이동 이질 멀티데이타베이스 시스템을 위한 이동 유연 트랜잭션의 실행 제어 알고리즘

구 경 이† · 김 유 성††

요 약

휴대용 컴퓨터와 무선 통신 기술의 발달로 이동 컴퓨팅 환경이 점차 확산되어가고 있다. 이동 사용자는 기존의 이질 분산 환경이 이동 호스트를 지원할 수 있도록 구성된 이동 이질 컴퓨팅 환경에 이동 트랜잭션을 제기함으로써 원하는 정보를 얻게 된다. 따라서, 이동 트랜잭션은 이질 멀티데이타베이스 시스템을 위한 트랜잭션의 특성과 이동 컴퓨팅 환경을 위한 트랜잭션의 특성을 함께 지녀야 한다.

본 논문에서 제안된 이동 유연 트랜잭션 모델은 멀티데이타베이스 시스템을 위해 제안된 유연 트랜잭션 모델을 이동 컴퓨팅 환경에 적합하도록 확장하였다. 또한 이동 컴퓨팅 환경에서 각 사이트별로 자치적으로 운용되는 데이타베이스 시스템을 상호 협력이 가능하도록 통합하여 이동 이질 멀티데이타베이스 시스템으로 발전시키며, 이를 통하여 이동 사용자의 요구사항을 지원하도록 이동 유연 트랜잭션 모델의 실행 제어 알고리즘을 정의하였다. 확장된 이동 유연 트랜잭션 모델은 기존의 제안된 이동 트랜잭션 모델들에 비하여 유연성과 병렬성의 지원, 위치 기반 실행, 핸드오버의 효율적인 처리 및 부분 실행의 결과 허용 등의 이동성을 지원함으로써 이동 호스트의 이동, 전력적 제약 조건, 낮은 대역폭과 잦은 접속 단절로 인해 고장이 발생하기 쉬운 이동 컴퓨팅 환경에서 트랜잭션 처리율 측면의 효율성을 갖는다.

An Execution Control Algorithm for Mobile Flex Transactions in Mobile Heterogeneous Multidatabase Systems

Kyong-I Ku† · Yoo-Sung Kim††

ABSTRACT

As the technical advances in portable computers and wireless communication technologies, mobile computing environment has been rapidly expanded. The mobile users on mobile host can access information via wireless communication from the distributed heterogeneous multidatabase system in which pre-existing independent local information systems are integrated into one logical system to support mobile applications. Hence, mobile transaction model should include not only the features for heterogeneous multidatabase systems but also the ones for mobile computing environment.

In this paper, we proposed a mobile flex transaction model which extends the flexible transaction model that previously proposed for heterogeneous multidatabase systems is extended to support the requirements of mobile heterogeneous multidatabase systems. We also presented the execution control mechanism of the mobile flex transaction model. The proposed mobile flex transaction model allows the definition of location-dependent subtransactions, the effective support of hand-over, and the flexibility of transaction executions. Hence, the proposed mobile flex transaction model can be suit to mobile heterogeneous multidatabase systems that have low power capability, low bandwidth, and high communication failure possibility.

※ 이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음.
† 준 회 원 : 인하대학교 대학원 전자계산공학과
†† 종신회원 : 인하대학교 전자계산공학과 교수
논문접수 : 1999년 7월 12일, 심사완료 : 1999년 10월 5일

1. 서 론

최근의 이동 컴퓨터 하드웨어 기술의 발달과 무선 통신 기술의 발달에 따라 이동 정보 시스템의 발전이 지속적으로 이루어지고 있다. 사용자들은 이동 중에도 이동 호스트를 이용하여 무선 통신으로 원거리 컴퓨터에 접속해서 필요한 정보를 구한다. 이동 컴퓨터는 임의의 시간에 임의의 장소로 이동할 수 있는 장점을 가지고 있지만, 여러 가지 단점도 가지고 있다. 우선, 이동 호스트들은 휴대 장치에서 이용되는 충전지를 사용하기 때문에 무한히 사용 가능한 고정형 호스트에 비해 전력 사용이 매우 제한적이다. 두 번째, 무선 통신에서는 통신 대역폭이 제한되어 있기 때문에 송수신되는 데이타 양에 제약이 가해지게 된다. 세 번째, 전송 매체에 대한 외부 간섭(기후, 전자 간섭, 사용자의 일방적인 단절)과 이동 호스트들의 빈번한 이동은 잦은 접속 단절을 야기해 전송되는 데이타에 대한 파손 및 분실을 증가시킬 수 있다[1]. 그러나 이러한 치명적인 약점들에도 불구하고 이동 컴퓨팅을 가능하게 하기 위해 위에서 열거한 제약 조건을 최소화하기 위한 연구로서 데이타 방송, 캐쉬 관리, 트랜잭션 관리 등에 대한 연구가 활발히 진행되고 있다[2, 3].

