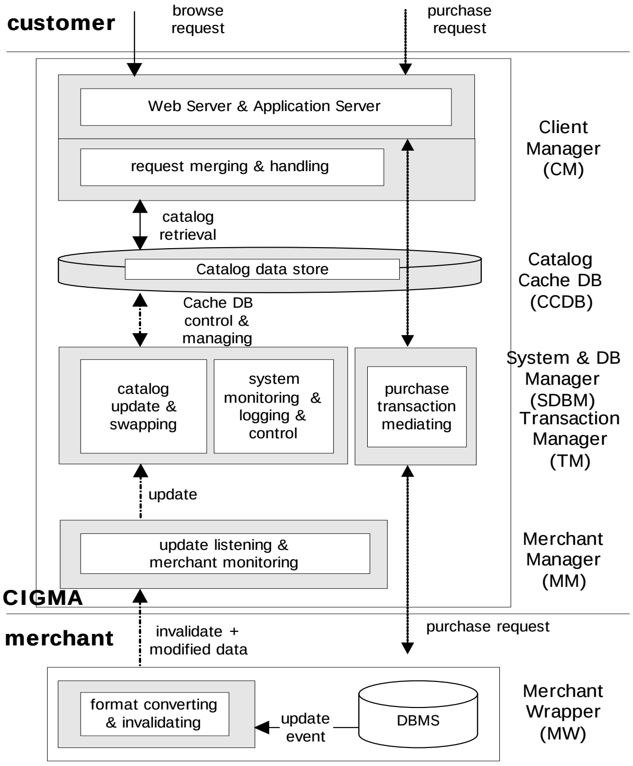
customer
browse request
purchase request
Web Server & Application Server
request merging & handling
Client Manager (CM)
catalog retrieval
Catalog data store
Catalog Cache DB (CCDB)
Cache DB control & managing
catalog update & swapping
system monitoring & logging & control
purchase transaction mediating
System & DB Manager (SDBM)
Transaction Manager (TM)
update
update listening & merchant monitoring
Merchant Manager (MM)
CIGMA
merchant
invalidate + modified data
purchase request
format converting & invalidating
update event
DBMS
Merchant Wrapper (MW)

Figure 1: Service flows and major functions of the CIGMA

2.1 Client Manager (CM)

The CM takes charge of every communication with customers. It includes a Web server and a Web application server. The Web application server is required to generate responses with dynamic contents.[2] Via these servers occurs each interaction with customers. The CM receives both catalog browse requests and purchase requests from customers. As mentioned before, the secure HTTP is used to handle purchase requests with security.

2.2 Merchant Manager (MM)

The MM manages most interactions with merchants except those required for purchase transactions. Upon a catalog modification, the MM receives an invalidation message from a merchant server. Then, the MM

[2] Popular server applications are constructed by using JAVA Servlet, Java Server Page (JSP), Active Server Page (ASP), etc.

immediately notifies the SDBM of the event. There could be a certain situation such as network partitioning, merchant server failure and network congestion, in which the MM cannot receive invalidation messages. This may result in incorrect catalog information. To avoid this problem, the MM monitors heart beat messages from merchants to check the status of servers. If a merchant sends no messages during a predefined period, the MM notifies the SDBM of the situation.

2.3 Catalog Cache DB (CCDB)

The CCDB manages storing and retrieval of cached catalog data. Generally, the performance of DBMS is very critical to the overall performance of an e-commerce site. The CIGMA is expected to process a much higher number of update requests as well as customer requests than general shopping sites. Therefore, the performance of DBMS is even more critical in the CIGMA.

We use main memory as the primary data storage to process a high rate of requests. Disks are used as a supporting medium to compensate the shortage of main memory space. We argue that persistency is less critical in the CIGMA since it deals with only the cached data that has replication in the original merchant servers. In addition, it has been reported that, in disk based caching systems, the overhead of disk accesses is a significant portion of overall load. Thus, in a busy system like the CIGMA with frequent requests and updates, the overhead of keeping data in disks will be even more significant. Our system trades off the lack of persistency wit h such an overhead.

