모바일 환경에서 다수 클라이언트의 동시 동기화 시스템 설계 및 성능 평가

김홍기, 김동현
동서대학교

A Design and Performance Evaluation of Concurrent Synchronization System of Multiple Client on a Mobile Environments

Hong-Ki Kim, Dong-Hyun Kim
Dongseo University
E-mail : inthestream@nate.com, puserover@dongseo.ac.kr

요약

모바일 환경에서 기존의 양방향 동기화 시스템은 다수 클라이언트의 동기화를 순서대로 처리하였다. 이런 경우 처리시간이 긴 동기화 작업으로 인해 다른 클라이언트들의 동기화 작업이 오랜 시간 대기하는 문제가 있다. 이 논문에서 변경충돌이 없는 다수 클라이언트들간 동기화 작업을 다중 큐를 이용하여 동시에 수행하는 시스템을 설계·구현하였다. 또한, 동기화 작업 처리기간의 비율과 작업처리 패턴에 따른 순차적 동기화와 동시 동기화의 성능을 비교하였다.

ABSTRACT

The two-way synchronization system in mobile environment processes synchronization of multi clients by order. In this case, that a synchronization operation, which takes long time to process, makes other clients wait is a problem. In this paper, treats to design and materialize a system which processes synchronization operations between clients without update conflicts as using multi queues. Also, compares performance between the simultaneous synchronization and the synchronization in consecutive order by the rate of process term of synchronization operation and the pattern of operation process.

키워드

양방향 동기화, 모바일, 다중 큐

Ⅰ. 서 론

모바일 기반의 GIS 서비스는 최적의 서비스를 위해 최신의 공간 데이터를 신속하게 제공하는 방법들에 대한 연구가 진행되고 있다[1][2][3]. 최근에는 모바일 기기에서 공간 데이터의 변경 및 수집이 가능해지면서 모바일 기기를 이용하여 현장에서 변경된 공간 데이터를 변경 및 수집하여 무선 네트워크를 이용하여 서버와 동기화 하는 양방향 동기화 기법이 개발되었다[4]. 그러나 모바일 환경에서 양방향 동기화는 모바일 클라이언트가 서버와 단절 상태에서 변경하는 공간 데이터에 대한 처리와 다수 클라이언트로부터 동기에 요청되는 동기화 작업을 처리해야 한다. 기존의 동기화 시스템들은 보통 일대일 동기화를 지원하거나 다수 클라이언트에 대한 동기화를 지원하나 서버에서 클라이언트로의 단방향 동기화를 지원하였다. 최근 개발된 양방향 동기화 시스템은 다수 클라이언트에 대하여 순차적으로 동기화를 지원하고 있다.

대용량의 공간 데이터를 순차적 양방향 동기화는 다음과 같은 문제가 발생한다. 모바일 기기에서 변경되거나 수집되는 공간 데이터는 보통 저용량의 데이터들이다. 그러나 최적의 서비스를 위해 공간 데이터를 배포하는 서비스 제공자들은 대용량의 공간 데이터를 동기화할 수 있다. 만약 서비스 제공자가 대량의 공간 데이터에 대하여 동기화 작업을 처리중일 때 모바일 클라이언트의 저용량 동기화 작업들은 서비스 제공자의 동기화 작업이 끝날 때 까지 동기화 작업을 대기하고 있어야 한다.

이 논문에서는 다수 클라이언트의 동기화 작업

* 본 연구는 국토해양부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07국토정보C05)에 의해 수행 되었습니다.

에 대하여 변경충돌이 없는 클라이언트들에 대하여 다중 큐를 이용하여 동기화 작업을 동시에 처리할 수 있는 동시 동기화 시스템을 설계하고 구현하였다[5]. 또한, 구현된 동시 동기화 시스템과 순차적으로 동기화를 하는 시스템과의 성능을 비교하였다.