이동 사용자들은 전자 메일이나, 주식 또는 여행 중에 필요한 여러 가지 정보들을 이동 호스트를 이용하여 얻는다. 이동 호스트가 무선 통신을 통하여 접속하는 정보 시스템은 이동 이질 멀티데이타베이스 시스템(mobile heterogeneous multidatabase system)으로써, 이동 트랜잭션(mobile transaction)을 사용하여 액세스하게 된다[4]. 이동 이질 멀티데이타베이스 시스템은 자치성을 보장하면서 독립적으로 운용되는 지역 데이타베이스 시스템들을 상호 협력적으로 운용될 수 있도록 확장한 형태로서 이동 호스트가 이동 중에 액세스하는 데이타베이스 시스템이다. 따라서, 이동 트랜잭션은 기본적으로 멀티데이타베이스 시스템을 위한 트랜잭션 모델의 특성을 포함하면서, 동시에 이동 컴퓨팅 환경의 특성을 포함하고 있어야 한다. 지금까지 이동 컴퓨팅 환경을 위해 제안된 이동 트랜잭션 모델들은 존재하지만, 이질 멀티데이타베이스 시스템을 위한 트랜잭션 특성과 이동 컴퓨팅 환경을 위한 트랜잭션 특성을 함께 갖는 이동 트랜잭션 모델은 존재하지 않는다. 본 논문에서는 이동 이질 멀티데이타베이스 시스템을 위한 트랜잭션 모델로서 이동 유연 트랜잭션(Mobile Flex Transaction) 모델과 실행 제어 알고리즘을 제안한다.

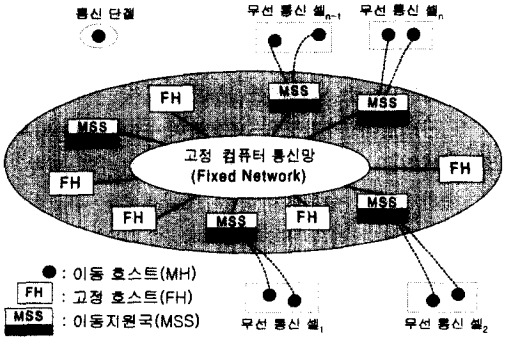
이동 유연 트랜잭션 모델(Mobile Flex Transaction Model)은 이질 멀티데이타베이스 시스템을 위해 제안된 유연 트랜잭션 모델[5]에 이동 컴퓨팅 환경의 특성이 반영되도록 확장한 이동 트랜잭션 모델이다. 이동 유연 트랜잭션 모델은 기본적으로 유연 트랜잭션 모델을 바탕으로 하여 확장되었으므로 기존의 이동 트랜잭션 모델과 다르게 이질 멀티데이타베이스 시스템을 위한 트랜잭션의 특성을 지원한다. 따라서 이동 유연 트랜잭션 모델은 지역 데이타베이스 시스템의 자치성을 보장할 뿐만 아니라, 데이타 일관성을 유지한다. 그리고 여러 개의 최종 실행 목표를 지닐 수 있는 유연성(flexibility) 및 부트랜잭션간의 병렬적 실행을 통하여 트랜잭션의 처리율을 높일 수 있다. 이동 유연 트랜잭션 모델은 또한 유연 트랜잭션 모델과는 다르게 이동 컴퓨팅 환경을 위한 트랜잭션 특성을 지원한다. 이동 유연 트랜잭션 모델은 이동 컴퓨팅에서 증대되고 있는 위치 기반 실행 요구, 핸드오버(hand-over)의 효율적인 처리, 부분 실행의 결과 허용을 통하여 이동 사용자의 이동성(mobility)을 지원한다. 즉 이동 유연 트랜잭션 모델은 트랜잭션 처리의 유연성 및 병렬 수행, 이동성 지원을 통하여 데이타의 일치성을 유지하면서 동시성을 높임으로 인해 이동 호스트의 이동, 전력적 제약 조건, 낮은 대역폭과 잦은 접속 단절로 인해 고장(failure)이 발생하기 쉬운 이동 컴퓨팅 환경에서 트랜잭션 처리율 측면의 효율성 증대를 가져오게 된다.

본 논문의 구성은 다음과 같다. 2장에는 이동 컴퓨팅 환경하에서 이동 트랜잭션의 특성을 기술하고, 기존의 이동 트랜잭션 모델의 제약 사항을 기술한다. 3장에서는 이동 유연 트랜잭션 모델을 정형화하고, 확장된 의미론에 대해 기술한다. 4장에서는 이동 유연 트랜잭션 모델을 이동 컴퓨팅 환경하에서 실행되도록 실행 제어 알고리즘을 제안한다. 5장에서는 유연 트랜잭션 모델 및 기존의 이동 트랜잭션 모델과의 특성 비교를 통하여 이동 유연 트랜잭션 모델이 이동 컴퓨팅 환경에 적합함을 설명한다. 마지막으로 6장에서 본 논문의 결론 및 향후 연구 방향에 대해 기술한다.

2. 이동 컴퓨팅 환경을 위한 트랜잭션 모델

2.1 이동 트랜잭션 모델의 특성

이동 컴퓨팅 환경이란 기존의 분산 컴퓨팅 환경이 이동 호스트들을 지원할 수 있도록 구성된 환경을 의미한다. 현재 널리 이용되고 있는 이동 컴퓨팅 환경은 (그림 1)과 같이 이동 호스트(Mobile Host: MH), 이동지원국(Mobile Support Station: MSS), 그리고 고정 호스트(Fixed Host: FH)들로 구성되어 있다[6].