2.4 System & DB Manager (SDBM)

The SDBM takes charge of updating cache data. Upon update requests from MM, the SDBM composes a proper query string based on the incoming request. Then, it updates cache data by sending the query to CCDB. After the update, the return value of the update operation (i.e., OK or NOT_OK) is forwarded to the MW through the MM. The SDBM also monitors the status of the system and controls system components. The SDBM also performs the catalog swapping.

2.5 Transaction Manager (TM)

The TM mediates purchase transactions. This is for providing online customers with a one-stop shopping opportunity. The CIGMA does not process the transaction on behalf of merchants, but simply relays all the purchase-related information, i.e., order form, payment information, buyer information, and transaction result, between a customer and a merchant. When a customer fills out and submits the form, the TM receives the purchase-related information through the CM and forwards it to the merchant server related to that purchase transaction. This approach enables the CIGMA to avoid the complication and overhead that may be incurred by related business issues.

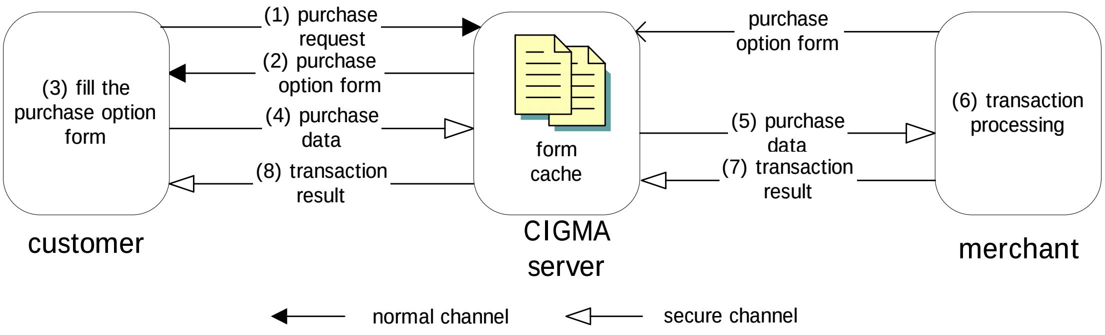

Figure 2: Message flow for Purchase Mediation

Figure 2 shows the basic procedure for the purchase mediation. Upon a customer's purchase request for an item (1), the CIGMA responds by sending the proper order form for the item (2). For the purchase, the customer simply fills out the delivered form (3). Then, upon receiving the purchase-related data from the customer (4), the CIGMA forwards that data to the original merchant (5). After processing the transaction (6), the original server sends the transaction result to the CIGMA (7). Lastly, the CIGMA forwards the result to the customer (8).

The TM mediates purchase transactions in the order of time. An assurance for ordered processing is important when many requests arrive within a short period for a few popular goods with insufficient supply. A fair ordering is guaranteed by tagging messages with its original arrival time at the CM. A purchase request is processed according to the tag until the processing is completed.

2.6 Merchant Wrapper (MW)

The MW covers all the required works for a merchant to interact with the CIGMA. As shown in Figure 1, the MW is deployed on a merchant server for the merchant to join the CIGMA service. Two main functions of the MW are the update propagation and the heart beat message transmission.

The update propagation is performed according to the real-time update scheme, which is described in section 3. The MW also periodically sends out heart beat messages to the CIGMA, if there is no update event during a predefined interval. Although the heart beat message carries no information, the message itself is important because it notifies the CIGMA of the fact that the merchant server is alive. If there are neither update messages nor heart beat messages, the CIGMA regards the merchant as having some critical problems. Thus, the CIGMA can invalidate all catalog data from the merchant to avoid providing catalog data that may be stale.

3. Real-time Update Scheme

Since the major purpose of the CIGMA is to effectively support online customers under the dynamic e-commerce situation, the CIGMA should be able to reflect the constant changes of items' information correctly. In other words, it is very important that the CIGMA should maintain the up-to-date information in terms of the service quality because the customers' decision during online shopping tightly depends on the information provided by the CIGMA.

We propose the real-time update scheme as a method to maintain up-to-dateness of the cached catalog data from multiple merchants. We consider the requirements of the real-time update scheme focusing on the following. Any modification of catalog data in merchant databases should be immediately reflected to the cached catalog data in the CIGMA's CCDB. The scheme should be efficient because a large number of update requests are expected.

