Journal of Korea Multimedia Society Vol. 10, No. 4, April 2007(pp. 421-431)

이동 컴퓨팅 환경에서 데이터의 주기성을 고려한 캐쉬 일관성 기법

임종원†, 황병연††

요 약

이동 통신 기술의 급격한 발전으로, 이동 컴퓨팅 환경에서 데이터 서비스에 대한 수요가 점차 증가하고 있다. 이동 컴퓨팅 환경에서의 여러 단점을 극복하고자 캐쉬가 등장하게 되었다. 이동 호스트 내에 캐쉬가 존재하면, 통신 대역폭의 절약 및 질의에 대한 빠른 응답을 가져올 수 있지만, 이동 호스트가 서버 데이터와의 캐쉬 일관성(consistency)을 유지시켜야 한다는 문제가 생기게 된다. 본 논문에서는 기존의 기법들이 가지는 문제점들을 보완하기 위해 데이터의 주기성을 고려한 캐쉬 일관성 유지 기법을 제안한다. 제안한 기법은 데이터를 주기적, 비주기적 데이터로 분류하고 주기적 데이터에 만료시간을 삽입한다. 또한, 무효화 메시지를 저장해서 단절 후에 선별적으로 캐쉬 데이터가 나누어질 수 있도록 해서, 캐쉬 내의 데이터가 접속 단절 후에 일방적으로 버려지는 것을 막을 수 있게 된다. 이렇게 함으로서 서버와 이동 호스트간의 통신에 사용하는 전체 대역폭의 낭비와 서버접속에 필요한 이동 호스트의 전력의 낭비를 줄일 수 있다. 마지막으로 기존의 기법과의 실험을 통해서 제안한 기법의 효율성을 알아보고자 한다.

A Cache Consistency Scheme to Consider Period of Data in Mobile Computing Environments

Jong-Won Lim†, Byung-Yeon Hwang††

ABSTRACT

Due to the rapid development of wireless communication technology, demands on data services in mobile computing environment are gradually increasing. In mobile computing environments, a mobile host began to use cache to overcome the weak point. Caching at mobile host could reduce the bandwidth consumption and query response time, but mobile host must maintain cache consistency. In this paper, we propose a cache consistency strategy to consider the period of data usage. The server allows an effective broadcasting by classifying data into two groups of periodic and non-periodic. By classifying the data, it prevent from elimination of cache after disconnection of periodic data by using expired time. By storing IR(Invalidation Report) messages, this scheme divides cached data by selection after disconnection. Consequently, we have shown much improvement in total consumption of bandwidth than the conventional scheme.

Key words: Mobile Computing(이동 컴퓨팅), Cache(캐쉬), Cache Consistency(캐쉬 일관성)

1. 서 론

무선 네트워크 환경에서의 통신 기술의 발달과 함께 고성능 휴대용 컴퓨터와 기기의 등장은 이동 컴퓨팅 환경을 현실화시켰다. 이동 컴퓨팅 환경은 크게 네트워크 안에서 이동하는 이동 호스트와 이동하지 않는

※ 교신저자(Corresponding Author) : 황병연, 주소 : 경기도 부천시 원미구 역곡2동 산 43-1(420-743), 전화 : 02)2164-4363, FAX : 02)2164-4777, E-mail : byhwang@catholic.ac.kr

접수일 : 2006년 2월 24일, 완료일 : 2006년 8월 28일

† 준회원, 가톨릭대학교 컴퓨터공학과
(E-mail : skycid@catholic.ac.kr)
†† 종신회원, 가톨릭대학교 컴퓨터정보공학부 교수

※ 본 연구는 2006년도 산학협동재단 학술연구비와 2007년도 가톨릭대학교 교비연구비의 지원으로 이루어졌음.

고정 호스트의 두 가지 요소로 구성된다. 고정 호스트는 기존의 고정된 위치의 유선 네트워크에 연결된 서버와 무선 통신 인터페이스를 갖추면서 이동 호스트와 통신하여 서비스를 제공해주는 이동 지원국(Mobile Support Station: 이하 MSS)으로 나누어진다. 하나의 이동 지원국이 서비스해줄 수 있는 영역을 셀이라 하며, 같은 셀 안에 있는 이동 호스트들이 무선 채널을 통하여 동일한 이동 지원국과 통신을 하는 형태로 결합되어 있다[1].

이러한 이동 컴퓨팅 환경은 제한된 대역폭과 이동 호스트가 가진 자원의 한계[2] 때문에 낮은 대역폭(low bandwidth), 잦은 접속 단절, 제한적인 전원과 같은 문제점이 생기게 된다. 이런 이동 컴퓨팅 환경의 문제점을 해결하기 위한 대표적인 기법으로 서버 측면에서의 방송(broadcast)기법과 이동 클라이언트 측면에서의 캐쉬(cache)를 이용한 방법이 제안되었다[2-6]. 그 중 캐쉬를 사용하는 방법은 자주 사용하는 데이터를 이동 호스트의 캐쉬에 저장하는 방법이다. 하지만 캐쉬를 사용하는 방법은 불안정한 연결로 인해 잦은 접속 단절이 발생하면 이로 인해서 이동 호스트 캐쉬 내의 데이터가 서버 데이터와의 일관성을 보장할 수 없다는 부담이 생기게 된다[3]. 이런 데이터 일관성 문제를 해결하기 위한 기존의 방법으로 방송 스탬프 기법[6]과 이를 보완한 기법인 서버 요청을 통한 캐쉬 유효화 기법[7]이 있는데, 이 기법들은 무효화와 관련된 여러 문제점을 가지게 된다.