(그림 1) 이동 컴퓨팅 환경

이동 호스트는 노트북 컴퓨터 혹은 이동 컴퓨팅 환경에 적합하게 개발된 휴대용 컴퓨터를 사용한다. 이러한 이동 호스트는 기본적으로 이동 사용을 위해서 충전지 전원을 사용하기 때문에 사용 가능 시간이 제약된다. 이동 호스트는 이동지원국을 통하여 고정 컴퓨터 통신망에 접속할 수 있다. 즉, 이동지원국은 이동 호스트를 고정 컴퓨터 통신망에 접속시키기 위한 인터페이스 역할을 담당한다. 하나의 이동지원국이 포괄하는 통신 지역을 무선 통신 셀(cell)이라 하며, 해당 이동지원국은 자신 셀 안에 있는 모든 이동 호스트들과의 통신을 관장한다. 이동지원국은 관장하고 있는 셀 내에 위치한 이동 호스트의 관련 정보(소속 정보, 보안 정보, 위치 정보 등)를 자체 관리하고 있으며, 관련 정보를 이용하여 이동 호스트 사용자의 정보 액세스를 조정한다. 따라서 이동 호스트가 한 이동지원국이 관리하고 있는 셀에서 다른 이동지원국으로 이동시 관련 정보를 새로운 이동지원국으로 옮겨서 이동 호스트 사용자에게 아무런 부담 없이 정보 시스템을 이용할 수 있도록 핸드오버(hand-over)작업을 수행한다. 고정 호스트는 고정 사용자 및 이동 사용자에게 제공할 컴퓨팅 능력 및 정보를 관리한다. 고정 호스트에는 메인프레임, 워크스테이션 및 개인용 컴퓨터 등을 용도 및 처리 능력에 따라 설치하여 사용한다.

이동 컴퓨팅 환경에서 이동 호스트와 이동지원국간에는 무선 컴퓨터 통신으로 정보를 교환하며, 이동지원국과 고정 호스트사이에는 고정 컴퓨터 통신으로 정보를 교환한다. 이동 사용자 및 고정 사용자에게 정보를 제공하기 위해 이동지원국 및 고정 호스트에는 지역 데이타베이스 시스템이 운용된다. 지역 데이타베이스 시스템은 서로 독립적으로 운용되도록 설계, 구현되었기 때문에 데이타 모델, 데이타베이스 언어, 트랜잭션 모델, 트랜잭션 관리 기법들간의 이질성이 존재한다. 따라서 이동 컴퓨팅 환경에서 이동 사용자를 지원하기 위한 정보 시스템은 기존의 분산 데이타베이스 시스템의 개념이 확장된 이동 이질 멀티데이타베이스 시스템의 형태를 갖는다[3, 7]. 이동 이질 멀티데이타베이스 시스템의 데이타베이스는 이동지원국 및 고정 호스트상의 지역 데이타베이스 시스템에서 독립적으로 관리되며, 이동 사용자의 요구에 따라 각 지역 데이타베이스 시스템에서 관리하는 정보를 이동 사용자에게 전달한다. 이동 컴퓨팅 환경하에서 이동 사용자가 이동 이질 멀티데이타베이스 시스템을 액세스하기 위해 제기하는 트랜잭션을 이동 트랜잭션(mobile transaction)이라고 한다[4].

기존의 데이타베이스 시스템을 위한 트랜잭션과는 달리 이동 트랜잭션은 이동성(mobility)과 이질성(heterogeneity)이 결합된 이동 이질 멀티데이타베이스 시스템 형태의 특성을 함께 지원해야 한다. 즉, 여러 자치적인 이질 지역 데이타베이스 시스템들을 데이타베이스 시스템의 자치성과 데이타 일관성을 유지하면서 트랜잭션 단위의 액세스가 가능하도록 이동 트랜잭션이 지원해야 한다. 또한, 지리 정보를 요구하거나 위치 변화에 적응하는 응용 시스템의 구성을 위해 이동 호스트의 이동에 따라 실행 결과가 다른 위치 기반적(location-dependent)인 정보 제공이 필요하다[8]. 그리고 이동 호스트의 사용 시간을 줄이고 낮은 통신 대역폭을 효율적으로 사용하기 위해 핸드오버의 발생시 이동 트랜잭션을 구성하는 부트랜잭션의 특성에 따라 실행을 결정할 수 있는 구조 및 트랜잭션내의 부분 실행의 결과를 허용할 수 있어야 한다[9].

이동 이질 멀티데이타베이스 시스템을 위해 이동 트랜잭션이 가져야 할 특성을 요약하면 다음과 같이 정

리될 수 있다.

① 이질 지역 데이타베이스 시스템의 액세스를 지원하는 트랜잭션
② 지역 데이타베이스 시스템의 자치성을 허용하는 트랜잭션
③ 지역 데이타베이스 시스템의 데이타 일관성을 유지하는 트랜잭션
④ 이동 트랜잭션의 정의 및 실행 단계에서 유연성을 제공하는 트랜잭션
⑤ 위치 기반 실행을 지원하는 트랜잭션
⑥ 핸드오버를 지원하는 트랜잭션
⑦ 부분 실행의 결과를 허용하는 트랜잭션

2.2 기존에 제안된 이동 트랜잭션 모델

트랜잭션(transaction)은 전통적으로 사용자가 데이타베이스를 액세스하는 작업의 기본 단위이며, ACID(원자성, 일치성, 분리성, 영구성)속성을 만족시키도록 규정되었다[10]. 지금까지의 트랜잭션 구조나 개념은 주로 중앙 집중식 데이타베이스 시스템이나 분산 데이타베이스 시스템을 지원하기 위한 개념이다. 높은 신뢰성을 지닌 고정 컴퓨터 통신망에서 데이타베이스를 액세스하기 위해 제안된 기존의 트랜잭션 모델은 통신면에서 낮은 신뢰성을 갖는 이동 컴퓨팅 환경에는 적합하지 않다. 이동 컴퓨팅 환경하에서 트랜잭션은 통신 지연, 한정된 전력 사용, 잦은 접속 단절 때문에 오래 지속(long-lived)되는 특성을 지닌다. 이러한 특성을 지원하기 위해 제안된 이동 트랜잭션 모델로서 Reporting and Co-Transaction 모델[11], Kangaroo Transaction 모델[12], Clustering Transaction 모델[4], Semantic-Based Mobile Transaction 모델[13], 그리고 MDSTPM (Multidatabase Transaction Processing Manager)모델[14] 등이 있다.

