공간 데이타베이스 구축을 위한 수치지도 타일 합병 트랜잭션 모델

(DXF TILE Merge Transaction Model for Spatial DB)

이상현*, 김 동현**, 홍봉희***

(Sang-Hyun Lee*, Dong-Hyun Kim**, Bong-Hee Hong***)

초 록

지형정보시스템(이하 GIS)에 있어서 공간 데이타베이스의 구축은 GIS 전체 구축비용의 많은 부분을 차지하며 막대한 시간과 비용이 소요되는 과정이다. 수치지도를 이용한 공간 데이타베이스 구축은 데이타 수집비용의 절감이라는 측면에서 점차 그 유용성이 증대되고 있지만 경계선 불일치, 객체의 불연속성과 같은 새로운 문제점들이 나타나게 되어 공간 데이타베이스를 생성하기 전에 수치지도에 대한 수정 및 편집작업이 수행되어야 한다.

이 논문에서는 클라이언트-서버 환경에서 여러 클라이언트가 수치지도에 대하여 동시에 수정작업을 하기 위해 영역 잠금(region lock)을 이용한 협동 트랜잭션 모델을 제시한다. 그리고 경계선 작업 시에 클라이언트간의 협동작업을 위한 메시지 전파 프로토콜을 제시한다.

키 워 드

GIS, DXF, 트랜잭션, 캐쉬, Lock, 합병, 타일, 작업공간

* 부산대학교 GIS학과 석사과정

** 부산대학교 컴퓨터공학과 박사과정

*** 부산대학교 컴퓨터공학과 교수

1. 서 론

공간 데이타베이스의 구축은 지형정보시스템(GIS) 전체 구축비용의 많은 부분을 차지하며 막대한 시간과 비용이 소요되는 과정이다.[1] 따라서 최근에는 데이터 수집을 위한 인적, 물적 비용의 절감을 위하여 기존에 구축되어 있는 수치지도를 이용하여 공간 데이타베이스를 구축하는 경우가 늘어나고 있다. 대표적인 예로 국가지리정보체계(NGIS) 구축사업의 수치지도화사업에 의해 구축된 수치지도가 있다.[10]

일반 수치지도를 이용하여 특정 공간 데이타베이스를 구축하기 위해서는 일반적으로 많은 수정과 편집 작업이 이루어진다. 특히 대형 공간 데이타베이스를 생성하기 위하여 다량의 수치지도를 이용하는 경우에 수치지도간의 지형객체 경계선 불일치(Boundary Disconnected)와 엔티티간의 불연속성을 해결하여 하나의 seamless map을 만들어 주어야 할 필요가 있다. 기존에는 이를 해결하기 위하여 각각의 수치지도에 대하여 개별적으로 수정작업을 한 후에 반복적으로 합병하는 방법을 이용하였다.

그러나 개별 작업과 반복 합병을 이용한 수정은 다량의 수치지도에 대하여 작업하는 경우에 데이타의 물리적인 크기가 커지기 때문에 작업의 생산성과 작업 효율이 극히 나쁘게 된다. 따

라서 다량의 수치지도에 대한 직접적인 합병작업 없이 여러 명의 편집자가 동시에 합병작업을 할 수 있는 알고리즘의 개발이 필요하다.

대형 수치지도에 대하여 여러 명의 편집자가 동시에 수정작업을 하는 경우에 작업의 선후에 따라 앞선 작업 내용을 잃어버리는 경우(lost work)가 발생할 수 있다. 또한 경계일치 혹은 합병의 작업 대상이 되는 경계선에서 만나는 두 엔티티에 대해 해당 클라이언트들이 서로 교차해서 잠금을 설정하는 경우 상대방의 잠금이 해제될 때까지 기다리는 deadlock이 발생할 수 있다. 이러한 문제점들을 해결하기 위해 이 논문에서는 편집자 상호간의 메시지 통신을 통한 협동 트랜잭션 모델을 제안한다.