따라서, 본 논문에서는 갱신 데이터의 내용을 직접 전송해주는 전파 기법과 갱신 여부를 알려주는 무효화 기법, 서버 요청을 통한 유효화 기법을 혼용하여 방송하는 기법을 제안하고자 한다. 제안한 기법은 서버 내의 데이터를 주기성 여부에 따라 구분해서 그 중 만료기간이 지난 주기적 데이터에 대해선 방송을 하지 않고 무효화 메시지를 저장함으로써 이동 호스트가 접속 단절 후 연결 재개 시 서버에 저장된 무효화 메시지 테이블을 비교함으로서 발생하는 기존의 캐쉬 일관성 기법들이 지니는 문제점들을 개선하고 비효율적인 대역폭 낭비를 줄이고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 연구들과 문제점을 기술하고 3장에서는 본 논문에서 제안하는 데이터 주기성에 따른 캐쉬 일관성 유지 기법을 제안한다. 4장에서는 기존의 캐쉬 일관성 유지 기법과 제안한 기법의 시뮬레이션을 통한 성능 평가를 수행하고, 마지막으로 5장에서는 결론을 맺고자 한다.

2. 관련연구

캐쉬 무효화(cache invalidation) 기법으로서 방송 타임스탬프 기법[8]은 주기적인 무효화 보고(Invalidation Report: 이하 IR)를 방송함으로써 갱신 데이터에 대한 정보를 이동 호스트에게 전달한다. IR은 일정 기간 동안에 갱신된 데이터의 식별자와 해당 데이터가 마지막 갱신된 시점에 대한 타임스탬프의 쌍으로 이루어지게 된다. 이때, 일정 기간이란 서버에서 시스템 파라미터로 설정된 시간을 의미하며 보통 w(≥1)개의 IR 방송 구간으로 정한다. 이동 호스트는 갱신된 데이터의 식별자와 타임스탬프와 비교하여 갱신된 데이터를 캐쉬에서 삭제함으로써 서버와의 캐쉬 일관성을 유지한다. 그러나 오랜 접속 단절이 일어날 경우, 서버가 주기적으로 방송하는 무효화 보고를 받지 못할 수 있다. 그럴 경우, 이동 호스트는 현재 시점으로부터 과거 어느 시점까지의 갱신 사실을 알려주는 무효화 보고만으로는 자신의 현재 캐쉬 내용이 유효한 것인지를 확인할 수가 없으며, 이러한 경우 자신의 캐쉬 내용을 모두 무효화하게 된다. 이 기법은 짧은 접속 단절이 발생한 경우에는 좋은 기법이지만 오랜 접속 단절 이후 다시 서버와 접속한 이동 호스트에 대해서는 그다지 좋은 기법이 될 수 없다. 또한 사용빈도가 높은 캐쉬 데이터가 무효화될 경우 이동 호스트들의 요청이 빈번하게 되어서 대역폭의 소모가 많을 수 있다.

캐쉬 유효화(cache validation) 기법으로서 SCC (Simple Checking Caching) 기법[9]은 접속 단절로 인하여 무효화 보고만으로는 캐쉬의 유효성 확인이 어려울 경우에 이동 호스트가 서버에 캐쉬 데이터에 대한 유효성 확인 요청을 함으로써 일관성을 유지시키는 기법이다. 이 기법은 캐쉬 내 데이터들의 식별자와 이들의 유효성을 가장 최근에 확인했던 타임스탬프를 서버에게 보내고 서버는 이에 대한 정보를 요청한 이동 호스트에게 전송한다. 하지만 이 기법은 캐쉬 내 데이터가 많을 경우에 각각의 식별자 모두를 서버로 전송하기 때문에 대역폭의 심한 낭비를 초래할 수 있다는 단점을 가지고 있다.

SGC(Simple Grouping Caching) 기법[9]은 접속

단절 후 각각의 이동 호스트가 서버 요청을 할 경우 대역폭 낭비가 심하다는 단점을 보완하기 위해 제안된 기법으로서, 데이터 각각의 식별자를 서버에 보내지 않고 데이터를 미리 정해진 규칙에 의해 그룹화하고 각 그룹의 식별자를 서버에 보내어, 그룹별로 유효성 확인 요청을 하게 된다. 캐쉬 일관성 유지에 필요한 대역폭의 양은 줄일 수 있지만 이 기법의 경우 그룹 내의 갱신되지 않은 데이터도 무효화 그룹 대상에 포함되면 캐쉬에서 삭제되는 잘못된 무효화가 발생하게 된다. 결국 이것은 잘못된 무효화에 따른 데이터에 대해서 이동 호스트가 그에 대한 또 다른 확인을 서버에 요청하게 되므로 결론적으로 그에 따른 데이터 검색에 필요한 대역폭의 양이 늘어나게 되는 단점을 지니게 된다.

적응적 캐쉬 일관성 유지 기법[10]은 무선 네트워크의 연결 상태에 따라 무효화와 전파 메시지의 방송 비율의 경계값을 기준으로 동적으로 조절해서 캐쉬 일관성 유지를 효율적으로 하게 된다. 그림 1은 페이지의 접근 횟수를 전파와 무효화의 경계값 S로 설정한 예를 보여주고 있다. 무선 네트워크의 상태가 안정적일 경우, 경계값을 S에서 S′로 설정하여 전파 메시지의 방송에 많은 부분을 할당하여 캐쉬 요청을 감소시키고 서버의 응답 시간을 단축시키도록 한다. 반면에 전체 시스템의 평균가용 통신 대역폭이 감소하여 무선 네트워크의 연결 상태가 불안정하게 될 경우 경계값 S에서 S″로 재설정하여 전파메시지를 줄이고 무효화 메시지의 양을 늘린다. 이렇게 함으로써 메시지 파손 및 손실 확률을 최소화한다. 하지만 이 기법은 이동 호스트의 갱신과 서버의 갱신을 구별하지 않기 때문에 이동 호스트에 존재하지 않는 데이터에 대한 서버 측의 잦은 갱신으로 인해 전파가 이루어질 경우 불필요하게 많아지는 데이터의 방송으로 인한 대역폭의 낭비를 초래한다는 단점이 있다. 그리고 단절이 길어질 경우 기존의 기법과 같이 데이터 전체가 무효화되는 단점이 있다.