기존에 제안된 이동 트랜잭션 모델들은 이동 컴퓨팅 환경에서 제약적인 트랜잭션의 ACID속성을 재정의하거나 완화시켰다[3]. 또한 고정 통신망과의 통신 지연을 줄이기 위해 이동 호스트의 이동에 따라 고정 통신망의 구성 요소들을 부분적 또는 전체적으로 재구성하는 트랜잭션 재배치(transaction relocation)를 지원하기도 한다[11, 12]. 그러나, 기존 이동 트랜잭션 모델들은 이질 지역 데이타베이스 시스템의 자치성과 데이타 일관성을 유지하면서 액세스가 가능하도록 하는 이질 멀티데이타베이스 시스템의 특성을 지니지 않는다[4, 11, 13]. 또한, 트랜잭션 실행의 유연성, 그리고 위치 기반 연산 및 핸드오버, 부분 실행의 결과를 허용하지 않는다[4, 11, 12, 13, 14]. 즉, 기존에 제안된 이동 트랜잭션 모델은 고장이 발생하기 쉬운 이동 이질 멀티데이타베이스 시스템 환경하에서 오래 지속되는 특성을 지닌 이동 트랜잭션의 처리에 있어서 효율성을 갖지 않는다.

3. 이동 유연 트랜잭션 모델

3.1 유연 트랜잭션 모델

이질 멀티데이타베이스 시스템은 사용자에게 필요한 정보를 제공하기 위해서 여러 지역의 독립적인 이질 지역 데이타베이스 시스템을 상호 협력이 가능하도록 통합한 데이타베이스 시스템이다. 유연 트랜잭션 모델(Flexible Transaction Model)은 이질 멀티데이타베이스 시스템을 위한 트랜잭션 모델로서 각 지역 데이타베이스 시스템의 자치성을 보장하며, 지역 데이타베이스의 일관성을 유지한다. 또한, 트랜잭션내의 일부 작업이 불완전하게 수행되더라도 트랜잭션을 완료할 수 있는 트랜잭션 처리의 유연성을 제공한다[5].

유연 트랜잭션 모델은 3가지 특성을 갖는다. 첫 번째, 유연 트랜잭션은 여러 개의 최종 실행 목표를 가질 수 있는 유연성을 제공한다. 사용자는 이질 멀티데이타베이스 시스템을 구성하는 지역 데이타베이스 시스템의 자치성을 보장하면서도 트랜잭션 실행 성공률을 높이기 위해 승인 가능한 실행 최종 상태로 여러 값을 정의할 수 있다. 즉, 함수적으로 동치인 부트랜잭션을 허용하고, 그 중 하나의 성공적인 수행을 통하여 유연 트랜잭션의 성공적인 완료가 가능하다. 이러한 유연성은 일부 부트랜잭션이 실패하더라도 대안적(alternative)인 부트랜잭션의 실행을 통하여 최종 실행 목표 중의 하나에 도달함으로써 실패에 대해 탄성을 갖게 한다.

두 번째, 한 부트랜잭션은 다른 부트랜잭션의 실행 상태나 산출값, 외부 조건을 기초로 하여 실행 시작, 재개, 완료를 결정하는 실행 종속성을 갖는다[16]. 이러한 실행 종속성은 유연 트랜잭션의 병렬적 실행시 정확성 판단을 입증하거나, 유연 트랜잭션 모델의 명세시 부트랜잭션간의 종속성(성공 종속성, 실패 종속성, 외부 종속성)을 정의함으로써 부트랜잭션의 실행을 위한 필요 조건을 구성하는데 이용된다.

세 번째, 유연 트랜잭션은 보상(compensability) 개념을 이용한다. 이질 멀티데이타베이스 시스템의 트랜잭션은 기본적으로 오래 지속되는 특성을 갖기 때문에 prepared-to-commit 상태에서 결정을 기다리는 부트

랜잭션에 의해 소유되는 로크들은 데이타의 유용성을 떨어뜨린다[15]. 유연 트랜잭션 모델은 각 사이트에서 수행되는 몇 개의 연산들을 모아놓은 작업의 논리적 단위로 부트랜잭션을 구성하며, 각각의 부트랜잭션이 보상 가능하거나 보상 불가능한 특성을 갖도록 한다[5]. 보상 가능한 부트랜잭션은 지역적인 완료가 가능하며, 이러한 완료 가능성은 데이타의 유용성을 증가시키고, 교착 상태의 가능성을 낮게 한다[17].

이러한 세 가지 특성을 갖는 유연 트랜잭션 모델은 이질 멀티데이타베이스 시스템을 위한 트랜잭션 모델 및 처리에 적합한 것으로 판명되었다[15].

3.2 이동 유연 트랜잭션 모델

3.2.1 이동 유연 트랜잭션 모델의 정형화

이동 컴퓨팅 환경하에서 이동 사용자가 액세스하는 이동 이질 멀티데이타베이스 시스템을 위한 이동 유연 트랜잭션 모델(Mobile Flex Transaction Model)의 이동 유연 트랜잭션 T는 <정의 1>과 같이 정형화된다.

〈정의 1〉 이동 유연 트랜잭션 T는 7개의 튜플 (M, S, F, Π, H, J, G)로 구성되며, 각각의 의미는 아래와 같다.