협동 트랜잭션 모델은 클라이언트의 작업공간(Work Space)에 대한 잠금 설정을 위해 영역 잠금[9]을 도입한다. 그러나 클라이언트간의 작업경계 부분에서 서로 만나는 엔티티들에 대한 수정작업은 영역 잠금만으로는 한계가 발생한다. 수정하고자하는 엔티티를 완전히 포함하는 영역에 대해 영역 잠금을 설정해야 하므로 다른 클라이언트와의 불필요한 메시지 통신이 발생할 수 있기 때문이다. 그러므로 경계선에서의 수정작업은 관련된 클라이언트간의 메시지 통신을 통해 작업이 수행될 수 있도록 하는 통신 프로토콜이 필요하다. 이 프로토콜을 통해 경계선에서의 작업은 관련 클라이언트간의 합의에 의해 작업을 수행함으로써 작업의 동시성을 높일 수 있게 된다. 이 논문에서는 이러한 협동 작업을 위한 메시지 전파 프로토콜을 제시한다.

이 논문의 구성은 다음과 같다. 2장에서는 이 논문과 관련된 연구들을 기술하고, 3장에서는 영역 잠금(Region Lock)에 대해 기술하며, 협동 트랜잭션 프로토콜을 4장에서 기술하고, 5장에서는 협동 트랜잭션 프로토콜의 예를 보이고 6장에서 결론 및 향후연구에 대해 기술한다.

2. 관련연구

수치지도를 이용해서 공간 데이타베이스를 구축하는 기존의 상용 프로그램들이 이미 개발되어 실제 산업현장에서 사용되고 있다. 그러나 기존의 프로그램들은 수치지도 한 장에 대한 작업을 한사람의 작업자가 작업할 수밖에 없도록 되어 있기 때문에 다수의 작업자가 동시에 작업을 하고자 할 경우에는 각각의 작업자가 따로 작업을 한 뒤 또 다른 작업자가 이들을 모아서 합병하는 작업을 해야만 한다. 이러한 작업방식은 인적, 물적, 시간적인 비용의 증대를 초래하게 되는 비효율적인 방식이다.

[2]에서는 클라이언트-서버 환경에서 클라이언트 캐쉬의 일관성 유지와 변경 전파에 대한 알고리즘을 제안하고 있다. 특히, 다수의 클라이언트가 중복된 영역에 대한 수정 작업을 할 경우의 트랜잭션 모델을 위해 [9]에서 정의한 영역 잠금에 기반한 CS(Cached Shared)-잠금, CX(Cached eXclusive)-잠금 등의 확장된 잠금 기법을 제안하고 있다. 그러나, 작업공간 간의 경계선에서 만나는 엔티티들에 대해서는 영역 잠금 방식으로는 불필요한 영역까지 잠금을 설정해야하는 불합리성이 있기 때문에 [2]에서 제시한 트랜잭션 모델을 적용할 수가 없다. 예를 들면, 도로 중심선에 대한 합병작업을 진행하고자 할 경우에 이미 다른 클라이언트들에 의해 모두 합병작업을 했다고 가정하면 자신의 영역에 포함된 도로 중심선 및 다른 클라이언트들에 의해 수정된 도로 중심선들을 모두 포함하는 영역 잠금을 설정해야만 하며, 다른 클라이언트의 작업 공간을 모두 포함하는 경우까지도 발생하게 된다. 따라서, 서로 관련성이 없는 클라이언트끼리 불필요한 메시지 통신을 하게 되며 그로 인한 오버헤드가 발생하게 된다.

[3]에서는 이 논문과 같이 클라이언트마다 독자적인 작업공간을 가지고 독자적인 객체 변경이 가능하며 변경 전의 객체를 중복 저장하고

있는 다른 클라이언트에게 변경을 전파를 하는 알고리즘으로 "두단계 델타 합병(two-phase delta-merge)"을 제안하고 있다. 그러나 이 알고리즘은 클라이언트가 객체를 수정할 때마다 변경 전파를 하기 때문에 잦은 객체 수정이 발생할 경우에는 메시지 과부하 현상이 나타나며 자신의 작업공간의 경계선과는 만나지만 자신의 작업공간 외부에 존재하는 객체에 대해서는 수정 권한을 가질 수 없다는 한계가 있다.