II. 관련연구

모바일 기반의 양방향 동기화 시스템은 대용량의 공간 데이터를 무선 네트워크를 이용하여 양방향으로 동기화하는 시스템이다. 이 시스템은 제한된 대역폭을 가진 무선 네트워크를 이용하여 대용량의 공간 데이터를 동기화 하기위해 부분 업데이트를 지원한다. 부분 업데이트는 동기화 시 모든 공간 데이터를 동기화 하는 것이 아니라 변경된 데이터만을 동기화 한다. 또한, 시공간 데이터를 계층 및 분할로 구분하여 부분적으로 동기화를 진행한다. 모바일 기반의 양방향 동기화 시스템은 각 분할 영역에 대하여 하나의 타임스탬프를 가진다. 서버의 분할들은 분할 안에 있는 객체들 중 마지막으로 갱신된 객체의 시간을 타임스탬프로 가진다. 모바일 클라이언트의 분할들은 서버와 마지막으로 동기화한 시간을 타임스탬프로 가진다. 이 타임스탬프는 동기화 시 모바일 클라이언트의 변경 데이터에 대한 변경충돌 검사에 이용된다. 또한 동기화 시 모바일 클라이언트가 서버로 업데이트해야 하는 데이터를 구분하는데 이용되며, 서버에서는 각 클라이언트로 업데이트 해야 하는 공간 데이터를 구분하는데 이용한다.

모바일 기반의 양방향 동기화 시스템은 공간 데이터의 양방향 동기화를 지원하는 동기화 프로토콜을 제안한다. 이 시스템에서 제안하는 동기화 프로토콜은 서버에서 클라이언트로만 업데이트가 필요한 경우, 클라이언트에서 서버로만 업데이트가 필요한 경우, 서버와 클라이언트 모두 업데이트가 필요한 경우에 대한 프로토콜을 제시한다.

그러나 이 시스템은 다수 클라이언트의 동시 동기화 작업에 대한 처리를 고려하지 않는다.

III. 다수 클라이언트의 동시 동기화 시스템

이 장에서 사용할 용어는 다음과 같다.

- 최종 갱신 시간(Last Update Time, LUT)
  - 서버 파티션의 최종갱신 시간
- 최종 동기화 시간(Last Sync Time, LST)
  - 클라이언트의 파티션이 서버와 최종으로 동기화한 시간
- DOS(P_i)
  - 서버에서 클라이언트 파티션 P_i로 갱신할 객체들의 집합
- TDOS(P_i)
  - 클라이언트에서 서버로 갱신할 파티션 P_i의 객체들의 집합
- CR(Copy Region)
  - 클라이언트가 서버로부터 전송 받은 시공간DB의 영역정보

3.1 변경충돌 검사

다수 클라이언트의 동시 동기화 시스템은 동시에 처리되는 모바일 클라이언트들의 변경 데이터들이 서로 변경충돌이 없는 경우 다중 큐에 삽입되어 동시에 동기화 작업을 진행한다. 다수 클라이언트의 변경충돌 검사는 표 1과 같이 두 가지 형태로 구성이 된다.

표 1 변경충돌 검사 표

대상	분할 단위	객체 단위	충돌
서버 vs 클라이언트	$LUT(P_i) \leqq LST(P_i)$		X
	$LUT(P_i) > LST(P_i)$	$DOS(P_i) \cap TDOS(P_i) = \varnothing$	X
		$DOS(P_i) \cap TDOS(P_i) \neq \varnothing$	O
클라이언트 vs 클라이언트		$TDOS(P_i) \cap TDOS(P_i) = \varnothing$	X
		$TDOS(P_i) \cap TDOS(P_i) \neq \varnothing$	O

모바일 클라이언트가 동기화를 요청하면 먼저 클라이언트와 서버의 변경충돌 검사를 한다. 서버와 변경충돌이 없으면 동기화를 요청한 클라이언트의 동기화 데이터를 다중 큐에 삽입한다. 동기화 데이터를 다중 큐에 삽입할 때 모든 큐가 비어 있으면 문제가 없지만 동기화 데이터가 들어있는 큐가 있으면 다중 큐에 삽입된 클라이언트와 변경충돌 검사를 한다. 클라이언트가 변경충돌 검사는 각 영역별로 동일한 객체가 있는지 비교를 하여 동일한 객체가 존재하면 변경충돌 발생이다. 클라이언트들 사이에 변경충돌이 발생하면 나중에 요청된 동기화 작업을 취소한다.