- M : 부트랜잭션의 집합 = $\{t_1,t_2, \ldots ,t_n\}$
- S : 부트랜잭션간의 성공 종속성 집합
- F : 부트랜잭션간의 실패 종속성 집합
- Π : 부트랜잭션의 외부 종속성 집합
- H : 부트랜잭션의 핸드오버 제어 규칙 집합
- J : 부트랜잭션의 승인되어질 수 있는 결합 규칙 집합
- G : 이동 유연 트랜잭션 T의 최종 실행 목표들의 집합($G \subseteq P(M)$)

M은 이동 유연 트랜잭션 T의 모든 부트랜잭션들의 집합이다. 이동 유연 트랜잭션 T를 구성하는 부트랜잭션이 보상 가능한 트랜잭션이면 C(Compensable)로 표시하고, 보상 불가능한 트랜잭션이면 NC(Non-Compensable)로 표시를 한다. 이때, 부트랜잭션 t_i는 C나 NC 형태를 가질 수 있다. S는 이동 유연 트랜잭션 T를 구성하는 부트랜잭션들간의 성공 종속성을 나타낸다. 성공 종속성은 부트랜잭션 t_i와 t_j사이에서 t_i가 성공적으로 수행된 후에 부트랜잭션 t_j가 실행될 수 있음을 의미하며, $t_i <_s t_j$로 표현한다. F는 이동 유연 트랜잭션 T를 구성하는 부트랜잭션들간의 실패 종속성을 나타낸다. 실패 종속성은 부트랜잭션 t_i와 t_j사이에서 t_i가 실패적으로 수행된 후에 부트랜잭션 t_j가 실행될 수 있음을 의미하며, $t_i <_f t_j$로 표현한다. Π는 이동 유연 트랜잭션 T를 구성하는 부트랜잭션이 갖는 외부 종속성을 나타낸다. 외부 종속성이란, 부트랜잭션 t_i가 외부 술어 조건이 참일 경우에만 실행이 가능함을 나타낸다. 외부 술어 조건은 시간, 비용, 그리고 이동 호스트의 위치 등이 될 수 있다. 외부 술어 조건에서 이동 호스트의 위치를 기술하는 경우 부트랜잭션은 위치 기반 연산을 갖는다. H는 핸드오버 제어 규칙으로서 핸드오버 발생시 이동 유연 트랜잭션 T내의 실행 중인 부트랜잭션 t_i에 대해 실행을 결정할 수 있도록 한다. T를 구성하는 부트랜잭션은 보상 가능(C)이나 보상 불가능(NC)의 특성과 더불어 위치 기반 특성을 갖는다. 이동 유연 트랜잭션 모델은 이러한 부트랜잭션의 속성을 고려하여, 핸드오버 제어 규칙의 값으로서 continue(t_i), restart(t_i), split_resume(t_i), split_restart(t_i) 사이에서 하나의 값을 기술한다. J는 승인되어질 수 있는 결합 규칙을 의미한다. 이동 호스트의 사용 시간을 줄이고 통신 비용을 낮추기 위해 분할되어 수행된 부트랜잭션의 실행 결과값은 분할되기 전 트랜잭션의 실행의 결과값과 동치이기 위해 분할된 부트랜잭션 t_i에 대해 결합 연산을 수행한다. 분할되어서 실행된 결과값들의 결합을 승인되어질 수 있는지 판별하기 위해 join(t_i), user(t_i) 사이에서 하나의 값을 기술한다. 마지막으로 G는 이동 유연 트랜잭션 T의 최종 실행 목표들의 집합을 의미한다. 최종 실행 목표는 이동 유연 트랜잭션 T를 구성하는 부트랜잭션들 중에서 이동 유연 트랜잭션 T의 요구사항을 만족시키는 최소 집합으로 구성된다. 유연 트랜잭션 모델에서 이동성을 지원하기 위해 확장된 의미는 3.2.2절에서 자세히 기술한다.

제안된 이동 유연 트랜잭션 모델에서 이동 유연 트랜잭션 T를 구성하는 각각의 부트랜잭션 t_i는 <정의 2>와 같은 네 가지의 실행 상태를 갖는다.

〈정의 2〉 n개의 부트랜잭션을 갖는 이동 유연 트랜잭션 $T = \{t_1,t_2, \ldots ,t_n\}$의 실행 상태 X는 n개의 튜플 $(x_1,x_2, \ldots ,x_n)$로 구성되며, 실행 상태 변수 x_i는 아래의 값 중 하나를 갖는다.

x_i = 'N': 부트랜잭션 t_i가 아직 실행이 제기되지 않은 상태

'E': 부트랜잭션 t_i가 동시적으로 실행 중인 상태
'S': 부트랜잭션 t_i가 성공적으로 수행을 끝난 상태
'F': 부트랜잭션 t_i가 실행을 실패하고 수행을 끝난 상태

이동 유연 트랜잭션 T가 n개의 부트랜잭션으로 이루어지면, 이동 유연 트랜잭션의 실행 상태 X는 n개의 튜플로 구성되며 $(x_1,x_2,\ldots,x_n)$로 표현된다. 부트랜잭션 t_i의 실행 상태 변수 x_i='N'이면 실행을 위해 아직 제기되지 않은 상태를 의미한다. 'E'이면 부트랜잭션 t_i가 다른 부트랜잭션들과 동시적으로 실행 중임을 의미한다. 'S'이면 부트랜잭션 t_i가 성공적으로 실행을 마침을 의미한다. 만약 t_i가 보상 가능한 경우에는 완료되어진 상태이며, 보상 불가능한 경우에 'S'의 의미는 prepared-to-commit 상태임을 나타낸다. 'F'이면 t_i가 보상 가능한 경우 실행이 철회되거나, 자신의 목적을 달성하지 못하고 보상(compensate)으로 종료됨을 의미하며, 보상 불가능한 경우 실행이 철회됨을 의미한다[5].