[5]에서는 긴 트랜잭션일 경우 널리 쓰이고 있는 체크아웃(check-out) 모델에 관해 기술하고 있다. 이 모델에서는 작업자가 임의의 영역에 대한 객체들을 서버의 데이타베이스로부터 자신의 저장소에 복사한 뒤 작업이 완료된 객체에 대해 서버의 데이타베이스로 체크인하는 방법을 사용한다. 그러나 이 방법은 기본적으로 잠금을 사용하기 때문에 긴 시간 대기해야만 하므로 이 논문에서와 같은 협동 트랜잭션을 수행하기에는 적합하지 않다.

3. 영역 잠금(Region Lock)

이 논문에서는 사용자의 작업공간을 정의하기 위하여 [9]에서 정의한 영역 잠금(region lock)을 사용한다. 영역 잠금은 잠금의 단위(granularity)를 트랜잭션이 작업할 영역으로 줄여서 동시성을 높이고자 하는 기법으로서 READ 잠금을 허용하는 weak-SIX (shared intention exclusive)잠금이다. 영역 잠금은 기존의 잠금 모드인 SIX(shared intension exclusive) 잠금 모드와 유사하지만 다른 트랜잭션의 READ를 허용한다. 지도 수정이 그래픽 화면을 보면서 대화식으로 이루어진다는 특성을 가지고 있기 때문에 다른 클라이언트들이 자신의 작업공간이 아닌 다른 작업공간에 대한 참조를 요청할 수도 있기 때문이다.

이 논문에서 각 클라이언트는 그림2에서 보듯이 자신의 작업공간(Work Space)에 대해서 영

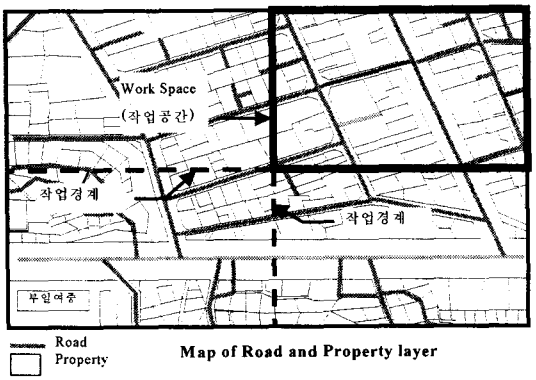


그림 2. 영역 잠금의 예

역 잠금을 설정한다. 그리고 각 클라이언트의 작업공간의 최외곽 경계선들과 위상학적으로 'Meet'의 관계에 있는 엔티티들에 대한 수정작업 시에는 해당 엔티티가 'Meet'하는 경계선을 사이에 둔 인접한 작업공간을 수정하는 다른 클라이언트에게 작업 확인 메시지를 보내어 작업을 공유할 수 있도록 한다. 그럼으로써 영역 잠금만으로는 해결할 수 없는 경계선에 'Meet'하는 엔티티들에 대한 수정작업을 이웃한 클라이언트와 함께 동시에 진행할 수 있게 한다.

4. 협동 트랜잭션 프로토콜

4.1 경계선 작업

이 논문에서 사용하는 경계선은 각 클라이언트가 작업의 권한을 가진 작업영역을 둘러싸는 네 개의 최외곽 경계선이다. 그리고 경계선 작업은 인접한 다른 클라이언트와의 경계선에 대해 'Meet' 관계에 있는 엔티티들에 대한 클라이언트의 수정작업이다. 그림3은 클라이언트 A와 B의 경계선 작업을 보여 준다.

경계선 작업을 동시에 여러 클라이언트가 수행하는 경우에 한 클라이언트에서 일방적으로 작업을 진행하는 것이 아니라 관련이 있는 클라이언트들 간의 협동 트랜잭션을 통하여 수정해

야 할 필요가 있다.

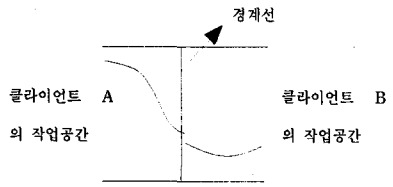


그림 3. 경계선 작업

그림4는 두 클라이언트가 협동 트랜잭션을 통해 경계선 작업을 수행하는 것을 보여준다.