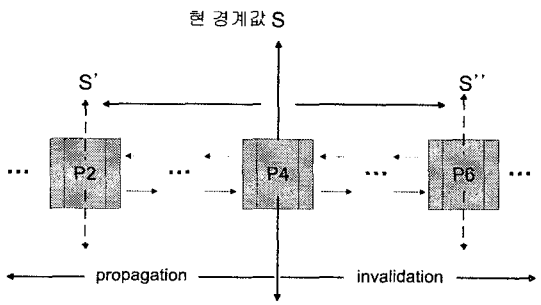

그림 1. 경계값 설정

선택적 캐쉬 일관성 유지 기법[11]에서 서버는 적응적 캐쉬 일관성 유지 기법을 기반으로 해서 주기적으로 데이터 갱신 정보를 방송하게 된다. 이동 호스트는 이 정보를 수신하여 캐쉬 데이터의 유효성 여부를 판단할 수가 있다. 각 이동 호스트 내에 보유하고 있는 캐쉬에 대해서 서버에 캐쉬 상태 테이블(Cache State Table: 이하 CST)을 두어서 잘못된 무효화나 이동 호스트 내에 존재하지 않는 데이터에 대해선 방송을 하지 않도록 함으로써 서버와 이동 호스트 간의 캐쉬 일관성을 유지한다. 하지만 이 기법은 이동 호스트가 잦은 접속 단절이 일어날 경우 비동기적 방송이 많아져서 효율적이지 못할 수가 있다. 그리고 기존의 캐쉬 유효화 기법과 마찬가지로 셀 내의 수많은 이동 호스트가 접속 단절이 일어날 경우 유효성 요청으로 인해 대역폭의 낭비가 많을 수 있다는 단점을 지닌다.

3. 데이터의 주기성을 고려한 캐쉬 일관성 유지 기법

3.1 제안한 기법의 설계

시스템 환경을 구성하기 위해서 여러 방식의 다양한 통신 모델을 사용할 수가 있는데 일반적으로 이동 컴퓨팅 환경은 업 링크와 다운 링크 통신 채널로 구성된다. 본 논문에서는 데이터의 전달 형식을 서버에 의한 비대칭 통신 환경에 기인하는 방송 형태의 전파(propagation) 방식과 이동 호스트의 요청에 의해서 링크 채널을 통해 송·수신하는 메시지 전달 방식을 사용하기로 한다. 본 논문에서 가정하는 시스템 모델은 다음과 같다.

① 서버에서는 읽기와 갱신 트랜잭션이 모두 가능하다.
② 이동 호스트에서는 단지 읽기 트랜잭션과 데이터에 대한 요청만이 가능하다(자체갱신 불가).
③ 이동 호스트내의 데이터는 서버로부터의 다운 링크 채널에 의한 갱신이나 방송에 의한 갱신으로만 가능하다.

이동 호스트는 캐쉬 데이터를 사용하여 트랜잭션

을 수행하기 때문에 서버 트랜잭션과는 별도로 캐쉬 데이터의 유효성을 검사하는 과정이 필요하다. 예를 들어 A라는 데이터가 필요하다고 가정할 경우 이동 호스트내의 캐쉬에 해당 데이터가 있는지를 검색하고 없을 경우 서버에 요청한다. 이동 호스트로부터의 이와 같은 데이터 요청을 받은 서버는 서버의 유효성 확인 모듈을 통해서 그 여부를 확인 후 해당 이동 호스트로 전달하게 된다.

기존의 연구에서는 서버 방송 시 이동 호스트의 요구를 일정 시간만큼의 요청을 수렴하여 저장한 뒤에 그것을 종합하여 다음 주기 방송에 포함시키는 방식을 사용했다. 이 방식은 대역폭의 소모는 적은 반면에 이동 호스트의 요청을 받는 즉시 처리하기가 어렵다. 또한, 이동 호스트의 요청이 빈번해져서 주기가 길어지게 될 경우 적절한 시간에 이동 호스트에게 데이터를 전달할 수 없어서 응답시간이 지연될 수 있다는 단점이 생기게 된다. 그러므로 본 논문에서는 서버가 자체적으로 이동 호스트의 요청을 주기적이 아닌 요청을 받을 때마다 즉시 데이터를 전송하도록 하였다. 이렇게 함으로써, 기존의 기법보다 이동 호스트 캐쉬 내 데이터의 적중률을 높일 수 있고 서버와의 일관성 유지에 대한 효율성을 증대시킬 수 있게 된다. 이동 호스트의 접속이 끊긴 채 시간이 많이 경과할 경우, 기존의 연구에서는 이동 호스트들의 캐쉬 내의 데이터가 서버 데이터와의 일관성을 유지할 수 없어서 모두 삭제하는 반면에 본 논문에서는 데이터의 주기성에 따른 만료시각 삽입으로 오랜 접속 단절 후에 생기는 이 문제를 보완하고자 하였다.

일반적으로 이동 컴퓨팅 환경에서 다루는 데이터들에는 객체의 상태를 나타내는 여러 가지 종류가 있을 수 있다. 그 중에서 일기예보나 TV프로그램 편성표, CP(Content Provider)와의 계약에 의한 콘텐츠, 증권 소식, 뉴스 등과 같은 종류의 데이터가 있을 수 있는데 이 데이터들을 살펴보면 공통점이 존재한다. 그것은 데이터가 한 번 갱신된 후 다음 갱신이 이루어질 때까지의 기간이 길던지 짧던지 간에 일정한 기간이 존재한다는 것이다. 그리고 이 기간 동안의 데이터는 유효한 데이터라고 말할 수가 있다. 이렇게 특정한 데이터가 그 데이터의 내용이 일정기간 동안 유효하다면 그 데이터를 주기적 데이터라 하고 그 외의 데이터를 비주기적 데이터라고 칭하기로 한다. 이렇게 일정한 주기로 갱신되어야하는 데이터에 한해서 타임스탬프 형태의 만료시각을 삽입해서 사용할 경우 접속 단절 후의 이동 호스트의 서버에 대한 캐쉬 유효성을 요청할 때 서버에 저장된 데이터 테이블을 전체적으로 검색하는 것보다 본 논문에서 제안하는 저장된 무효화 보고 테이블(Invalidation Report Table: 이하 IR_table)만 검색하게 함으로써 보다 빠른 검색을 제공할 수 있으므로 효율적인 방법이 될 수 있다.