이동 유연 트랜잭션의 성공적인 실행이란, 실행 중에 실행 상태가 승인되어질 수 있는 상태의 값 중 하나의 값과 동일할 경우이며, 이동 유연 트랜잭션의 승인되어질 수 있는 상태 A는 <정의 3>과 같이 정의된다.

<정의 3> 이동 유연 트랜잭션 T = (M, S, F, Π, H, J, G)에서 G는 최종 실행 목표, X를 실행 상태라고 할 때, $\forall t_k \in g_i$, x_k='S'의 성질을 만족하는 $g_i \in G$의 값이 존재하는 경우 승인되어질 수 있는 상태 A라고 하며, T의 실행 상태 X가 A의 값 중에 하나에 도달하게 되면 이동 유연 트랜잭션의 실행을 성공적으로 종료되었다고 한다.

이동 유연 트랜잭션에서 G는 최종 실행 목표들의 집합을 의미하며, 이동 사용자의 요구 사항의 명세화에 따라 최종 실행 목표들의 집합을 구성한다. 이 때 G에 속하는 원소값 g_i를 구성하는 모든 부트랜잭션들의 실행 상태 변수값이 x_k='S'이면 승인되어질 수 있는 상태 A라고 한다. G의 원소 값 g_i를 구성하지 않은 부트랜잭션들의 실행 상태 변수값은 'S'이외의 값('N','E','F')을 가질 수 있다. 이동 유연 트랜잭션의 실행은 부트랜잭션을 병렬적으로 실행하며, 부트랜잭션 실행 후 실행 상태 변수의 값을 'E'에서 성공적인 실행의 경우는 'S'로, 실패적인 실행의 경우는 'F'로 변화시키게 된다. 실행 도중 실행 상태 변화로 A에 도달하게 되면, 이동 유연 트랜잭션은 실행이 성공적으로 수행되었다고 판단하고, 트랜잭션을 완료한다[5].

3.2.2 이동 유연 트랜잭션 모델에서 확장된 트랜잭션의 의미론

(1) 위치 기반 실행의 지원

기존의 분산 환경에서 위치 은폐성(location transparency)을 추구하는 것과는 달리, 이동 컴퓨팅 환경은 이동 호스트에게 위치 기반적인 정보를 제공해야 한다. 위치 기반 연산은 데이타의 값이 특정한 위치에 강결합되어 있고, 질의에 대한 응답이 질의를 생성한 지리적인 위치에 의존하며, 질의 결과가 특정한 위치에서 유효함을 의미한다[17]. 이동 유연 트랜잭션 모델에서는 사용자 또는 시스템에게 위치 기반 실행을 외부 술어 조건 Π으로 명시하도록 허용함으로써 이동 컴퓨터의 현재 위치에 따라 다른 정보를 액세스할 수 있도록 하며, 시스템 자체적으로 위치 기반 실행을 구분할 수 있는 구조를 포함한다. 즉, 기존의 유연 트랜잭션 모델에서는 외부 술어 조건으로 시간과 비용만을 기술하였으나, 이동 유연 트랜잭션 모델에서는 위치 기반의 술어 조건을 허용한다.

이동 유연 트랜잭션 모델에서 부트랜잭션에 대한 외부 술어 조건은 P, Q, L로 구성된다. P는 시간에 관계된 외부 술어, Q는 비용에 관계된 외부 술어, L은 위치 기반 종속성을 갖는 외부 술어를 의미한다. 만약, 사용자가 부트랜잭션 t_1, t_2에 대하여 시간은 8시에서 17시 사이에 이루어지도록 하며, 부트랜잭션 t_2, t_3의 비용이 ₩100미만에서 수행되고 t_1, t_4가 위치 기반 종속성을 가질 경우, 다음과 같이 기술한다.

Π = { P, Q, L }
P = { 8<time(t_1)<17, 8<time(t_2)<17 }
Q = { cost(t_2)<₩100, cost(t_3)<₩100 }
L = { t_1, t_4 }

본 논문에서 제안된 이동 유연 트랜잭션 모델은 위치 기반 종속성을 외부 술어 조건에 기술함으로써 실행되는 위치에 따라 이동 사용자에게 다른 결과값을 제공하는 위치 기반 실행을 지원한다.

(2) 핸드오버의 지원

이동 컴퓨팅 환경하에서 이동 사용자는 트랜잭션의

실행 중에도 지속적으로 이동한다. 이동 유연 트랜잭션 모델은 핸드오버 발생시 낮은 대역폭을 갖는 이동 컴퓨팅 환경하에서 통신량이 최적화되도록 이동 트랜잭션 실행을 결정할 수 있는 구조를 포함한다.

이동 유연 트랜잭션은 부트랜잭션의 보상 가능성과 위치 기반적인 특성을 고려하여 이동 트랜잭션의 실행을 결정한다. 부트랜잭션이 보상 불가능한 경우, 실행을 완료하기 위해 전역 결정을 위해 대기해야 하므로 기존의 데이타베이스 시스템을 위한 전역 트랜잭션처럼 한 사이트에서 실행을 관리해야 한다. 이와 다르게 보상 가능한 경우, 이동 유연 트랜잭션의 완료 이전에 부트랜잭션의 완료가 가능하므로 한 개 이상의 사이트에서 실행 관리가 가능하다. 이동 유연 트랜잭션 모델에서는 또한 위치 기반 트랜잭션을 포함한다. 핸드오버 발생시 수행 중인 위치 기반 트랜잭션의 유용성을 검사해야 하며, 위치 기반 트랜잭션이 유용하면 해당 셀에서 실행을 지속시키고, 그렇지 않으면 이전 셀에서의 실행을 철회하고 새로운 셀에서 실행을 재시작해야 한다.

