제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

미디에이터 시스템에서 커서 기반의 통합 결과 접근 방법

이명철, 김영균, 이미영
한국전자통신연구원 컴퓨터소프트웨어연구소 컴퓨터시스템연구부
e-mail : mclee@etri.re.kr

A Cursor-based Approach to Access of Integrated Results in Mediator Systems

Myungcheol Lee, Young-Kyoon Kim, Mi Young Lee
Computer System Department, CSRL, ETRI

요 약

미디에이터/랩퍼 구조를 기반으로 하는 정보 통합 시스템에서 모든 통합 결과를 한꺼번에 미디에이터 클라이언트로 보낸다면, 클라이언트에 큰 문제를 초래할 수 있다. 만일 사용자가 정밀하게 질의를 하지 않는다면, 서버는 많은 관련이 없는 결과를 만들어 낼 것이다. 그리고, 웹 사용자들은 관련된 결과라 할지라도 모든 결과를 접근하지는 않는 경향이 있다. 특히, 클라이언트가 제한된 하드웨어와 네트웍 자원을 가진 모바일 단말기 상에서 동작하는 경우, 문제가 더욱 심각하다. 본 논문에서는, 관계형 DBMS 의 커서 개념을 채택하여 미디에이터 시스템에서 통합 결과를 클라이언트에 부분적으로 전송하는 새로운 방식을 제안한다.

1. 서론

최근, XML 기반의 미디에이터/랩퍼 구조를 통하여 인터넷에 분산된 이질적이고 다양한 정보 자원을 통합하기 위한 많은 연구가 수행되고 있다[1,2]. 대부분의 연구에서는, 미디에이터 측에서 최적화된 질의 실행 계획을 생성하고 수행함으로써 정보 자원으로부터 미디에이터로 전달되는 결과를 줄이는 방식으로 검색 성능을 향상하고자 한다.

그러나, 최적화된 질의 처리 이후에도 여전히 통합 결과가 너무 많을 수도 있다. 그리고, Ludäscher[2]가 지적한 바와 같이, 웹 환경의 사용자들은 일반적으로 꼭 필요한 결과만을 얻을 수 있을 만큼 정밀하게 질의를 제시하지 않는 경향이 있다. 대신에 비교적 덜 정밀한 질의를 던지고, 겨우 몇 개의 결과만을 살펴보고, 의도한 결과를 찾았던지 아니면 결과들이 관련이 없어 보이기 때문에 멈추게 된다. 따라서, 통합 결과 전체를 한꺼번에 사용자에게 보내는 것보다는 사용자가 필요로 하는 통합 결과만을 생성해서 반환하는 것이 바람직할 것이다. 게다가 제한된 하드웨어와 네트웍 자원을 가진 모바일 환경의 사용자라면, 이런 기능은 필수적이다.

현재까지, 이와 같은 문제점을 해결하기 위한 두 개의 시도가 있었다. Ludäscher 는 가상 중개 뷰(virtual mediated view)의 평가가 항해(navigation)에 의해 구동되는 DOM-VXD 라는 프레임워크를 제안했다. DOM-VXD 방식에서는 미디에이터가 클라이언트의 질의에 대해 가상의 결과 문서를 반환하고 사용자가 DOM API 의 부분 집합을 이용하여 가상의 결과 문서를 항해함에 따라 결과 문서를 만들어 내게 된다. 그리고, Sae-Tung[3]은 모바일 사용자를 위한 통합 방식으로서 항해를 통한 통합 방식을 제안했는데, WWW 의 하이퍼링크를 기반으로 각 웹 정보 자원의 데이터베이스 호출을 이용한다.

본 논문에서는, 관계형 DBMS 에서의 커서 개념을 채택하여 미디에이터 시스템에서 통합 결과를 접근하는 새로운 방식을 제안한다. 본 방식은 필요 시에 통합 결과의 일부 만을 클라이언트로 전송하는 것을 가능하게 한다.