본 논문에서 서버는 주기적으로 데이터 갱신에 대한 정보를 이동 호스트에게 전파 또는 무효화함으로서 방송하게 된다. 그러면 이동 호스트에서는 방송되는 정보를 수신해서 캐쉬 내 데이터의 유효성을 검증할 수 있게 된다. 또한, 개별 이동 호스트가 서버와의 접속이 끊어진 후 오랜 시간이 경과한 경우, 해당 이동 호스트는 자신의 캐쉬 데이터의 유효성을 알아보기 위해 서버에 유효성 확인 요청을 하게 된다. 그 메시지를 받은 서버는 평상 시 IR_table에 저장된 서버 자체 갱신 무효화 데이터와 주기적 데이터의 만료시각 삽입에 따른 무효화 데이터를 검색, 비교 후 요청한 데이터 중 만료시각이 지난 것이 존재한다면 해당 데이터를 무효화하는 메시지와 접속 단절기간 동안의 갱신 정보는 데이터 링크채널을 통해 송신하게 된다.

3.2 서버 작업 모델

본 논문에서 제안하는 서버는 데이터 방송 서버를 하나 가지고 있고 이동 호스트가 요청하는 데이터를 갱신 또는 무효화를 통해서 방송하게 된다. 서버가 관리하는 데이터는 크게 방송 데이터, 무효화 보고 데이터, 그리고 개별 방송 메시지 데이터로 분류할 수 있다. 또한 서버는 이렇게 나누어지는 데이터를 테이블의 형태로 저장하고 있다. 데이터를 저장하는 테이블은 크게 데이터 테이블(Data_Table)과 IR_table로 나누어진다. 개별 방송 메시지 데이터는 이동 호스트의 개별적인 요청이 있을 때마다 해당 이동 호스트로 전송이 되기 때문에 따로 테이블을 만들어 저장하지 않는다.

표 1은 데이터가 실제로 저장되는 테이블을 나타내는 것으로 Data_table로 표현한다. 여기서 type은 주기적/비주기적 데이터를 구별하기 위한 식별자로서 type 1은 주기적 데이터를 type 2는 비주기적 데이터를 나타낸다. did는 Data Identifier로서 각 데이

터들의 식별자를 의미하고 fid는 Fragment Identifier로서 각 데이터가 여러 개의 조각으로 쪼개져서 전송된다고 할 때의 각 조각의 식별자이다. data는 실제 전송되는 데이터의 내용이고, regdate는 RegistrationDate로서 데이터가 서버에 등록된 생성 시간을 의미한다. 마지막으로 edate는 타임스탬프로 구성된 해당 데이터의 서비스 만료시각을 나타낸다. 그러므로 어떤 데이터가 얼마동안 유효한지 알아보려면 regdate와 edate를 알아보면 가능하다.

표 1. 서버에 저장되는 Data_table의 예

type	did	fid	data	regdate	edate
1	1	1	a1	2005-10-20 오전 12:00:00	2005-10-22 오전 12:00:00
		2	a2		
		3	a3		
2	2	1	b1	2005-10-20 오전 10:00:00	
		2	b2		
1	3	1	c1	2005-10-21 오후 1:00:00	2005-10-23 오전 9:00:00
		2	c2		
1	4	1	d	2005-10-21 오전 2:00:00	2005-10-22 오전 12:00:00

표 2. 서버에 저장되는 IR_table

did	updatedate	updatemode	bflag
		D or U	8N or Y

표 2는 SIO(Server Invalidation Operation)에 의한 무효화 보고와 서버 자체 갱신에 의한 무효화 보고를 데이터베이스 형태로 저장한 것을 보여주고 있다. 표 2에서 did는 표 1과 마찬가지로 Data Identifier로서 각 데이터들의 식별자를 의미하고 updatedate는 이 데이터가 언제 갱신이 되었는지를 알려주는 타임스탬프 형태의 갱신시간 식별자이다. 그리고 bflag는 처음에 SIO에 의해서 무효화된 데이터가 IR_table로 저장할 때 초기 값으로 'N'을 설정하고 그 데이터가 무효화 보고 방송을 통해서 이동 호스트에게 전송되었을 경우 'Y'로 바뀌게 된다. 마지막으로 updatemode는 이 데이터가 서버 자체 갱신에 의한 무효화 보고 데이터인지 아니면 만료시각 초과에 따른 SIO 무효화보고 인지를 나타내는 식별자이다. SIO에 의한 갱신일 경우 D로 표시하고 서버 자체 갱신일 경우 U로 표시한다.

기존의 기법들이 전파와 무효화 방송을 하는 경우를 가정할 때, 이동 호스트는 첫 번째 방법인 접속 단절 후에 자신의 캐쉬 내의 내용을 전부 버리고 새로 받는 방법이 있을 수 있고, 두 번째로 자신이 가지고 있는 캐쉬 내용을 서버에 요청해서 유효성 요청 검사를 받는 방법이 있을 수 있다. 여기서 첫 번째 방법은 대역폭 측면이나 데이터의 자치성 측면에서 모두 좋지 않은 결과를 가져오게 되고 두 번째 방법에서도 방송 데이터에 대한 데이터 테이블을 전부 검색하게 되므로 시간이나 연산에 있어서 비효율적인 면이 있을 수 있다.