3.2 다중 큐

다중 큐는 다수의 모바일 클라이언트에서 요청되는 동기화 작업을 동시에 처리하기 위한 버퍼이다. 다중 큐는 리스트 형태의 자료구조를 가지며 구성된 노드는 두 가지 형태를 가진다. 첫 번째 형태는 동기화 데이터를 객체단위로 가지고 있는 쓰기 노드이다. 쓰기 노드는 동기화 작업 시 서버의 데이터베이스에 클라이언트의 객체를 업데이트 하는데 사용된다. 다른 하나의 형태는 읽기 노드이다. 읽기 노드는 동기화 작업 시 동기화를 요청한 클라이언트의 노드 리스트의 마지막에 위치하며 서버의 데이터베이스로부터 클라이언트가 업데이트를 받아야하는 동기화 데이터 요청하데 사용된다.

다중 큐는 둘 이상의 클라이언트에 대하여 변경충돌 검사 후 변경충돌이 없는 클라이언트의 동기화 데이터를 서로 다른 큐에 삽입하고 라운드 로빈 방식으로 병행 처리한다. 이 논문에서는 다중 큐의 수를 3개로 하였다. 표 2는 3개의 큐로 구성된 다중 큐의 삽입 방법을 보여준다.

표 2 다중 큐 삽입 방법

데이터가 있는 큐의 수	중복영역 검사	중복 발생 큐의 수	삽입방법
0		0	비어 있는 큐에 삽입
1	$CR(M_i) \cap CR(M_i) = \varnothing$	0	비어 있는 큐에 삽입
	$CR(M_i) \cap CR(M_i) \neq \varnothing$	1	중복이 발생한 큐에 삽입
2	$CR(M_i) \cap CR(M_i) = \varnothing$	0	비어 있는 큐에 삽입
	$CR(M_i) \cap CR(M_i) \neq \varnothing$	1	중복이 발생한 큐에 삽입
		2	노드의 수가 많은 큐에 삽입
3	$CR(M_i) \cap CR(M_i) = \varnothing$	0	노드의 수가 가장 작은 큐에 삽입
	$CR(M_i) \cap CR(M_i) \neq \varnothing$	1	중복이 발생한 큐에 삽입
		2	노드의 수가 많은 큐에 삽입
		3	노드의 수가 가장 많은 큐에 삽입

다중 큐의 삽입은 동시에 동기화하는 클라이언트들의 동기화 영역도 고려해야 한다. 동시 동기화를 진행하는 경우 쓰기 노드와 읽기 노드 작업의 처리 순서에 따라 문제가 발생할 수 있다.

3.3 다수 클라이언트 동기화 프로토콜

다수 클라이언트에 대한 동기화 프로토콜은 그림 1과 같다. 그림 1은 모바일 클라이언트 M_1과 M_2에 대한 동시 동기화 프로토콜을 보여준다.