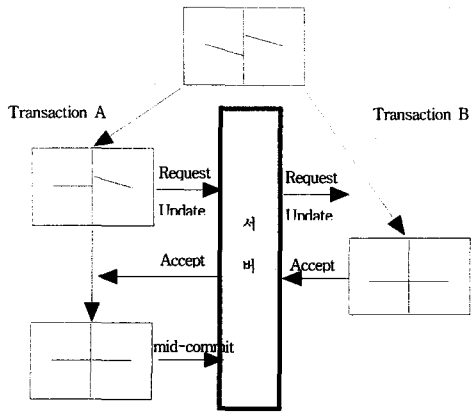


그림 4. 경계선에서 협동 트랜잭션의 예

서버는 협동 트랜잭션을 위해 경계선 작업과 관련된 클라이언트들의 정보를 유지하고 있어야 한다. 서버는 작업테이블(Working Table)이라는 정보저장소를 생성하여 클라이언트들의 작업공간에 대한 MBR(Maximum Boundary Rectangle)정보와 클라이언트에 대한 연결정보를 저장한다. 클라이언트로부터 협동 트랜잭션 요청 메시지가 오면 서버는 작업테이블을 참조하여 관련 클라이언트에게 협동 트랜잭션 요청 메시지를 전파한다.

4.2 협동 트랜잭션 모델

그림 5에서는 영역 잠금을 이용한 클라이언트 수정 트랜잭션 모델을 보여 준다. 클라이언트 트랜잭션 T는 영역 잠금을 설정한 후 서버에 통보한다(set-region-lock). 그리고 영역 잠금된 엔티티들 중에서 수정하고자하는 하나 이상의 엔티티에 대해 WRITE 잠금을 설정하고 서버에 통보한다(set-X-lock). WRITE 잠금을 설정한 후 클라이언트는 엔티티들에 대한 수정작업을 수행하며, 수정 완료 후 변경한 결과를 서버에 전파한다(mid-commit).

T

```
start transaction T
set-region-lock
    set-X-lock
        (Update)
        (Request Update) ->경계선 작업
        :
    mid-commit
    release-X-lock
    set-X-lock
        (Update)
        (Request Update) ->경계선 작업
        :
    mid-commit
    release-X-lock
release-region-lock
commit T
```

그림 5. 트랜잭션 모델

이때 경계선에 'Meet'한 엔티티에 대한 수정 작업일 경우에는 서버에게 작업허가를 요청한다(request-update). 서버는 작업허가 요청을 관련 클라이언트에게 전파하여 작업허가 확인 메시지를 작업을 요청한 클라이언트에게 보낸다. 작업허가 메시지를 받은 클라이언트는 변경한 결과를 서버에 전파한다(mid-commit). 서버에 변경 전파가 완료되면 해당 엔티티들에 대한 WRITE 잠금을 해제하며(release-X-lock), 다음에 수정하고자 하는 엔티티들에 대해 WRITE 잠금을 설정하고 동일한 방법으로 진행한다.

그림 6에서는 협동 트랜잭션이 필요한 경계선 작업의 경우를 보인다. 두 작업공간에 포함된 엔티티 a1, a2, b1, b2에서 a1과 b1, a2와 b2는

서로 경계선 일치 혹은 합병되어야 한다.

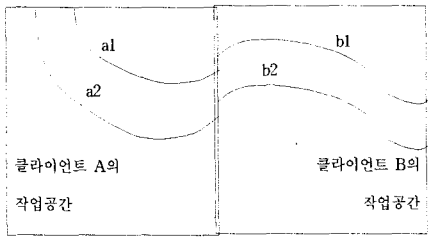
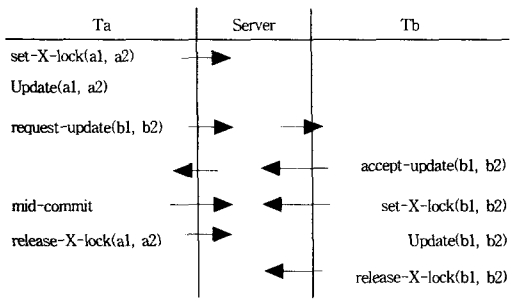