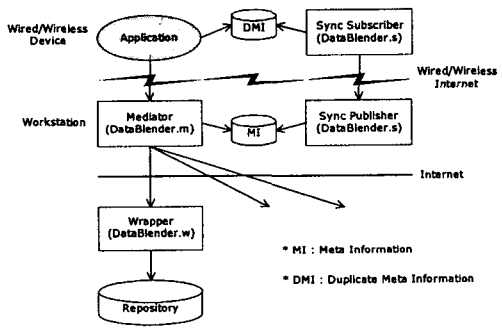
본 논문의 구성은 다음과 같다. 먼저, 2 장에서 본

논문에서 제안하는 기법이 적용된 DataBlender 시스템에 대해서 간략히 살펴 본다. 3 장에서는 통합 결과 관리자에 대해서 기술하고, 4 장에서는 결과 관리에 대해서 기술하고, 5 장에서는 커서 운행에 대해서 기술한다. 마지막으로 6 장에서는 본 논문의 결론을 맺는다.

2. DataBlender

DataBlender[4]는 미디에이터/랩퍼 구조 기반의 정보 통합 시스템이며, XML 을 공통 데이터 모델로서 사용하고, XQuery 를 미디에이터와 랩퍼를 위한 질의 언어로서 이용하고, XML Schema 를 통합/지역 스키마 명세를 위해, 그리고 XQuerySD[5]를 통합 스키마 정의 언어로서 이용하는 등, 최신의 XML 기술을 포함하고 있다. XQuerySD 는 통합 스키마 정의를 위해 XQuery 의 구문을 활용한다.

[그림 1]에서 보는 바와 같이, DataBlender 는 하나의 미디에이터 서버(DataBlender.m), 여러 개의 랩퍼(DataBlender.w), 그리고 스키마 동기화기(DataBlender.s)로 구성이 된다.



[그림 1] DataBlender 의 구조

1) DataBlender.m

미디에이터는 지역 스키마로부터 생성된 통합 스키마를 관리하며, 전역 질의를 처리한다. 미디에이터는 전역 질의를 각 랩퍼를 위한 지역 질의로 분해하고, 해당 랩퍼로 전달한다. 그리고, 최종적으로 각 랩퍼로부터의 결과를 통합하여 사용자에게 DOM 트리 형태로 통합 결과를 제공한다.

2) DataBlender.w

랩퍼는 미디에이터와 정보 자원(source)과의 연결을 담당한다. 랩퍼는 미디에이터 서버로부터의 질의를 정보 자원을 위한 질의 언어로 변환을 하고 정보 자원으로부터의 결과를 미디에이터 서버를 위한 XML 문서로 변환을 한다.

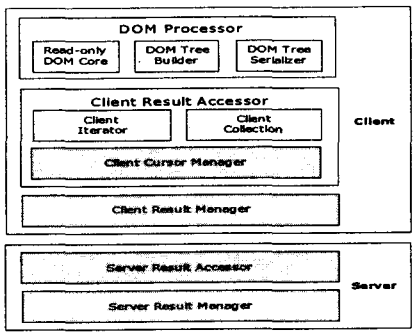
(3) DataBlender.s

스키마 동기화기는 미디에이터 서버로부터 메타 정보를 복제하고 서버와 클라이언트 간의 메타 정보의 일관성을 유지하는 역할을 한다

3. 통합 결과 관리자

DataBlender.m 은 크게 지역 스키마를 전역 스키마로 통합하고 통합된 전역 스키마를 관리하는 GSP(Global Schema Processor), 사용자 질의를 지역 질의로 분해하고 이를 수행하는 GQP(Global Query Processor), 질의 처리 후에 또는 질의가 처리되는 동안에 통합 결과를 관리하고 사용자에게 제공하는 IRM(Integrated Result Manager)으로 구성이 된다.

IRM 은 [그림 2]에서 보듯이, DOM 처리기, 결과 접근자, 그리고 결과 관리자 등으로 구성이 된다.



[그림 2] IRM 의 구성 모듈

3.1 DOM 처리기(DOM Processor)

DOM 처리기에서는 커서에 의해 제공되는 부분 결과 접근 연산을 이용하여 통합 결과를 처리하기 위한 DOM 레벨 1 코어 명세서의 읽기 전용 인터페이스를 제공한다. 그리고 DOM 처리기는 통합 결과로부터 DOM 트리를 구성하기 위한 인터페이스와 DOM 트리를 XML 문서로 직렬화하기 위한 인터페이스를 제공하는데, 이것들은 DOM 레벨 3 "Load and Save" 명세서에 정의되어 있다.