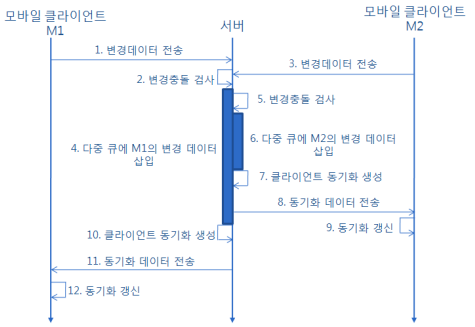

그림 1 모바일 클라이언트 M_1과 M_2의 동시 동기화 프로토콜

기본적으로 양방향 동기화 프로토콜은 동기화를 시작하면 모바일 클라이언트에 변경한 데이터를 서버로 전송한다. 서버는 전송받은 데이터의 변경충돌을 검사하고 변경충돌이 없으면 다중 큐에 삽입한다. 삽입된 데이터는 다중 큐에 의해 처리된다. 다중 큐 처리는 클라이언트의 변경 데이터를 서버 데이터베이스에 갱신하고 마지막에 서버에서 클라이언트로 동기화할 데이터를 생성한다. 서버는 다중 큐 처리 중 생성되는 DOS를 해당 클라이언트로 전송한다. DOS를 전송 받은 클라이언트는 모바일 데이터베이스에 DOS를 갱신하고 동기화가 완료된다. 다수 클라이언트의 동기화는 위의 과정에서 다른 클라이언트에서 동기화가 요청되면 동기화를 요청한 클라이언트의 변경 데이터의 변경충돌 여부를 검사한다. 서버와 변경충돌 검사 이후 다중 큐에서 처리 중인 클라이언트와도 변경충돌 검사를 하고 충돌이 없으면 다중 큐에 삽입한다. 둘 이상의 클라이언트 동기화 데이터는 다중 큐 내에서 라운드 로빈 방식으로 처리되면 먼저 쓰기 노드의 작업이 끝난 클라이언트에 대한 읽기 노드의 작업이 진행된다. 다수 클라이언트 동기화는 동기화 데이터가 작은 클라이언트의 작업이 먼저 끝나게 된다.

Ⅳ. 시스템 구현 및 성능평가

이 논문에서 구현한 다수 클라이언트의 동시 동기화 시스템의 구현 환경은 표 3과 같다. 서버는 window XP 환경에서 구현하였으며, 모바일 클라이언트는 window mobile 5.0 기반의 PDA에 구현하였다. 사용한 DBMS는 ETRI에서 모바일 기반의 DBMS이다.

표 3 시스템 구현 환경

구분	서버	클라이언트
OS	Window XP	Window mobile 5.0
DBMS	FUNs	FUNs
Language	C++	C++

구현된 동시 동기화 시스템과 기존의 순차적 동기화 시스템과의 성능을 비교하였다. 처리시간이 긴 동기화 작업과 처리시간 짧은 동기화 작업의 비율에 따라 다수의 모바일 클라이언트에 대하여 각 클라이언트의 동기화 완료시간을 측정하였다.

각 그림 2,3,4는 30개의 클라이언트에 대하여 동기화 데이터 길이 비율을 조정하여 측정된 결과 차트이다.

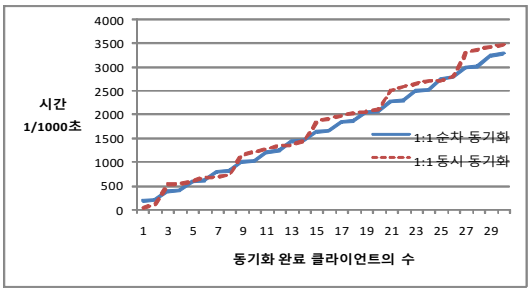

그림 2 긴 동기화 작업과 짧은 동기화 작업의 비율이 1:1 경우

그림 2는 동기화 데이터가 100개 이상인 클라이언트와 10개 이하인 클라이언트의 비율을 1:1로 하였을 때 결과이다. 이 경우는 순차적 동기화 시스템의 성능이 우수하게 나왔다.