그림6. 협동 트랜잭션이 필요한 경우

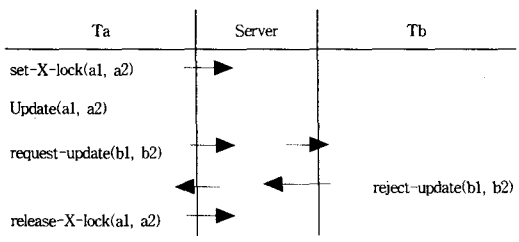
경계선 일치와 합병의 경우에 서로 다른 프로토콜이 적용되므로 각각에 대해서 기술한다.

-경계선 일치의 경우

먼저 클라이언트 A가 자신의 작업공간에 포함된 엔티티 a1, a2를 수정함으로써 b1, b2와 일치시키는 경우에는 a1, a2에 대해서만 WRITE 잠금을 설정한 뒤 작업을 수행하면 되므로 클라이언트 B와의 협동 트랜잭션이 필요없다. 그러나 b1, b2가 수정되어야 하거나 a1, a2, b1, b2가 모두 동시에 수정되어야 할 경우에는 클라이언트 B와의 협동 트랜잭션이 필요하다. b1과 b2에 대해서는 클라이언트 B가 수정할 수 있는 권한을 가지고 있기 때문이다. 클라이언트 A의 트랜잭션을 Ta라고 하고 클라이언트 B의 트랜잭션을 Tb라고 했을 때의 협동 트랜잭션 프로토콜은 그림7과 같다.



(가) Tb가 Accept한 경우



(나) Tb가 Reject한 경우

그림 7. 협동 트랜잭션 프로토콜(경계선 일치)

먼저 Ta가 서버로부터 경계선 작업 대상 엔티티에 대한 WRITE 잠금을 얻은 뒤 수정작업을 수행한 후 서버에게 협동 트랜잭션을 요구하는 메시지를 보낸다. 서버는 작업공간 정보를 참조하여 관련 클라이언트에게 협동 트랜잭션을 요청하는 메시지를 보낸다. Tb가 서버의 요청을 accept한 경우에 Tb는 요청 메시지에 포함된 수정 엔티티에 대한 정보를 이용하여 캐시를 수정한 뒤 자신의 트랜잭션을 계속 수행하고, 서버는 accept 메시지를 Ta에게 전파한다. 서버의 데이타에 대한 수정작업은 Ta의 mid-commit 연산에 의해 수행된다. 이때 트랜잭션 T는 다른 트랜잭션의 요청을 받음으로써 이루어지는 수정작업에 대해서는 mid-commit을 수행하지 않는다고 정의한다.

-합병의 경우

이 경우에는 경계선에서 'Meet'하는 두 엔티티가 새로운 하나의 엔티티로 재작성되는 경우이므로 어떤 경우라도 Ta와 Tb의 합의가 이루어져야 하는 트랜잭션이다. 합병의 경우 협동 트랜잭션 프로토콜은 새로운 엔티티가 생성된다는 점 외에는 그림6과 동일하므로 생략한다.

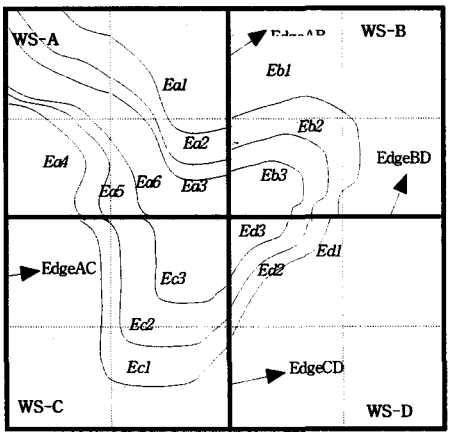
클라이언트가 서버에게 혹은 서버가 관련 클라이언트에게 보내는 경계선 작업 요청 메시지는 다음과 같은 내용을 포함한다.