본 논문에서 제안하는 SIO는 서버 데이터의 만료시각 검사에 의한 무효화 작업이라고 할 수 있다. 이것은 주기적 데이터 안에 만료시각을 설정한 데이터로 만료시각이 지난 데이터에 한해서 서버가 주기적으로 검색해서 따로 IR_table로 저장시키게 되는 것을 말한다. 서버는 이 작업을 기본적으로 반복하게 된다. 그림 2는 SIO를 수행하는 SQL구문을 나타낸 것이다.

```
① select did from Data_table
     where edate<sysdate
② delete from Data_table
     where did
     in (select did from Data_table
       where edate<sysdate)
③ insert IR_table values(did)
```

그림 2. SIO SQL 구문

그림 3은 서버 프로세스를 나타낸 것이다. 서버는 다음과 같은 작업을 수행하게 된다. 첫째, IrUpdateF() 함수는 만료시각 초과에 따른 무효화 보고 갱신 함수로서 서버는 반복적으로 먼저 SIO()함수를 수행하면서 각 데이터의 만료시각을 검사하게 된다. 그리고 SIO()함수에 의해서 만료시각이 지난 데이터들은 이동 호스트에게 무효화 방송을 하는데 이때 bfalg='N'인 값을 선택해서 방송을 하고 방송한 다음에는 해당 데이터의 bflag를 'Y'로 바꾸게 된다. 둘째, SUpdateF() 함수는 서버 데이터 갱신함수로서 서버 데이터에 갱신이 일어날 경우 사용하게 된다. 예를 들어 서버의 데이터 항목 x에 갱신이 일어난다고 가정할 경우, data_table에서 데이터 항목 x를 갱신하고(Execute

UpdateQuery), IR_table로 갱신된 데이터 항목 x를 저장하게 된다(ExecuteInsertQuery). 그리고 마지막으로 해당 데이터가 존재하는 이동 호스트에게 갱신 방송(BroadcastingUp)을 하게 된다.

서버는 이동 호스트의 요청에 따라서 데이터를 방송하고 서버 갱신에 의한 갱신 데이터 방송을 함으로써 이동 호스트와의 일관성을 유지하게 된다. 하지만 접속 단절이 일어나고 서버와의 통신이 재개될 경우 이동 호스트는 단절 기간 동안 방송을 수신할 수가 없어서 현재 캐쉬 데이터가 유효한지 확인할 수가 없다. 따라서 본 논문에서는 접속이 재개된 이동 호스트가 서버에 유효성 확인 요청을 하면 서버는 이때까지 갱신된 데이터와 무효화 된 데이터를 모아놓은 IR_table에서 이동 호스트에서 요청한 데이터가 있는지 확인하도록 한다.

```
Function
(i) IrUpdateF()  // 만료시각 무효화 갱신 함수
    {
    while(){
          while(){
               SIO_Function();
          }
   // IR테이블에서 bflag='N'을 갖는 데이터를 선택
          SelectQuery(bflag=N) from IR_table;
          while(){
   // SelectQuery의 결과 값인 did를 저장
             Resultdid = getString("did");
             for(i=1,i<mem.size();i++){
                mem.sendMsg("IRD");
             }
          }
   // 방송한 데이터일 경우
       UpdateQuery(bflag=Y) from IR_table;
   // IR테이블의 bflag를 'N'에서 'Y'로 갱신하는 함수
       }
    }
(ii) SUpdateF() // 서버 데이터 갱신 함수
   {
      if(ServerUpdate = true){
         ExecuteUpdateQuery();
         ExecuteInsertQuery();
         BroadcastingUp();
      }
   }
```

그림 3. 서버 수행 동작

그림 4는 서버가 접속 단절 후에 이동 호스트가 자신의 캐쉬 데이터의 유효성 확인 요청을 할 경우에 사용되는 함수인 MobileQuery()함수에 대한 설명이다. 예를 들어 이동 호스트로부터 데이터 항목 x에 대한 질의 요청(MobileRequest)을 받은 경우에 서버는 x의 만료시각 무효화 여부를 IR_table로부터 검색(CheckIR_table)하고 만약 x가 만료시각 무효화가 되어있다면(DataIR=true), x에 대한 무효화 메시지(SendIrMsg)를 해당 이동 호스트에게 전송한다. x가 만료시각 무효화가 안 되어있을 경우 서버 자체 갱신 업데이트를 검색(DataUp)하고, 되어 있을 경우 이동 호스트에게 data_table에서 갱신된 서버 데이터를 전송(SendUpdata)하게 된다.

```
Function
  // 이동호스트로부터의 질의 처리 함수
   MobileQueryFu()
      {
           MobileRequest = true
           CheckIR_table();
              if (DataIR=true)
                 SendIrMsg();
              else{
                 if(DataUp()=true)
                    SendUpData();
                 }
      }
```

그림 4. 서버 질의 처리 함수

이 방법을 사용하게 되면 이동 호스트는 접속 단절로 인해 방송을 수신 받지 못한 경우에도 재접속 후에 갱신 정보를 얻을 수 있게 된다. 또한 기존의 기법들은 접속 단절 후 캐쉬에서 모든 데이터가 삭제되지만 이 경우에 만료시각 초과에 의해 데이터가 무효화 되지 않는다면 그 데이터는 계속 이동 호스트의 캐쉬 내에 존재하게 되므로 캐쉬에서 버려지는 데이터가 적게 되고 또한 최신 갱신 정보를 수신 받을 수 있어서 캐쉬의 손실을 줄일 수가 있다.

3.3 이동 호스트 작업 모델

본 논문에서 이동 호스트는 서버로부터 IR 정보를 받아 캐쉬의 유효성 여부를 판단한다. 또한 방송 채널을 통해 서버의 방송 데이터를 수신 받게 되고 업링크 채널을 통해서 서버에게 데이터를 요청한다. 그

리고 각각의 이동 호스트는 캐쉬를 유지하며 캐쉬 데이터를 사용하기 전에 서버의 IR메시지를 사용하여 캐쉬의 유효성 여부를 판단한다. 이동 호스트에서 수행하는 동작은 다음과 같다.