Currently, time-to-live (TTL) based schemes are the most popularly used as cache consistency mechanisms in the Web [6]. However, TTL-based schemes are not proper for the CIGMA service since they cannot immediately propagate updates to a cache. Prompt propagation of updates may be achieved if a cache frequently polls changes of servers' data in a very small interval. Frequent polling, however, incurs excessive overhead to the cache. On the contrary, server-push based cache consistency schemes enable caches to be more consistent, but incurs the heavy load on a server [7]. This load comes from the fact that a server should keep track of a large number of caches, which hold copies of original data per data item basis. However, servers that participate in the CIGMA service can avoid suffering from such a heavy burden because the merchant server only takes charge of the CIGMA like a reverse proxy cache. Thus, a server-push based scheme is preferable to the CIGMA.

Now, we explain the details of the real-time update scheme. The real-time update scheme is basically based on server-push approach. The merchant server instantly identifies any modification in the database, and initiates an update in the CIGMA by sending out an invalidation message. The invalidation message piggybacks the modified data including the field value and the field name. Most of modifications will occur in the fields of small size, such as item price and inventory information. Therefore, the piggybacking enables the update to be completed in very small delay and improves processing efficiency. We use the trigger mechanism for the instant identification of the modification in the database. Using the trigger mechanism also contribute to the efficiency because it makes fine-grained invalidation and update possible since a modification can be detected in the unit of a field. Trigger mechanisms are provided in most popular database systems such as Oracle [8], DB2 [9], Sybase [10], MySql [11], PostgresSQL [12], etc.

Figure 3 shows the structure of the MW and the whole process from DB modification to sending an invalidation message. When an update event occurs at the merchant DB (1), the Update Trigger[3] is automatically called by the merchant's DBMS and the Update Trigger sends the modified data to the Event Reporter (2). The Event Reporter sends the modification information to the MW that is running on the merchant server (3). The

[3] The Update Trigger is a kind of DB trigger that is constructed by using SQL statement. It is installed in the merchant DB in advance.

modification information includes the table ID, the primary key, the field name, and the modified field value. The Event Listener within the MW receives the information. Catalog Converter converts the category as well as the format of the modification information, if needed (4) referring Catalog Mapping Table[4] (5). The Catalog Converter composes an invalidation message (6). Lastly, the Communication Module sends the message to the CIGMA (7). The remaining update process is performed internally in the CIGMA through the MM and the SDBM mentioned in previous section.

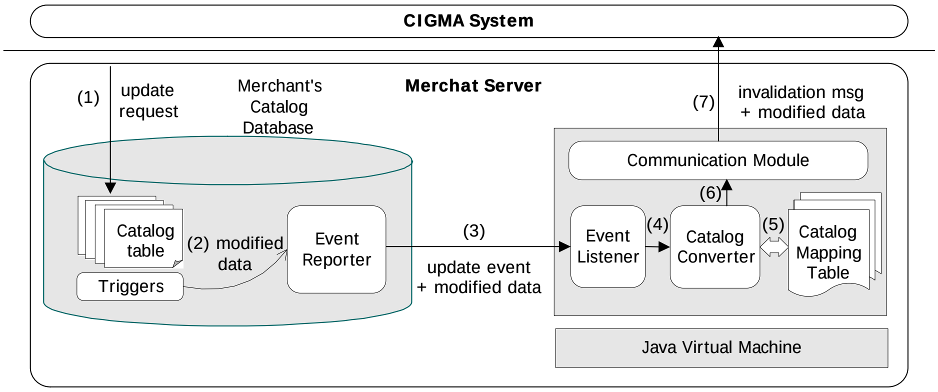

Figure 3: Update process and the structure of the Merchant Wrapper at a merchant server

4. Related Work

As described before, price comparison services [1, 2, 3, 4] are similar to our work in terms of providing comparison information. However, they do not guarantee the accuracy of the provided information because they update the data manually or periodically. Thus, these services cannot effectively support customers to shop online under a dynamic e-commerce environment.

AddAll [5] provides price comparison service based on instant catalog searching and gathering upon customer's requests. Thus, it can provide fresh information about an item. However, it has a serious problem: the response time may be very long because it has to visit many shopping sites in an on-demand fashion. In addition, it issues multiple Web requests per each customer request and thus may cause a heavy traffic on the Internet.

