DyDa : Data Warehouse Maintenance in Fully Concurrent Environments*

Jun Chen, Xin Zhang, Songting Chen, Andreas Koeller, and Elke A. Rundensteiner

Department of Computer Science, Worcester Polytechnic Institute
Worcester, MA 01609-2280
{junchen, xinz, chenst, koeller, rundenst}@cs.wpi.edu

Introduction. Modern data integration environments integrate heterogeneous and autonomous information sources (ISs), so that those sources may change without being controlled from a higher data integration layer. In such an environment, source changes from different ISs can occur concurrently, and both data and schema changes are possible. We distinguish between three types of tasks of data warehouse (DW) maintenance in dynamic environments. View maintenance (VM) incrementally maintains the view extents under data updates, assuming the views have been materialized at the DW. View synchronization (VS) evolves the view definition kept at the DW when a source schema has been changed. View adaptation (VA) adapts the view extent after the view definition has been modified.

In this demonstration, we will display our *DyDa* (Dynamic Data Warehouse) framework, the first system capable of addressing such concurrency problems by integrating the VS, VM and VA modules. We will demonstrate two solution approaches within the DyDa family that both handle mixed sequences of concurrent data updates and schema changes. They differ however in the DW's responsibility of concurrency handling, which leads to different trade-offs on the dimensions of overall system complexity, performance and storage overhead.

***DyDa* System Overview.** Figure 1 (a) depicts the architecture of *DyDa with Light Wrappers* [2]. It is an enhanced DW architecture with additional functionality to handle concurrency problems in the middle layer. The *update message queue* module there is used to buffer all the update messages, and the *assign time stamp* module will assign an unique ID for each incoming messages.

Figure 1 (b) depicts the architecture of *DyDa with Smart Wrappers* [1]. The main difference to the previous approach

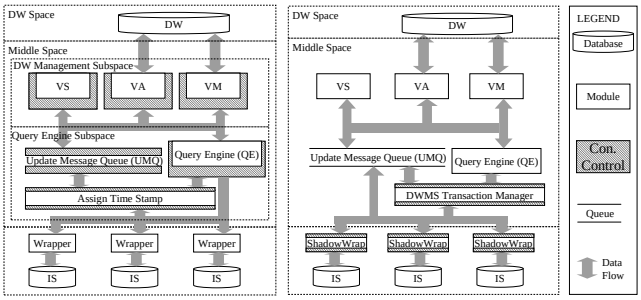

Figure 1: Architecture of DyDa with (a) Light Wrappers or (b) Smart Wrappers.

is that we move the responsibility of concurrency handling from the middle space to the wrappers. In particular, we propose a multiversion concurrency control algoithm of wrapper materialization to ensure a consistent view of ISs from the DW. We call such a wrapper ShadowWrapper. A DWMS Transaction Manager module is added in order to communicate with these ShadowWrappers. As a consequence, the concurrency handling at middle layer is no longer needed, and the Assign Time Stamp module for the detection of concurrent updates can be removed.

Trade-off of Two Approaches. *DyDa with Light Wrappers* requires no extra storage and thus will perform well at a low load with no concurrency. But it leads to complex VS, VM, and VA algorithms. Also, the system performance may decrease under a high load, because the maintenance query may repeatedly be broken. On the other hand, *DyDa with Smart Wrappers* performs elegantly under a high load since maintenance queries will never be aborted. This approach is based on sound transaction theory, and parallel processing and recovery are possible. Extra storage at the wrappers is required to store versions of ISs.

1. REFERENCES

[1] J. Chen and E. A. Rundensteiner. A Transactional Approach for Dynamic Data Warehouse Maintenance. Technical report, Worcester Polytechnic Institute, Dept. of Computer Science, November 2000.

[2] X. Zhang and E. A. Rundensteiner. DyDa: Dynamic Data Warehouse Maintenance in a Fully Concurrent Environment. In *Data Warehousing and Knowledge Discovery, Proceedings*, pages 94–103, September 2000.

*This work was supported in part by the NSF NYI grant #IIS 9796264, the NSF CISE Instrumentation grant #IIS 97–29878, the NSF grant #IIS 9988776, IBM, and Verizon Laboratories Inc.

SIGMOD 2001 May 21-24, Santa Barbara, California USA