첫째, 서버로부터 방송이나 갱신 데이터 x를 수신 받았을 경우 먼저 그 데이터가 방송데이터면 현재 이동 호스트 내의 캐쉬에 존재하는지 검색한 후 존재하면 현재 캐쉬 데이터를 사용하고 존재하지 않을 경우 현재 캐쉬 데이터로 저장한다. 여기서 갱신 데이터일 경우에는 기존의 캐쉬 데이터를 삭제하고 현재 서버로부터 받은 데이터를 삽입하는 것으로 대체하게 된다. 둘째, 서버로부터 IR메시지를 받은 경우 IR메시지를 현재 캐쉬 데이터와 비교해서 일치하는 데이터가 있으면 해당 데이터를 캐쉬에서 삭제하고 그렇지 않을 경우 해당 캐쉬를 유지한다. 마지막으로 접속 단절 후 서버에게 유효성 요청을 하는 경우에는 서버에게 해당 데이터에 대해서 유효성 확인 요청 메시지를 보내고 그 메시지를 받은 서버는 IR_table검색 후 갱신 또는 무효화에 대한 데이터를 이동 호스트에게 전달하게 된다. 이동 호스트가 이 메시지를 받는다면 과정 (i)과 (ii)를 수행하게 된다. 이동 호스트는 이렇게 서버와 통신을 함으로써 서버 데이터와 캐쉬 데이터와의 일관성을 유지하게 된다. 그림 5는 이 과정을 나타낸 것이다.

```
Operation
  (i) 서버로부터 방송 데이터 x를 수신 받았을 경우
      if x∈BroadData{
          if x ∈ CurrentCacheData{
              return CurrentCacheData;
          else x ∉ CurrentCacheData
              Insert to CurrentCacheData(x);
          }
      }
  (ii) 서버로부터 IR메시지 x를 수신 받았을 경우
      if x ∈ IRMsg{
          if x ∈ CurrentCacheData{
            delete Data_IRMsg;
          else x ∉ CurrentCacheData
            return CurrentCacheData;
          }
      }
  (iii) 접속 단절 후 서버에게 데이터 x에 대한
      유효성 확인 요청을 하는 경우
      if x ∈ RequestData{
          Send to Server_request_msg(x)
        if receipt to Sever_response_msg(x)
          (i), (ii) 과정 수행
        }
      }
```

그림 5. 이동 호스트의 동작

본 논문에서 제안하는 전체적인 시스템의 개괄적인 구조는 다음과 같다. 그림 6에서 보여 주고 있는 각 구성 요소들을 설명하면 다음과 같다.

<서버 측 구성요소>

- Validity Process
  - 이동 호스트와의 유효성을 검증하기 위한 프로세스
- Sync Process(Synchronization Process)
  - 서버에서 주기적으로 행하는 작업인 서버 내부 무효화 작업(Server Invalidation Operation)을 담당하는 프로세스
- IR Manager(Invalidation Report Manager)
  - 서버 자체 갱신과 만료시각이 지난 데이터들의 무효화 보고 방송을 위한 관리자
- BC Manager(Broadcasting Manager)
  - 서버 자체 갱신에 대한 방송을 담당하는 관리자
- Data Manager
  - 서버에 저장되는 데이터베이스와의 연동을 위한 관리자

<이동 호스트 측 구성요소>

- Validity Demander
  - 서버와의 접속 단절 시 일정 시간 단위로 접속 재개 요청을 하는 관리자
  - 서버와 접속이 재개되면 캐쉬 안의 데이터 유효성 확인 요청을 함
    (결과적으로 캐쉬 데이터의 일관성을 유지하는 역할을 담당)

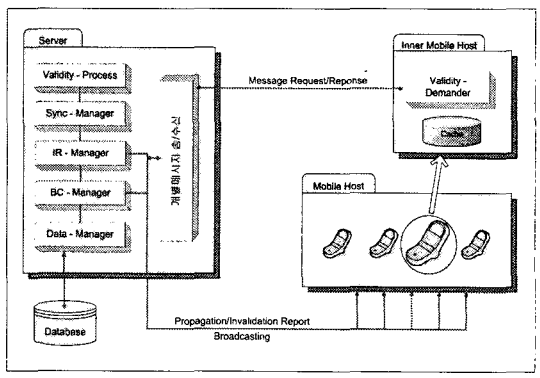

그림 6. 제안 기법의 구조

4. 성능평가

4.1 실험 환경

표 3은 본 논문에서 성능 평가를 수행한 실험 환경을 나타낸 것이다. 이 중 모바일 환경언어는 기존의 방법들이 서버와 클라이언트를 무선으로 연결하거나 서버 내에서 클라이언트를 생성하는 방식을 취해왔으나 이것은 실제 환경과는 차이가 있다고 생각해서 실제 환경과 가장 근접하게 이동 컴퓨팅 환경을 구축할 수 있는 J2ME를 사용하였다.

이동 호스트의 생성 개수는 두 대의 클라이언트 내의 동일한 수로 나누어서 생성되도록 실험하였다. 실험 내의 단절 부분에 있어서는 모든 이동 호스트가 단절의 원인에 구별 없이 동일한 단절 상태로 될 수 있도록 하였으며, 전체 네트워크상의 모든 셀을 대상으로 한 것이 아닌 단일 셀 내의 이동 호스트를 가정하므로 실험 도중에 셀 내에 새로운 이동 호스트가 진입하거나 반대로 이동 호스트가 셀 밖으로 탈퇴하는 경우는 없다고 가정하였다. 또한, 셀 내의 모든 이동 호스트는 처음 실험을 시작할 경우 서버에서 보내주는 전체 서버에서 생성되는 데이터 중 일정 부분에 해당하는 동일한 캐쉬 데이터를 갖고 생성된다고 가정한다.

본 논문에서 사용하는 성능평가 매개 변수는 표 4와 같다. 매개변수의 값은 측정하고자 하는 성능 평가 요소에 따라 각각 다르게 설정될 수 있으며, 매개변수 값의 변화 내용과 실험에 부가적으로 사용되는 매개변수들은 성능 평가 수행 시에 추가적으로 설명하고자 한다.