Recently, several researches have proposed techniques for dynamic data caching [13, 15, 16, 17, 18, 19, 21, 22].

[4] The CMT is a table that contains all description for catalog conversion. Catalog conversion is necessary because the catalog formats of a merchant and the CIGMA are different from each other. Before the CIGMA service begins, the catalog format information is collected from the merchant and the CIGMA. Then, both parties cooperate to refine the table. Once the

Although consistency mechanism has not been described explicitly in [13, 20, 21, 22], cache consistency is an important issue in dynamic data caching. Generally, approaches for maintaining cache consistency can be classified as client-pull and server-push approaches.

Client-pull approaches: In client-pull approaches, cache is responsible for maintaining the consistency. Cache checks a data modification in a server to revalidate or update cached data without the knowledge of update. Most current Web proxy caches use the *If-Modified-Since* request header to revalidate cached objects [6]. This method may incur much unnecessary traffic during the revalidation. The mechanism proposed in [18] maintains the consistency of a cached content by setting a TTL field for different CGIs. This method is acceptable because the paper focuses on the digital library applications, where most contents are not changed frequently.

Server-push approaches: In server-push approaches, a server notifies caches of a modification for the invalidation or update of the cached data. Server-push approaches can relatively well maintain the consistency. However, they incur heavy load on a server in the case of Web proxy caching because there could be a large number of proxy caches that a server should notify of a modification. Thus, server-push approaches are usually used in caching for a specific server: reverse proxy caching [15, 16, 23] or database caching [19]. In [15, 16, 23], dynamically generated HTML pages or fragments are cached. These researches focus on how to identify the dependency between the modified underlying data and cached objects. Invalidator module proposed in [15] uses DB log files or periodic polling to detect a modification of data in a database and invalidate related objects. DBCache proposed in [19] uses a tool for asynchronous data replication. In this approach, an administrator subscribes replication requests or specifies the frequency of update propagation. Contrary to these two approaches, a modification of data can be detected automatically and there is no unnecessary overhead for periodic polling in our real-time update scheme. An algorithm for update propagation using a dependency graph was proposed in [16]. It is similar to our system that it uses the trigger mechanism of a DBMS. However, as mentioned above, that approach is used in the HTML fragments caching for a specific server.

The CIGMA is close to proxy caching because the CIGMA caches data from multiple merchant servers. But, the CIGMA can also be considered as reverse proxy caching for a specific server because the CIGMA serves for each merchant server individually. Thus, our real-time update scheme is based on server-push approach for the consistency without incurring a server's heavy load to notify of a modification. In addition, our approach provides a mechanism that can detect the status of each server for more consistently caching in an exceptional situation such as network partition or server failure. Our approach can also support heterogeneous servers by providing a universally deployable wrapper module.

5. Conclusions

Despite many advantages of the EC, online shopping is an overloaded activity for most ordinary customers due

CIGMA service is started, the conversion is automatically processed through the MW.

to a highly dynamic e-commerce environment. For instance, they are not sure if a chosen sales condition is really a good one even after exhaustive Web surfing. With its real-time update scheme, the CIGMA service can effectively support customers to do online shopping under a highly dynamic e-commerce environment. It suggests the best sales conditions over multiple shopping sites and provides a convenient shopping environment for customers. Thus, customers can save time and money by using the service. The CIGMA also helps merchants increase their business profit because the CIGMA can act as a sales agency.

References

[1] http://www.nextag.com/, NexTag - The Search Engine for Shoppers.

[2] http://www.mysimon.com/, mySimon - Compare products and prices from around the Web.

[3] http://www.dealtime.com/, DealTime - compare products, prices & stores.

[4] http://www.pricescan.com/, PriceSCAN - Comparison Shop for the Lowest Prices.

[5] http://www.addall.com/, AddALL - Book Search and Price Comparison.

[6] Balachaner Krishnamurthy and Jennifer Rexford. (2001) Web Protocols and Practice - HTTP/1.1, Networking Protocols, Caching, and Traffic Measurement, Addison-Wesley.

[7] Pei Cao and Chengjie Liu. (1998) “Maintaining strong cache consistency in the World Wide Web”, *Computers, IEEE Transactions on*, 47(4), pp. 445 -457.