이동 유연 트랜잭션의 실행 중에 핸드오버가 발생될 경우, 보상 불가능하며 위치 기반 연산을 포함하지 않은 부트랜잭션은 새로운 셀로 이전되어 실행되지 않고 이전 셀에서 실행을 지속시키는 구조를 갖는다. 보상 불가능하며 위치 기반 연산을 포함하는 연산의 경우, 이전 셀에서의 실행을 취소하고 새로운 셀에서 재실행하도록 하는 구조를 갖도록 해야 한다. 이와 다르게 보상 가능한 트랜잭션은 위치 기반 실행 여부에 관계없이 이전 셀의 실행을 완료시키고, 새로운 셀에서 계속적인 실행이 가능하다. 이를 위해서 보상 가능한 트랜잭션은 분할 연산을 실시하게 된다. 분할 연산은 한 개의 부트랜잭션을 두 개의 독립적으로 직렬 가능한 부트랜잭션으로 분할함으로써 동적으로 트랜잭션을 재구성한다. 분할된 트랜잭션은 서로 독립적으로 완료나 철회가 가능하며, 분할된 트랜잭션들은 다시 결합 연산을 통하여 결과값들을 재결합함으로써 분할되기 전의 트랜잭션과 실행 결과의 동치값을 얻게 된다[17]. 즉 핸드오버 발생시, 한 트랜잭션을 분리하여 먼저 완료되어진 자원을 반환함으로써 동시성을 증가시키며, 완료되어지지 않은 자원만을 전송함으로써 전송 비용을 낮춘다.

일반적인 핸드오버 제어의 값은 <표 1>과 같은 실행 제어 테이블을 참조하여 이동 사용자가 기술하게 된다. 부트랜잭션이 보상 가능하고 위치 기반일 경우 디폴트값으로 restart(t_i)의 값을 갖으나, 만약 이전 셀에서의 실행이 유효성을 지닌다고 판단될 경우 split_restart(t_i)의 값으로 선택하여 기술한다.

<표 1> 핸드오버 제어 규칙 테이블

	보상 가능(C)	보상 불가능(NC)
비위치 기반	split_resume(t_i)	continue(t_i)
위치 기반	restart(t_i) split_restart(t_i)	restart(t_i)

핸드오버가 발생한 경우에 이동 유연 트랜잭션을 구성하고 있는 H의 값이 다음과 같고, 현재 실행 중인 부트랜잭션이 t_1, t_2, t_3, t_4라고 가정하자.

H = { restart(t_1), continue(t_2), split_resume(t_3), split_restart(t_4) }

실행 중인 이동 유연 트랜잭션은 핸드오버의 발생시 실행 상태를 제어하기 위해 H의 값을 참조하게 된다. 부트랜잭션 t_1의 restart(t_1)값은 핸드오버 발생시 재시작됨을 의미하므로 이전 셀에서의 실행은 철회되고, 새로운 셀에서의 실행이 재시작되어진다. 또한, t_2의 continue(t_2)값은 이전 셀에서의 지속을 의미하므로 핸드오버가 발생되더라도 이전 셀에서 실행을 지속한다. 부트랜잭션 t_3의 split_resume(t_3)값은 분할 연산을 통해 이전 셀에서의 실행을 완료시키고, 새로운 셀에서 완료 시점 이후의 실행을 지속시킨다. 부트랜잭션 t_4의 split_restart(t_4)값은 분할 연산을 통해 이전 셀에서의 실행을 완료시키고, 새로운 셀에서 실행이 처음부터 재시작되어진다.

본 논문에서 제안된 이동 유연 트랜잭션 모델은 핸드오버 발생시 실행 중인 부트랜잭션의 특성에 따라 실행 재시작, 실행 계속, 분할 재실행 및 분할 재시작을 한다. 이러한 핸드오버 제어는 이동 호스트의 트랜잭션 실행 및 위치 정보를 최적으로 전송함으로써 낮은 대역폭을 효율적으로 운용할 수 있게 한다.

(3) 부분 실행의 결과 허용

이동 유연 트랜잭션을 구성하는 일부 부트랜잭션은 핸드오버 제어 규칙에 따라 분할하여 실행된다. 부트랜잭션 t_i에 대해 분할 연산을 수행하면, 하나의 t_i는 t_{if}(t_i의 앞부분)과 t_{il}(t_i의 뒷부분)으로 분할되어 실행된다. 분할된 트랜잭션에 대한 실행 결과값은 분할되기

전 트랜잭션의 실행 결과값과 동치이기 위해서 결합 연산을 수행한다. 결합 연산을 수행시 분할되어서 실행된 결과값들의 결합을 받아들여질 수 있는지에 대해 사용자는 자신의 요구 사항을 승인되어질 수 있는 결합 규칙 J으로써 명세한다. 승인되어질 수 있는 결합 규칙 J의 값으로써 join(t_i)은 분할되어서 실행된 결과값들의 결합이 분할되지 않고 수행한 트랜잭션의 수행값과 동일해야만 그 트랜잭션을 성공적으로 수행됨을 의미한다. 이와 다르게 user(t_i)는 사용자의 요구 사항에 따라 분할되어서 수행된 결과값의 결합이 불완전한 정보를 제공하더라도 사용자에게 필요한 정보 내용을 제공하면 해당 트랜잭션의 실행을 성공적인 완료로 간주한다.