본 성능평가에 사용되는 서버와 이동 호스트간의 메시지의 종류는 다음과 같이 분류되며 각 메시지들은 이동 호스트와 서버에서 사용하는 형태들로 구분되고 표 5와 같다. 표 5에서 flag는 서버와 이동 호스트가 주고받는 메시지가 어떤 종류에 해당되는지 보여주는 값이고 Hostid는 각각의 서버와 연결된 이동 클라이언트의 식별자이다. Did는 데이터의 식별자이고 Set은 데이터들의 집합을 의미한다. 그리고 접두사인 IR_와 Up_는 각각 무효화와 갱신을 의미하는 식별자이다. Data는 Did에 해당하는 실제 데이터 값들을 뜻한다. 각 메시지가 주고받을 경우 요청/응답하는 일련의 과정이 끝나면 S(Success), D(Delete), U(Update)등의 확인자를 덧붙여 다시 전송할 수 있도록 하였다.

표 3. 실험 환경

구 분	명 칭
CPU & Memory	P4 1.6GHz, RAM 512MB
운영체제	Windows 2000server, XP
데이터베이스 관리시스템	MS-SQL 2000
구현 언어	Java 1.4.2_07
모바일 환경 언어	J2ME Wireless Toolkit 2.2

표 4. 성능평가 매개 변수

매개변수	설정 값(단위)
서버 내 데이터의 수	1000(개)
데이터 조각의 수	1~5(개)
이동 호스트의 수	10(개)
접속 단절 주기	60~180(초)
데이터의 크기	1024(bytes)
이동 호스트 캐쉬의 크기	200(개)
IR 주기, 서버무효화연산 주기	30(초)

표 5. 실험에 사용된 자료 구조

이동 호스트 메시지 형태	
종 류	형 태
서버 유효성 확인 요청	<flag,Hostid,Set(Did)>
데이터 요청	<flag,Hostid,Set(Did)>
서버 메시지 형태	
유효성 확인 응답	<flag,Hostid,(Data(Did)), (IR(Did))>
데이터 요청 응답	<flag,Hostid,(Did,Data)>
무효화/갱신	<flag,IR_Set(Did),Up_Set(Did)>

4.2 실험결과

대역폭 소모량은 유선 환경에 비해 협소하다는 특징을 가지고 있어서 이동 컴퓨팅 환경에서는 가장 민감한 부분이기 때문에 이동 컴퓨팅 환경의 성능평가에 있어서 중요한 지표가 될 수 있다. 대역폭 소모량 측정에 있어 본 논문의 성능평가의 기준은 서버와 이동 호스트가 주고받는 데이터의 양을 통합적으로 계산한다. 여기에는 전체 갱신 방

송과 무효화 보고 방송 그리고 서버와 개인 이동 호스트가 송수신하는 개별 메시지 전송도 포함하게 된다. 이것을 전체 실험 수행 시간동안의 분당 소모되는 전체 데이터양으로 측정하게 된다. 또한 이것은 이동 컴퓨팅 환경에서 보면 전체 대역폭에서 이용되는 대역폭 소모량과도 동일하다. 대역폭 소모량에 있어서 중요한 요소는 서버와 이동 호스트간의 단절 시간과 지속 시간 등이 있을 수 있다. 다음은 본 논문에서 사용하는 접속 단절에 대한 매개변수이다.

- 접속 단절 시간 : N
- 단절 후 재 연결 시간 : N
- 연결/재연결 지속시간 : 2N

위와 같이 가정했을 경우 첫 번째 N=15초일 경우 단절에서 그 다음 단절까지의 주기는 위의 매개 변수들을 전부 합한 값인 (N+N+2N=4N)이 되므로 60초가 되고 N=30초일 경우 120초가 되며 마지막으로 N=45초가 될 경우 180초가 되게 된다. 본 논문에서 대역폭 소모량은 이 세 개의 단절 주기를 변화시키면서 소모되는 대역폭의 양을 측정하였다. 그림 7, 8은 단절주기가 각각 1분, 3분인(N=15, 45초) 경우 전체 이동 컴퓨팅 환경에서 측정된 대역폭 소모량을 나타낸 것이다. 각 그림에서 제안기법은 본 논문에서 제안한 데이터 주기성을 고려한 캐쉬 일관성 유지 기법을 의미하고, TS전파 기법은 기존의 캐쉬 기법 중의 하나인 타임스탬프를 이용한 전파 메시지 전송 기법을 의미한다.