DOM 트리는 관계형 DBMS 에서의 뷰와 마찬가지로 실제 데이터를 갖지 않고 가상적으로 구성이 되며, 실제 데이터는 RMResultSet 내에 저장이 된다. RMResultSet 은 4.1 절에서 설명한다.

3.2 결과 접근자(Result Accessor)

통합 결과는 사용자가 반환되는 결과에 접근하는 방식의 유형에 따라 다음과 같이 분류할 수 있다.

1) 단일(Singleton) 결과: 통합 결과 셋 내에 하나의 결과 만이 존재하며, 클라이언트로 즉시 전송이 된다.

2) 컬렉션(Collection) 결과: 통합 결과 셋 내에 여러 개의 결과가 존재하며, 사용자는 결과 셋 전체를 대상으로 작업을 한다. 각 결과는 DOM 트리의 하나의 노드로서 취급이 되며, 사용자는 통합 결과 셋이 완전히 구성된 이후에야 결과에 접근하는 것이 가능하다.

3) 반복자(Iterator) 결과: 통합 결과 셋 내에 여러 개의 결과가 존재하며, 사용자는 각 결과를 별개의 대상으로 보고 작업을 한다. 각 결과는 반복자 메소드를 사용하여 접근하며, 사용자는 통합 결과 셋이 완전

히 구성되기 이전에 이미 구성된 결과 셋에 대한 접근이 가능하다.

IRM 에서는 서버와 클라이언트에서 결과 셋 위에 결과 접근자를 제공함으로써 컬렉션 결과 접근 방식과 반복자 결과 접근 방식을 지원한다.

3.3 결과 관리자 (Result Manager)

미디에이터 클라이언트가 미디에이터 서버에 연결하면, 하나의 통신 세션이 그 사이에 수립이 되며, 세션 내에서 클라이언트는 여러 개의 질의를 수행할 수 있다. 하나의 세션 내에서 처리된 모든 질의의 통합 결과는 세션이 종료되거나 사용자가 통합 결과의 접근자를 명백하게 삭제하기 전까지는 서버 내에서 유지가 된다.

그리고, 클라이언트에서도 역시 서버로부터 받은 통합 결과가 유지되어야 한다. RM(Result Manager)이 서버와 클라이언트에서 통합 결과를 관리하는 역할을 수행한다.

4. 결과 관리

결과 관리자는 결과를 유지하는 기능 이외에 효율적 결과 관리를 위해 결과 직렬화 기능을 제공하며 결과 접근자를 위해서는 커서 ID 부여 기능을 제공한다.

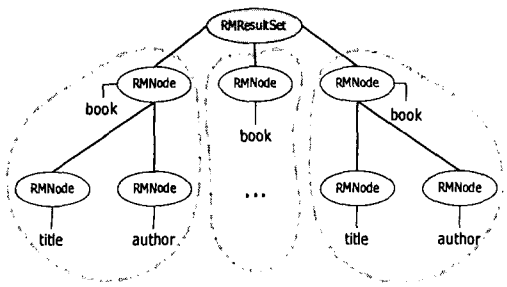
4.1 결과 구성

하나의 세션 내에서는 오직 하나의 RM 이 생성이 되며 세션 내의 모든 결과를 관리한다.

RMResultSet 은 질의의 통합 결과 셋 전체를 나타내며, RM 에 의해 관리가 된다. RMNode 는 통합 결과 셋 내의 결과 하나를 나타낸다.

예를 들어, 사용자가 아래와 같은 XQuery 질의 문을 수행하면, 질의 처리 후에는 다음 [그림 3]과 같은 통합 결과 셋이 미디에이터 서버 내에 구성이 된다.

```
FOR $b IN document("bib.xsd")//book
WHERE $b/@year >= 2001
RETURN
        <book>
                {$b/title,$b/author}
        </book>
```



[그림 3] 통합 결과 셋의 예

4.2 결과 직렬화

미디에이터가 클라이언트에 의해 많이 사용되면 될수록, 그만큼 많은 RMResultSet 이 미디에이터 서버에서 관리가 되게 되고, 미디에이터 서버가 운영되는 호스트에 그만큼의 메모리 측면에서의 부담을 주게 될 것이다.

