제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

클라이언트 기반 웹하우스 유지 전략

이혁민*, 김경창*
*홍익대학교 전자계산학과

e-mail: greatmin@empal.com
kckim@cs.hongik.ac.kr

A Strategy for Maintaining Client-based Web House

Hyuk-Min Lee*, Kyung-Chang Kim*
*Dept of Computer Science, Hong-Ik University

요 약

본 논문에서는 기존의 서버 기반 데이터 웨어하우스 유지 전략에 대한 문제점들을 해결하기 위해 클라이언트 기반 웹 하우스 유지 전략을 제시한다.

소스 시스템에서 데이터 갱신이 발생했을 경우 브라우져 모니터를 통해 자동적으로 웹하우스에 실시간 반영하도록 하여 모든 사용자 요청을 서버에서 처리하는 부담을 줄이고 사용자가 많더라도 웹하우스 시스템의 처리부담을 최소화할 수 있도록 클라이언트에서는 결과 데이타 재사용/질의 재생성을 사용하여 서버의 자원 사용을 최소화할 수 있도록 하였다.

모든 클라이언트 프로그램은 자동적으로 설치되고 관리되므로 프로그램이 변동되더라고 쉽게 유지될 수 있으며 소스 시스템에는 어떠한 처리 부담도, 어떠한 프로그램도 설치하지 않기 때문에 실제 비즈니스 현실에서 적용하기가 용이하다.

1. 서 론

오늘날의 비즈니스는 고객 관계 중심의 마케팅인 CRM/eCRM, 일대일 마케팅을 실현할 수 있는 최적의 환경과 XML, SOAP 과 같은 최신 웹 기술을 요구하고 있다. 이러한 웹 기술의 발달로 기존의 클라이언트/서버 기반의 시스템들이 웹 기반 운영 시스템으로 바꾸어 가고 있어 이에 따른 모든 정보계 시스템들 또한 웹과의 연동이 불가피해 졌다. 따라서 대표적인 정보계 시스템인 데이터 웨어하우스가 웹과 연동할 것을 요구하는 것은 너무나도 당연하게 생각된다.

본 논문에서는 웹 환경에 적합한 새로운 데이터 웨어하우스 유지 전략을 제시함으로써 기존 운영 시스템의 부담을 최소화하였으며 소스 데이터 변경시 실시간으로 웹하우스에 반영되어 항상 최신의 데이터을 이용할 수 있도록 한다. 2장에서는 웹하우스의 기본 개념과 기존 데이터 웨어하우스의 일관성 유지 기법에 대해 설명하고, 3장에서는 웹하우스 일관성 유지 전략과 회복 기법에 대해 설명하고 결론을 내린다.

2. 관련 연구

본 장에서는 웹 환경에서의 데이터 웨어하우스인 웹하우스에 대한 개념을 간단히 설명한다. 또한 기존에 연구되었던 데이터 웨어하우스의 대표적인 서버기반 일관성 유지 기법인 ECA와, 웹과 웨어하우스를 접목시킨 DynaMat 시스템을 설명한다.

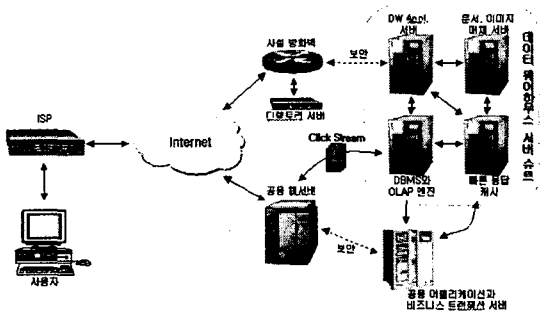
2.1 웹하우스

웹하우스[8]는 사용자가 웹 브라우져를 사용하여 데이터 웨어하우스에 접근하여 실시간으로 데이터를 분석할 수 있는 데이터베이스 시스템이다. 따라서 기존 데이터 웨어하우스의 모든 서비스는 웹 인터페이스를 이용하여 새롭게 개발 되어져야 한다.

그림 1은 가장 일반적인 웹하우스 시스템의 구성

도를 표현한 것이다.

데이터 웨어하우스를 웹에 가져온다는 것은 모든 사용자의 행동을 데이터 웨어하우스에 저장할 수 있다는 뜻이 된다. 웹은 우리에게 이전에 없던 새로운 데이터 소스, 즉 클릭스트림 데이터를 제공함으로써 사용자의 모든 행동과 의도를 파악할 수 있게 되었다. 클릭스트림을 통해 사용자의 미세한 행동을 시간에 따라 분석할 수 있으며, 어떤 것에 관심을 가지고 있는지 알 수 있어 사용자의 의도를 정확히 파악할 수 있게 되었다.