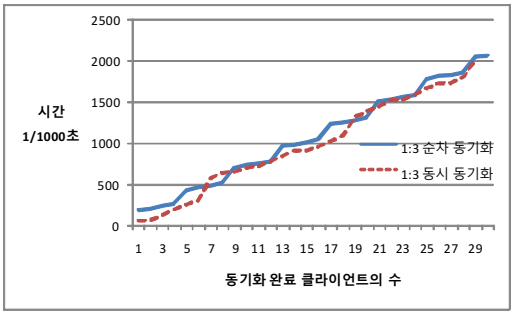

그림 3 긴 동기화 작업과 짧은 동기화 작업의 비율이 1:3 경우

그림 3은 동기화 데이터가 100개 이상인 클라이언트와 10개 이하인 클라이언트의 비율을 1:3로 하였을 때 결과이다. 이 경우는 동시 동기화 시스템의 성능이 우수하게 나왔다.

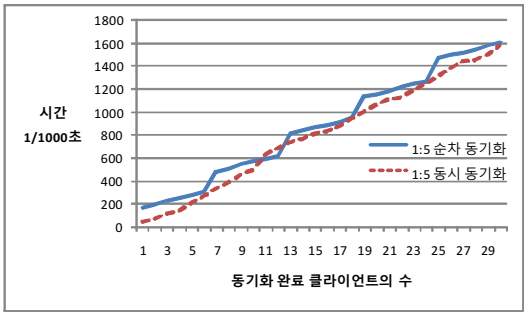

그림 4 긴 동기화 작업과 짧은 동기화 작업의 비율이 1:5 경우

그림 4는 동기화 데이터가 100개 이상인 클라이언트와 10개 이하인 클라이언트의 비율을 1:5로 하였을 때 결과이다. 이 경우 역시 동시 동기화 시스템의 성능이 우수하게 나왔다.

이 실험의 결과 소량의 공간 데이터를 자주 동기화하는 모바일 환경에서 다수 클라이언트의 동시 동기화 시스템이 효율적임을 알 수 있다.

V. 결론

이 논문에서는 다수 클라이언트에서 동시에 요청되는 동기화 작업에 대하여 변경충돌이 없는 클라이언트들의 동기화 작업을 동시에 처리하는 동기화 프로그램을 설계하고 구현하였다. 구현된 동기화 시스템은 다수 클라이언트의 동기화 작업을 다중 큐를 삽입하고 라운드 로빈 방식으로 동시 처리를 한다. 구현된 동시화 시스템과 기존의 순차적 동기화 시스템의 성능을 비교하여 동시 동기화 시스템이 다수 클라이언트의 동기화 작업에 모바일 환경의 동기화에 효율적임을 확인하였다. 향후 소량의 데이터를 자주 동기화하는 모바일 환경뿐 아니라 대용량의 동기화 데이터 처리에도 효율적으로 적용할 수 있는 동시 동기화 기법의 연구가 필요하다.

참고문헌

[1] “ActMAP White Paper and Interfaces to the FeedMAP framework," white paper, 2007

[2] 이혜진, 김진석, “모바일 환경에서 공간데이터 동기화 시스템 설계,” 한국정보과학회, 2004년 가을 학술발표논문집 제31권 제2호 (Ⅱ), 2004. 10, pp. 184~186

[3] 두용재, 진성일, “텔레매틱스 환경에서 모바일 단말과 중앙 서버간 파라미터를 이용한 동기화 기법 연구,” 한국정보과학회 2006년 가을 학술발표논문집 Vol.33, No. 2(C)

[4] 김홍기, 임창우, 이상신, 조대수, 김동현 “모바일 GIS DB를 위한 양방향 동기화 프로토콜 설계,” 한국해양정보통신학회, 한국해양정보통신학회 춘계종합학술대회, 2008년도 춘계종합학술대회 Vol. 12 No. 1, 2008. 5, pp. 183~186.

[5] 김홍기, 김동현 “모바일 환경에서 동시 양방향 동기화 프로토콜 설계,” 한국해양정보통신학회, 한국해양정보통신학회 분과학술대회, 2008년도 분과학술대회 Vol. 12 No. 12, 2008. 12, pp. 2226~2231.