-수정 전, 후 Ta의 엔티티 기하정보
-수정 전 Tb의 엔티티 기하정보
-수정작업타입(Edge Match, Merge)
-경계선 수정 기준점 좌표

5. 협동 트랜잭션의 예

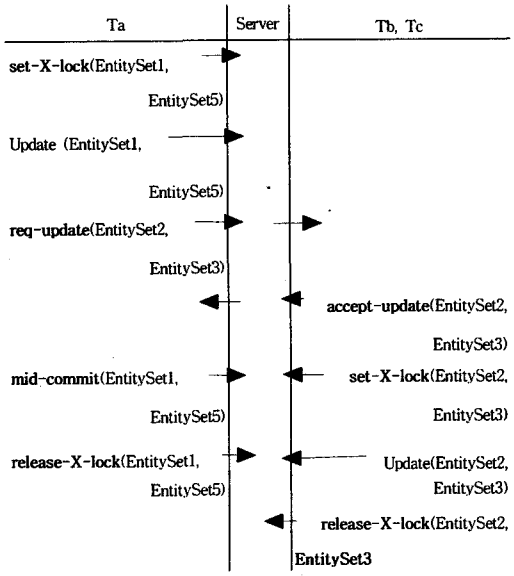
이 장에서는 협동 트랜잭션 프로토콜의 예제를 소개한다. 그림8에서 각각의 작업공간은 네 개의 타일로 구성되고, 네 개의 작업공간이 최종 결과지도를 구성한다. 따라서 이 경우에 전체 16매의 수치지도로 이루어진다. 네 개의 작업공간에 걸쳐있는 등고선을 하나의 연결된 엔티티로 생성하고자 한다. 이를 위해 우선 클라이언트 A가 클라이언트 B 및 클라이언트 C와 협동 트랜잭션을 수행하는 경우를 가정한다.

우선 클라이언트 A는 EdgeAB와 만나는 등고선 엔티티 집합 EntitySet1(Ea1, Ea2, Ea3)과 EdgeAC와 만나는 등고선 엔티티 집합 EntityeSet5(Ea4, Ea5, Ea6)에 대해 각각 WRITE 잠금을 서버에게 요청한다. 각각의 엔티티 집합에 대한 WRITE 잠금이 설정된 후 클라이언트 A는 각 엔티티 집합에 대한 수정작업을 수행한다. 작업이 완료된 후 클라이언트 A



WS-A : 클라이언트 A의 작업공간
WS-B : 클라이언트 B의 작업공간
WS-C : 클라이언트 C의 작업공간
WS-D : 클라이언트 D의 작업공간
EdgeAB : WS-A와 WS-B의 경계선
EdgeAC : WS-A와 WS-C의 경계선
EdgeBD : WS-B와 WS-D의 경계선
EdgeCD : WS-C와 WS-D의 경계선

그림8. 경계선에서 등고선 합병 예



EntitySet1 : Ea1, Ea2, Ea3
EntitySet2 : Eb1, Eb2, Eb3
EntitySet3 : Ec1, Ec2, Ec3
EntitySet4 : Ed1, Ed2, Ed4
EntitySet5 : Ea4, Ea5, Ea6

그림9. 협동 트랜잭션 프로토콜의 예 (accept-update 되는 경우)

는 수정된 작업에 대한 협동 트랜잭션을 서버에게 요청한다. 서버는 클라이언트 A가 요청한 경계선 작업의 관련 클라이언트인 클라이언트 B와 C에게 협동 트랜잭션을 요청한다. 서버는 두 클라이언트로부터 accept-update가 전달되면 클라이언트 A에게 accept-update를 전달하고 두 클라이언트 중 한 클라이언트라도 reject-update가 전달되면 reject-update를 클라이언트 A에게 전달한다. 클라이언트 A는 서버로부터 accept-update 메시지가 넘겨져 올 경우에 한해서 서버에게 mid-commit을 요청하며 그렇지 않은 경우에는 EntitySet1과 EntitySet5에 대한 수정작업을 취소하고 서버에게 WRITE 잠금 해제를 요청한다. 그림9에서는 accept-update인 경우의 협동 트랜잭션 프로토콜의 예를 보이고 그림10에서는 reject-update인 경우의 프로토콜 예를 보이고 있다.