[그림 1] 웹하우스 일반적인 아키텍쳐

2.2 기존 데이터 웨어하우스의 일관성 유지 기법

웹하우스 환경에서는 소스 데이터의 갱신이 발생시, 갱신 데이터를 신속하고 정확하게 반영하는 것이 무엇보다 중요하며, 현재 이와 같은 연구는 다양한 방식으로 상당히 많은 진전이 있었다.[1,2,3,5,7]

2.2.1 ECA(Eager Compensating Algorithm)

ECA는 소스 시스템에서 갱신 발생시 데이터 웨어하우스의 실체화 뷰(MV)에 반영이 안되어 소스 시스템과 데이터 웨어하우스 사이에 데이터 불일치 현상이 발생하는 문제점을 해결하기 위한 대표적인 서버기반 보상질의 알고리즘[6]으로 소스 시스템에서 갱신 발생시 데이터 웨어하우스에 갱신을 반영하기 위해서 갱신 질의를 데이터 웨어하우스에 보내면, 데이터 웨어하우스는 해당 변경 데이터의 실체화 뷰와 관련 있는 데이터를 알 수 없기 때문에 보상질의를 작성하여 소스 시스템에 보상 질의함으로써 그 결과를 수집하는 방식이다.

사용자가 많아지면 복잡한 보상질의를 처리해야 하기 때문에 소스 시스템에 상당한 부담을 가져올 수 있으며, 하나의 갱신 질의는 데이터 웨어하우스에 반영을 위한 여러 개의 서브 보상질의가 생성되므로 완료되는 동안까지의 공백현상이 생기게 된다.

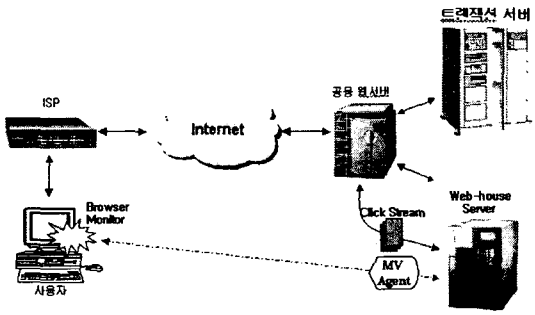
2.2.2 DynaMat 시스템

동적 페이지를 생성하는 웹 환경과 데이터 웨어하우스를 접목시켜 실시간으로 실체화 뷰를 효율적으로 관리하기 위해 제안된 시스템[4]으로 뷰 풀(View Pool)을 사용하여 이전에 수행되었던 결과 데이터를 저장하고 사용자가 질의시, 이전에 사용된 결과 데이터를 재사용할 수 있는지를 검사하여 성능을 향상시켰다. 하지만, 갱신 질의가 요청되면 실시간으로 데이터 웨어하우스에 반영되는 것이 아니라 주기적으로 오프라인 모드에서 전환되어 갱신 질의의 일괄처리 후, 다시 온라인 모드로 전환해야 하는 단점이 있다.

3. 웹하우스 유지 전략

본 연구는 기존 데이터 웨어하우스의 갱신에 대한 일관성 문제와 성능 향상을 고려한 최적의 시스템을 제안한다. 그림 2는 제안 시스템의 전체 구성도를 표현한 것으로 클라이언트에서 어떠한 갱신 질의가 발생되면 자동으로 감지하여 웹하우스에 즉시 반영하는 과정을 표현한 것이다.

소스 시스템에 어떠한 부담도 주지 않으면서도 데이터 갱신 질의시 자동적으로 클라이언트에서 실시간으로 웹하우스 시스템에 반영되도록 함으로써 사용자는 항상 최신의 데이터로 효과적인 분석을 수행할 수 있다.



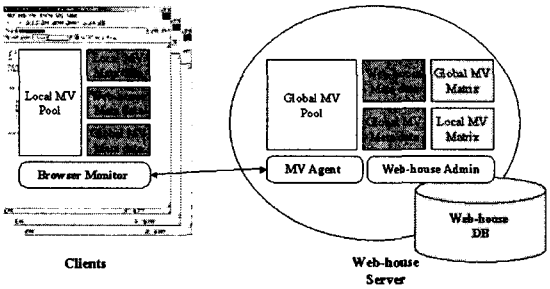
[그림 2] 제안 시스템 구성도

3.1 시스템 구조

그림 3은 제안 시스템의 각 필요 모듈의 구성을 도식화한 것이다.

사용자가 웹 사이트에 접근하게 되면 자동적으로 클라이언트 프로그램(Browser Monitor)이 설치된다.