따라서, 자주 사용이 되지 않는 RMResultSet 은 메모리 오버헤드를 경감시키기 위하여 메모리에서 파일로 직렬화가 된다. 이렇게 직렬화된 RMResultSet 들은 클라이언트에서 다시 접근을 요청하면 메모리로 올려 진다.

클라이언트에서도 같은 문제가 발생하는데, 모든 RMNode 가 동시에 사용이 되는 것이 아니므로, 자주 사용되지 않는 RMNode(즉, 현재 커서 위치에서 멀리 떨어진 위치에 있는)는 RM 에 의해 메모리에서 삭제가 된다. 이런 경우, RMNode 에 매핑된 DOM 객체 자체는 RMNode 삭제에 대한 투명성을 제공하기 위해 메모리에서 삭제되지 않는다. 삭제된 RMNode 가 나중에 다시 DOM 연산의 호출에 의해 필요하게 되면, 서버에 다시 요청하여 가져오게 된다.

4.3 커서 ID 부여

모든 통합 결과는 해당 세션 내에서 각 RM 에 의해서 관리가 되기 때문에, 커서 ID 는 각 RM 에 의해 부여가 되며, 그 세션 내에서만 유일하다.

커서 ID 는 1 에서부터 시작하며 CURSOR_ID_MAX 까지 1 씩 차례로 증가하며 부여가 된다. CURSOR_ID_MAX 는 미디에이터 서버가 하나의 세션 내에서 얼마나 많은 통합 결과를 처리할 수 있는 가를 의미한다. CURSOR_ID_MAX 는 시스템의 성능에 따라 가변적이기 때문에, 본 제안을 적용한 결과 관리 기에서는 CURSOR_ID_MAX 의 적정한 값을 제시하지 않는다. 대신에 하나의 세션 내의 허용 가능한 통합 결과의 최대 개수를 설정할 수 있도록 메소드를 제공한다.

5. 커서 운행

관계형 DBMS 에서, 커서는 프리패치(prefetch) 기법을 이용하여 구현이 된다. 프리패치는 요청된 결과보다 많은 결과를 클라이언트로 미리 전송하여 성능 향상을 꾀하고자 하는 것인데, 클라이언트에 의해 곧 사용될 가능성이 높은 결과 만을 전송하면 응답 시간을 줄일 수 있기 때문이다.

일단 결과가 클라이언트에 전송이 되면, 일반적으로 결과는 클라이언트 커서가 닫힐 때까지 클라이언트의 메모리 내에 머문다. 그러나 클라이언트에 메모리가 충분히 남아 있지 않다면, 그 중 일부는 결과 간의 공간적 지역성(spatial locality)을 고려하여 결과 관리자에 의해 삭제가 된다. 이 경우, 삭제된 결과는 클라이언트가 나중에 접근하려면, 서버로부터 재전송이 되어야 한다.

따라서, 클라이언트가 서버에 결과를 요청하는 유형은 다음 두 가지로 구분할 수 있다.

1. 프리패치를 적용하여 결과를 요청하는 경우
2. 하나의 문서만을 요청하는 경우

클라이언트는 아직 전송이 안 된 결과는 서버에게 프리패치를 적용하여 전송할 것을 요청한다.

그리고, 클라이언트가 삭제된 결과를 접근하고자 할 때는, 기본적으로 서버에게 삭제된 결과만을 전송할 것을 요청한다.

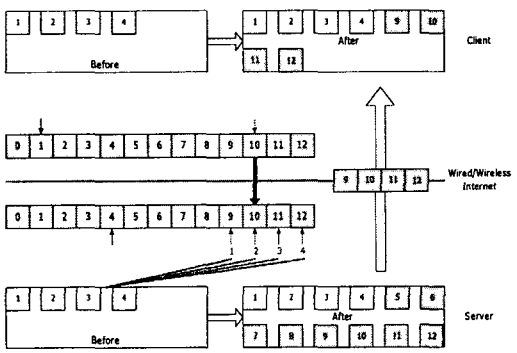
그러나, 클라이언트에서 아주 많은 결과가 삭제가 되었고, 결과적으로 클라이언트의 받은 결과 목록에 많은 연속된 구멍이 있다면, 접근하려는 결과 주변의 결과들도 삭제가 되었는지 살펴 보고, 삭제가 되었다면, 서버에게 결과 하나만을 전송하도록 요청하는 것보다는 프리패치를 적용하여 전송하도록 요청하는 것이 바람직할 것이다.