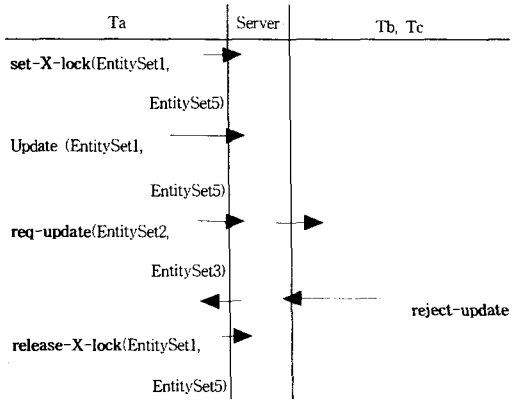


그림10. 협동 트랜잭션 프로토콜의 예
(reject-update 되는 경우)

6. 결론 및 향후 연구

이 논문에서는 클라이언트-서버 환경 하에서, 수치지도를 이용한 공간 데이타베이스 구축을 위한 협동작업에서 다수의 작업자가 동시에 작업함으로 인해 발생하는 작업자간의 작업 충돌을 제어하기 위해 영역 잠금을 설정하고, 협동작업을 위한 트랜잭션 모델 및 통신 프로토콜을 제시한다.

향후 연구로는 제시된 트랜잭션 모델 및 통신 프로토콜에 대한 알고리즘을 설계하고 이에 대한 구현을 통하여 작업의 동시성 증가도를 측정해 봄으로써 수치지도를 이용한 공간 데이타베이스의 구축에 있어서 기존의 방법과의 비교 및 검증이 필요하다.

참고문헌

[1] Christopher B. Jones, "Geographic Information Systems and Computer Cartography", 1997.

[2] 신영상, 최진오, 홍봉희, "클라이언트-서버 환경에서 캐쉬 공간 데이터의 변경 전파", 한국정보과학회 '99 봄학술발표논문집(B), 제 26권 1호, pp 86-88.

[3] AmSuk Oh, JinOh Choi, BongHee Hong, "An Incremental Update Propagation Scheme for a Cooperative Transaction Model", Preceedings of the International Conference on DEXA, 1996.

[4] Korth H. F., Speegle, G. D., "Long-Duration Transactions in Software Design Projects", Proceedings of the International Conference on Data Engineering, pp.586-574, 1990.

[5] Korth H. F., "On Long-Duration CAD Transaction", ACM Transactions on Information Systems, 1987.

[6] Korth H. F., "A Model of CAD Transactions", Proceedings of VLDB, 1985

[7] Edward T. Blair, "Version Management in the Work Order Processing Environment", AM/FM International, 1994

[8] Xin Chang Zhang, "Geometric Feature-based Edge-Matching", Proceedings of the 3rd Internatioanl Conf. on GeoComputation, 1998

[9] 최진오, 홍봉희, "분산 GIS 데이타베이스에서 중복 데이터의 변경전파", '98 동계 데이타베이스 학술대회 논문집 제 14권 1호, p27-32

[10] 국립지리원, http://www.ngi.go.kr

이 상 현

1992년 부산대학교 기계공학과(학사)
1994년~현재 아키정보기술주식회사
1998년~현재 부산대학교 지형정보협동과정(석사)
관심분야 : GIS, 공간 데이타베이스

김 동 현

1995년 부산대학교 컴퓨터공학과(학사)
1997년 부산대학교 컴퓨터공학과(석사)
1998년~현재 부산대학교 컴퓨터공학과(박사)
관심분야 : 개방형 GIS, 분산공간데이타베이스, 공학 데이타베이스

홍 봉 희

1982년 서울대학교 전자계산기공학과(학사)
1984년 서울대학교 전자계산기공학과(석사)
1988년 서울대학교 전자계산기공학과(박사)
현재 부산대학교 공과대학 컴퓨터공학과 정교수
관심분야 : 개방형 GIS, 병렬공간데이타베이스, 분산 공간데이타베이스