이 브라우져 모니터는 사용자의 클릭스트림 정보를 웹하우스에 자동적으로 저장할 수 있을 뿐만 아니라 갱신 질의에 대해서는 변경할 데이터를 웹하우스에 자동적으로 반영할 수 있는 변경질의를 생성하며, 분석(OLAP) 질의에 대해서는 결과 재사용/질의 재생성을 이용하여 질의를 최적화 시킨다. 로컬 실체화 뷰 풀(Local MV Pool)은 이전에 처리되었던 결과 데이터를 저장하고, 나중에 재사용할 수 있도록 관리된다.



[그림 3] 제안 시스템의 모듈 구성도

실체화 뷰 에이전트(MV Agent)는 브라우져 모니터가 보낸 클릭스트림 데이터를 실시간 저장하거나 변경질의를 수행할 뿐만 아니라 클라이언트 장애(Fail)시 회복 기법을 수행할 수 있다.

3.2 웹하우스 유지 프로세스

사용자가 요청하는 질의의 성격에 따라 브라우져 모니터에 의해서 자동적으로 처리되는 과정을 크게 2가지 경우로 나누어 설명하고자 한다.

3.2.1 소스 시스템에 갱신질의 요청시

사용자가 요청한 갱신 질의가 소스 시스템에 요청되는 경우, 브라우져 모니터에 의해서 웹하우스는 자동적으로 갱신된다.

① 사용자가 요청한 갱신 질의를 생성하여 웹 서버를 통해 소스 시스템에 질의를 요청한다.

② 이때, 브라우져 모니터는 로컬에 있는 웹하우스 메타데이터를 사용하여 웹하우스에 반영할 변경질의를 생성하고, MV Agent에게 전송하여 질의를 수행한다.

③ 소스 시스템은 요청한 결과 데이터를 사용자에게 전송하고, 이때 브라우져 모니터는 "질의 성공" 이벤트를 캡쳐한다.

④ 질의가 성공되면, 브라우져 모니터는 웹하우스의 변경질의를 "Commit"함으로써 웹하우스에 변경 데이터를 반영한다.

3.2.2 웹하우스 시스템에 분석질의 요청시

사용자가 요청한 분석 질의를 웹하우스 시스템에서 처리되는 과정이다.

① 사용자가 분석 질의를 생성하여 웹 서버를 통해 웹하우스 시스템에 질의를 요청한다.

② 이때, 브라우져 모니터는 자신의 로컬에 있는 결과 데이터(Local MV)에 존재하는지 검사한다. 만약 존재한다면 결과 데이터를 사용할 수 있도록 질의를 작성한다.

③ 만약 존재하지 않는다면 웹하우스에 존재하는 결과 데이터(Global MV)에 존재하는지 검사한다. 만약 존재한다면 웹하우스 실체화 뷰(Global MV)를 사용할 수 있도록 질의를 생성한다.

④ 만약 웹하우스 시스템의 실체화 뷰에 존재하지 않는다면, 웹하우스에서 질의를 수행한 후, 결과를 클라이언트에게 전송한다.

3.3 다중 클라이언트 환경에서의 회복 기법

단일 클라이언트 환경과 다르게 여러 브라우져가 접근하는 웹 하우스 환경에서는 같은 데이터에 대한 갱신 질의의 순서를 어떻게 보장할 것인가가 가장 큰 이슈라고 할 수 있다. 이를 위해 제안시스템에서는 다음과 같은 방식을 사용한다.

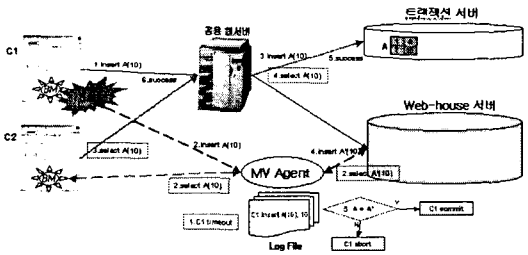
① 사용자가 갱신질의를 요청하여 결과를 받을 때, 브라우져 모니터는 "트랜잭션 종료시간"을 웹서버에서 받는다.

② 브라우져 모니터에서 트랜잭션의 상태(성공/실패)를 파악한 후, MV Agent에게 "Commit/Abort + 트랜잭션 종료시간"전송한다.

③ MV Agent에서는 같은 데이터에 대해서 다수의 변경 질의가 존재할 경우 트랜잭션 종료시간을 비교하여 웹하우스에 반영할 질의 순서를 결정한다.