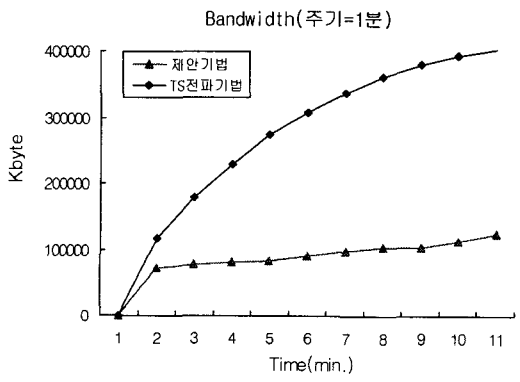

그림 7. 단절 주기가 1분일 경우 총 대역폭 소모량

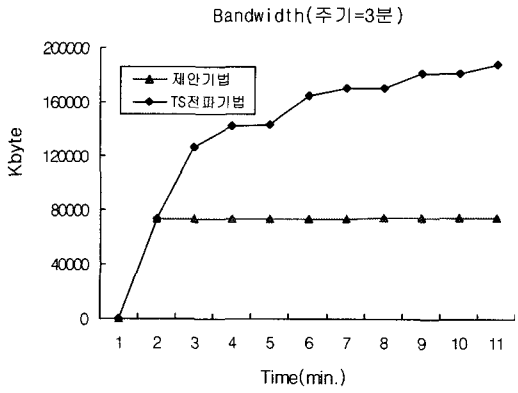

그림 8. 단절 주기가 3분일 경우 총 대역폭 소모량

위의 그림을 보면, 시간이 지나감에 따라서, 모든 기법의 대역폭 소모량이 증가하지만 제안된 기법이 TS전파기법에 비해서 훨씬 더 적게 대역폭을 소모하고 있다는 것을 볼 수 있다. 이것은 TS전파기법의 경우 본 논문에서 가정하고 있는 잦은 접속 단절이나 오랜 접속 단절의 경우에 있어서 단절 후 이동 호스트 내 캐쉬의 유효성을 확인해야 할 경우 접속이 재개된 후에 캐쉬 내의 데이터를 모두 제거하고 서버로부터 전부 다시 받아야 하기 때문에 이런 과정에서 소모되는 대역폭의 양이 매우 많다는 것을 의미한다. 한편, 본 논문에서 제안한 기법은 잦은 접속 단절의 경우 TS전파기법에 비해 훨씬 적은 대역폭을 소모하고 있고, 상대적으로 오랜 접속 단절의 경우에도 그 차이가 좁혀지긴 하지만, 또한 매우 우수한 성능을 보이고 있다. 이것은 제안한 기법이 접속 단절 후의 이동 호스트가 서버 자체 갱신과 무효화에 대해서 새로 생성된 IR_Table을 검색 시에 사용하게 되므로 방송 스탬프 기법에 비해서 적은 대역폭을 소모하는 원인이 된다. 그리고 잦은 접속 단절 즉 데이터 갱신이나 무효화가 많이 발생하여 캐쉬의 데이터를 빈번하게 교체해야 할 경우에도 만료시각에 따른 데이터의 재사용성을 높일 수 있게 되어서 대역폭의 사용량을 줄이는데 있어서 효율적인 측면을 가져오게 된다.

5. 결 론

본 논문에서는 이동 컴퓨팅 환경에서 기존의 캐쉬 일관성 유지 기법이 가지는 단점인 접속 단절에 따른 캐쉬 손실을 줄이는데 초점을 맞추고 있다. 즉 제안

한 기법은 데이터의 주기성을 고려하여 주기적/비주기적 데이터로 분류하고 그 중 주기적 데이터에 만료시각을 두어서 IR_table에 저장함으로써 기존의 접속 단절 후에 캐쉬 데이터를 재사용하지 못하거나 버려야 되었던 것을 IR_table을 검색 후 유효성을 검증할 수 있게 되어서 기존의 문제점을 최소화시켰다. 이런 점은 또한 캐쉬 내 데이터의 재사용성을 높이게 되므로 이동 호스트의 자치성 회복에도 도움을 줄 수 있게 된다. 결론적으로 제한된 통신 대역폭과 잦은 접속 단절의 특징을 가진 이동 컴퓨팅 환경에서의 방송 스탬프 전송기법에 비해 상대적으로 전체 대역폭 소모량에 있어서 우수한 성능을 가진다는 것을 실험을 통하여 증명하였다. 그러나 제안된 기법은 기존의 캐쉬 일관성 유지 기법과의 비교가 데이터의 분류에 의해 제한되므로 적절한 비교 대상을 찾기가 용이하지가 않다. 또한, 데이터의 만료시각에 따른 서버의 테이블 유지에 드는 부가적인 작업으로 인해서 서버 테이블의 부하와 그로 인한 서버와 이동 호스트 간의 대역폭 및 성능 문제가 야기될 수 있으므로 데이터의 만료시각의 명확한 기준을 마련하는 것과 IR테이블과 데이터 테이블의 전체적인 검색 성능과 시간을 향상시킬 수 있는 좀 더 효율적인 구조의 연구가 향후 과제로 남게 된다.

참 고 문 헌

[1] D. Barbara, "Mobile Computing and Database - A Survey," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 11, No. 1, pp. 108-177, 1999.

[2] J. Cai, K. Tan, and B. Ooi, "On Incremental Cache Coherency Schemes in Mobile Computing Environment," *Proc. Int'l Conf. on Data Engineering*, pp. 114-123, 1997.

[3] M. Wong and W. Leung, "A Caching Policy to Support Read-only Transaction in a Mobile Computing Environment," *Technical Report CS-TR-95-07*, The Chinese Univ, 1995.

[4] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments," *Proc. of the ACM SIGMOD Conf*, pp. 199-210, 1995.

[5] S. Acharya and M. Franklin, "Balancing Push and Pull for Data Broadcast," *Proc. of the ACM SIGMOD Conf*, pp. 183-194, 1997.

[6] D. Barbara and T. Imielinski, "Sleeper and Workaholics: Caching Strategies in Mobile Computing Environments," *Proc. Int'l Conf. on Data Engineering*, pp. 1-12, 1994.

[7] K. Wu, P. Yu, and M. Chan, "Energy-Efficient Caching for Wireless Mobile Computing," *Proc. Int'l Conf. on Data Engineering*, pp. 336-343, 1996.

[8] B. R. Badrinath, A. Acharya, and T. Imielinski, "Structuring Distributed Algorithms for Mobile Hosts," *Proc. Int'l Conf. on Distributed Computing System*, pp. 21-28, June 1994.

[9] K. Wu et al., "Energy-Efficient for Wireless Mobile Computing," *Proc. Int'l Conf. on Data Engineering*. pp. 336-343, 1996.

[10] 남성헌, 조성호, 황종선, "이동 컴퓨팅 환경하의 연결 상태를 기반으로 한 적응적 캐쉬 일관성 유지 기법," *Journal of Computer Science & Engineering Technology*, Vol. 3, No. 1, 2001.

[11] 김희숙, 황병연, "이동 컴퓨팅 환경에서 이동 호스트의 자치성 증대를 위한 선택적 캐쉬 일관성 유지 기법," *정보처리학회논문지*, Vol. 10, No. 4, pp. 655-660, 2003.

황 병 연

1986년 서울대학교 컴퓨터공학과(공학사)
1989년 한국과학기술원 전산학과(공학석사)
1994년 한국과학기술원 전산학과(공학박사)
1994년~현재 가톨릭대학교 컴퓨터정보공학부 교수
1999년~2000년 University of Minnesota Visiting Scholar
관심분야 : XML 데이터베이스, 데이터마이닝, 지리정보시스템, 정보검색

임 종 원

2003년 충북대학교 구조시스템 공학과(공학사)
2006년 가톨릭대학교 컴퓨터공학과(공학석사)
2006년~현재 선도소프트 근무
관심분야 : 무선데이터베이스, 공간데이터베이스, XML