서버는 프리패치를 적용한 결과 전송을 요청 받으면, 결과의 전송 여부에 상관없이, 초기 커서 위치를 찾아 내서 단지 prefetch_size 만큼의 결과를 클라이언트에 전송한다. 아래의 초기 커서 위치 계산 방법을 사용하면 initial_cursor_position ~ initial_cursor_position + prefetch_size - 1 까지의 범위에서는 부분 전송이 있을 수 없다.

초기 커서 위치는 아래의 간단한 계산을 통해서 결정이 된다.

```
s = c*n + 1
& s <= c
& s > c - p
```

여기서, s 는 서버에서의 초기 커서 위치, c 는 클라이언트로부터의 입력 위치, p 는 프리패치 크기, 그리고 n 은 임의의 양의 정수이다.



[그림 4] 커서 동작 방식 예

[그림 4]는 커서가 동작하는 하나의 시나리오를 예로서 나타낸다. 이때, prefetch_size 는 4 라고 가정한다.

자세한 동작 방식은 아래와 같다:

1. 클라이언트 커서는 1번째를, 서버 커서는 4번째를 가리키고 있다. 클라이언트는 이미 4개의 결과를 받은 상태이고 서버는 현재까지 4개의 결과만을 가지고 있다.
2. 사용자가 10번째 결과를 접근하려고 하면, 클라이언트는 10번째 결과와 주변 결과를 갖고 있는 지 검사한다.
3. 갖고 있지 않으면, 클라이언트는 서버에게 10번째 결과를 프리패치를 적용하여 전송해 줄 것을 요청한다.
4. 서버는 위에서 설명한 초기 커서 위치 계산 방법으로 초기 커서 위치를 9번째 결과로 설정한다.
5. 서버는 아직 10번째 결과를 갖고 있지 않으므로, 랩퍼로부터 결과를 가져와서 통합하여 5번째 ~ 12번째 결과를 만들어 낸다.
6. 서버는 9번째 ~ 12번째까지 4개의 결과를 클라이언트에 전송한다.

6. 결론

본 논문에서는, XML 기반의 미디에이터 시스템에서 사용자의 필요에 따라 적절한 수의 결과만을 제공할 수 있는 통합 결과 접근 방식을 제안했다. 이를 위해, 관계형 DBMS 에서의 커서 개념을 각 결과를 접근하기 위한 수단으로서 적용하였고, 사용자의 결과 접근 시 응답 시간을 줄일 수 있도록 프리패치 기법을 적용하였다. 커서 연산 및 프리패치 기법 자체는 자체는 사용자에게는 DOM 인터페이스 아래에 감추었다.

향후 연구 과제로는, 구현이 완료된 후 성능을 측정하고 실제 응용에서의 사용을 위해 커서의 이동 및 프리패치의 동작 방식을 향상시키고자 한다.

참고문헌

[1] S. Cluet, C. Delobel, J. Siméon, and K. Smaga, "Your Mediators Need Data Conversion!," ACM SIGMOD, pp. 177-178, 1998

[2] B. Ludäscher, Y. Papakonstantinou, and P. Velikhov, "Navigation-Driven Evaluation of Virtual Mediated View," EDBT, pp. 150-165, 2000

[3] Wisut Sae-Tung, Tadashi Ohmori, Mamoru Hoshi, "An Integration System of Web Information Sources for Mobile Users," IDEAS, pp. 250-256, 2000

[4] Mi-Young Lee, et al., "Design of an XML-based Database Integration Middleware: DataBlender," ICIS, pp. 863-867, 2002

[5] 김병섭, 이미영, "통합 스키마 정의를 위한 XQuerySD", 한국정보과학회 데이터베이스연구회 학술대회, 18 권 2 호, 2002