3.3.1 클라이언트 장애시 회복 과정

그림 4는 클라이언트가 일시적으로 장애가 발생할 경우, 회복하는 과정을 표현한 것으로 일정시간이 경과 되었으나 반응하지 않은 클라이언트에 대해서도 MV Agent에서 현재 클라이언트의 장애를 파악한다. 만약 장애 발생시 현재 연결되어 있는 다른 클라이언트가 존재하는지 검사하여 없다면, "요청받은 질의"를 바로 삭제한다.



[그림 4] 클라이언트 장애시 회복 과정

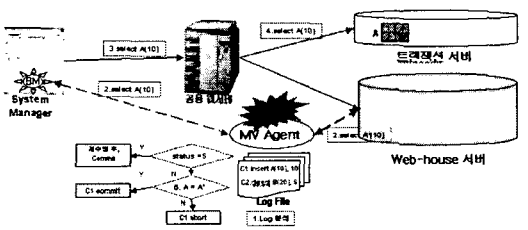
만약 다른 클라이언트들이 접속되어 있다면, 이 클라이언트를 이용하여 장애가 발생한 클라이언트로부터 받은 질의를 웹 서버에 요청한 후, 웹하우스의 결과와 비교하여 일치하면 웹하우스에 반영하고, 그렇지 않으면 삭제한다(Abort).

3.3.2 MV Agent 장애시 회복 과정

그림 5는 MV Agent가 장애가 발생할 경우에 시스템 복구 과정을 표현한 것이다.

MV Agent가 회복되면서, 현재 자신의 로그 정보를 분석하여 레코드 상태(status)가 "success"라면 소스 시스템에서의 갱신이 성공한 상태이므로 웹하우스에 해당질의를 재수행하고 Commit 한다.

하지만, 질의를 요청한 상태라면 레코드 상태가 존재하지 않으므로 웹하우스에 해당질의를 재수행하고, 또한 해당 요청 질의를 트랜잭션 서버에도 요청하여 결과값을 비교한다. 결과값이 일치하면 웹하우스에 반영하고 그렇지 않다면 해당질의를 삭제함으로 회복한다.



[그림 5] MV Agent 장애시 회복 과정

4. 결론

본 논문은 기존의 서버 기반 실체화 뷰에 대한 일관성 문제점과 소스 시스템의 변경이라는 현실적으로 불가능한 상황을 웹 기술과 클라이언트의 분담처리로 해결하기 위한 시스템을 제안하였다.

클라이언트 기반 웹하우스는 소스 시스템의 데이터 갱신이 발생했을 경우 자동적으로 웹하우스에 실시간 반영하도록 하였으며, 서버에서 모든 질의 요청을 처리하는 부담을 줄이기 위해 클라이언트 기반으로 구성하여 사용자가 많더라도 웹하우스 시스템의 처리부담을 최소화할 수 있도록 하였다. 모든 클라이언트가 자동적으로 설치되고 유지되므로 클라이언트 프로그램이 변동되더라도 쉽게 관리될 수 있다. 또한 소스 시스템에는 어떠한 처리 부담도, 어떠한 프로그램도 설치하지 않기 때문에 실제 현실에서 적용하기가 용이하다.

추후에는 현재 이슈가 되고 있는 eCRM 시스템으로 확장할 수 있도록 질의 추천 시스템을 제공하고자 한다.

참 고 문 헌

[1]
Q. Luo, J. F. Naughton, R. Krishnamurthy, P. Cao, and Y. Li. "Active Query caching for database Web servers",in Proc. of the Int. Workshop on Web and Databases (WebDB), 2000

[2] A. Labrinidis, N. Roussopoulos, "WebView Materialization", In Proc. of the ACM SIGMOD Conference, Dallas, May 2000

[3] Alexandros Labrinidis, Nick Roussopoulos, "On the Materialization of Webviews", ACM SIGMOD Workshop on the Web and Databases, June 1999

[4] Y. Kotidis, N. Roussopoulos, "DynaMat: A Dynamic View Management system of Data Warehouses", In Proc. of the ACM SIGMOD Conference, Philadelphia, 1999

[5] Budzik, J., Bradshaw, S., Fu, X., and Hammond, K. J.,"Watson : Anticipating and Contextualizing Information Needs", In Proc. of the 62 Annual Meeting of the American Society for IS., 1999.

[6] Y.Zhuge, H. Garcia-Molina, Janet L. Wiener,"The Strobe Algorithms for Multi-Source Warehouse Consistency", In proc. the International Conference on Parallel and Distributed Information system, 1997

[7] Elke A. Rundensteiner, "DyDa : DW Maintenance in fully concurrent Environments", TR-99-20, Worcester Polytechnic Institute, Dept. of CS. 2001

[8] Ralph Kimball, Richard Merz, "The Data Webhouse Toolkit", John Wiley & Sons, 2000.