[8] http://otn.oracle.com/doc/oracle8i_816/appdev.816/a76939/adg13trg.htm#376, Oracle8i Application Developer's Guide - Fundamentals Release 2 (8.1.6).

[9] http://www-4.ibm.com/software/data/db2/udb/ad/v7/adg/db2a0/frame3.htm#trigger, DB2 Application Development Guide: Using Triggers in an Actie DBMS.

[10] http://my.sybase.com/detail?id=1355, Sybase Manual: Triggers.

[11] http://www.mysql.com/doc/A/N/ANSI_diff_Triggers.html, MySQL manual: 1.7.4.4. Stored Procedures and Triggers.

[12] http://www.postgresql.org/idocs/index.php?plpgsql-trigger.html, PostgreSQL 7.1 Documentation: Chapter 24. PL/pqSQL-SQL Procedural Language.

[13] A. Brown, G. Kar, and A. Keller. (2001) “An active approach to characterizing dynamic dependencies for problem determination in a distributed environment”, *In Proceedings of IFIP/IEEE International Symposium on Integrated Network Management*, pp. 377 - 390. Seattle, WA.

[14] Luis M. B. Cabral. (2001), Introduction to Industrial Organization, 1st edition, MIT Press.

[15] K. S. Candan, Wen Syan Li, Qiong Luo, Wang Pin Hsiung, and D. Agrawal. (2001) “Enabling Dynamic Content Caching for Database-Driven Web Sites”, *In Proceedings of SIGMOD 2001*, Santa Barbara, CA.

[16] Jim Challenger, Arun Iyengar, and Paul Dantzig. (1999) “A Scalable System for Consistently Caching Dynamic Web Data”, *In Proceedings of INFOCOM 1999*, pp. 294-303. New York, NY.

[17] Jim Challenger, Arun Iyengar, Karen Witting, Cameron Ferstat, and Paul Reed. (2000) “A publishing system for efficiently creating dynamic web content”, *In Proceedings of INFOCOM 2000*, pp. 844-853, Tel Aviv, Israel.

[18] Vegard Holmedahl, Ben Smith, and Tao Yang. (1998) “Cooperative caching of dynamic content on a

distributed web server", *In Proceedings of the 7th IEEE International Symposium on High Performance Distributed Computing*, pp. 243-250, Chicago, IL.

[19] Qiong Luo, Sailesh Krishnamurthy, C. Mohan, Hamid Pirahesh, Honguk Woo, Brude G. Lindsay, and Jeffrey F. Naughton. (2002) "Middle-tier database caching for e-business", *In Proceedings of SIGMOD 2002,* Madison, Wisconsin.

[20] Qiong Luo and Jeffrey F. Naughton. (2001) "Form-based proxy caching for database-backed web sites", *In Proceedings of the International Conference on VLDB*, pp. 191-200, Roma, Italy.

[21] Qiong Luo, Jeffrey F. Naughton, Rajasekar Krishnamurthy, Pei Cao, and Yunrui Li. (2000) "Active Query Caching for Database Web Servers", *In Proceedings of WebDB 2000*, pp. 29-34, Dallas, Texas.

[22] Khaled Yagoub, Daniela Florescu, Cezar Cristian Andrei, and Valerie Issarny. (2000) "Building and Customizing Data-Intensive Web Sites Using Weave", *In Proceedings of the International Conference on VLDB*, Cairo, Egypt.

[23] Huican Zhu and Tao Yang (2001) "Class-based cache management for dynamic web content", *In Proceedings of INFOCOM 2001*, Anchorage, Alaska.

[24] S. M. Kim, S. Kang, H. K. Lee, and J. Song. "CIGMA: aCtive Inventory service in Global e-Market for enabling one-stop shopping over internet shopping sites" Technical Report CS-TR-2002-178, Korea Advanced Institute of Science and Technology (KAIST), 2002.

[25] S. M. Kim, S. Kang, H. K. Lee, and J. Song. "CIGMA: aCtive Inventory service in Global e-Market for enabling one-stop shopping over internet shopping sites" To appear in the 3rd International Symposium on Electronic Commerce (ISEC-02).