실행 중이던 부트랜잭션 t_1, t_2에 대해 분할 연산이 일어나면, 승인되어질 수 있는 결합 규칙 J는 t_1과 t_2에 대해 적용된다.

H = { split_resume(t_1), split_restart(t_2) }
J = { user(t_1), join(t_2) }

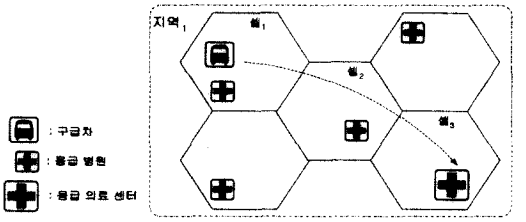
부트랜잭션 t_1은 분할되어서 수행된 결과값의 결합이 불완전한 실행으로도 완료가 가능하며, t_2는 분할되어서 수행된 결과값의 결합이 분할되지 않고 수행된 트랜잭션의 수행값과 동일해야만 실행을 성공으로 간주한다.

본 논문에서 제안한 이동 유연 트랜잭션 모델에서 부분 실행의 허용은 이동 호스트의 사용 시간을 줄이고, 이동 컴퓨팅 환경의 제한적인 통신 성능을 절약하게 한다.

3.2.3 이동 유연 트랜잭션 모델의 정형화의 예

응급 의료 체계는 적정 규모의 지역 내에서 응급 상황 발생시, 효과적이고 신속하게 의료서비스를 제공한다. 응급 의료 체계에서 응급 환자 수송 시스템은 응급 환자의 신고에 따라 환자의 상태에 따른 구급차 출동, 구급 처치, 적절한 의료 기관 선택 및 환자 이송 등 적절한 서비스를 제공한다. 응급 의료 체계에서의 응급 환자 수송 시스템은 (그림 2)와 같이 나타내어진다.

응급 환자 이송 시스템은 구급 출동을 하여 구급 처치 후 가장 빠르게 도착할 수 있는 적절한 의료 기관으로 환자를 이송하고자 하면, 이동 컴퓨터를 탑재한 구급차는 적정 규모로 조직된 지역$_1$내의 셀$_1$에서 가장 가까운 응급 병원을 찾는 위치 기반 트랜잭션을 제기



(그림 2) 이동 컴퓨팅 환경하에서 응급 환자 이송 시스템

한다. 셀$_1$에서 트랜잭션의 실행 중에 셀$_2$로 핸드오버 발생시 이송할 응급 병원을 찾는 트랜잭션을 재시작한다. 만약 가장 가까운 응급 병원을 찾는 위치 기반 트랜잭션이 셀$_1$, 셀$_2$에서 실패하면, 응급실의 시설, 인력 및 장비가 갖추어진 셀$_3$의 응급 의료 센터로 환자가 이송된다. 이송되어질 병원이 선정되면, 선정된 병원에 대해 응급 신호를 보낸다. 또한 가장 가까운 응급 병원이 이송되어질 병원으로 선정되면 셀 내의 지도 정보를 통해 응급 병원의 위치를 확인 후, 이송 작업이 이루어진다. 그러나, 응급 의료 센터는 빈번한 정보 요구로 인해 지역$_1$의 지도 정보를 이동 컴퓨터에 내재함으로써 지도 정보를 필요로 하지 않는 것으로 가정하였다. 응급 환자 이송 시스템은 의료 보험 시스템으로부터 환자 정보를 얻게 된다. 따라서 트랜잭션은 응급 환자의 도착 이전에 검사나 치료를 위한 모든 준비가 완료되게 되고, 환자 정보를 제공받음으로써 응급 환자의 효율적인 관리가 이루어지게 된다.

응급 환자 수송 시스템에서의 제기된 트랜잭션은 <예 1>의 부트랜잭션으로 표현된다. 부트랜잭션 t_1과 t_2는 환자를 이송할 병원을 선택함에 있어서 대안적인 관계에 있으며, <예 1>에서 외부 술어 Π에서 시간과 비용은 기술되지 않는다.

<예 1> 응급 환자 수송 시스템에서 응급 환자 이송 트랜잭션

t_1 : 지리적으로 가까운 응급 병원 찾기
t_2 : 정해진 응급 의료 센터로 이송 결정
t_3 : 응급 신호 보내어서 병원을 대기 상태로 만들기
t_4 : 셀 내의 지도 정보 얻기
t_5 : 의료 보험 시스템으로부터 환자 기록 검색

정의된 이동 유연 트랜잭션 모델을 토대로 하여 응급 환자 이송 트랜잭션을 정형화하면 (그림 3)과 같다.

M = { $t_1(C), t_2(C), t_3(NC), t_4(C), t_5(C)$ }
S = { $t_1 <_s t_3, t_2 <_s t_3, t_1 <_s t_4$ }
F = { $t_1 <_f t_2$ }
Π = { L }
L = { t_1, t_4 }
H = { restart(t_1), continue(t_2), continue(t_3), split_resume(t_4), continue(t_5) }
J = { user(t_4) }
G = { {t_1, t_3, t_4, t_5}, {t_2, t_3, t_5} }

(그림 3) 이동 유연 트랜잭션 모델을 이용한 〈예 1〉의 정형화

응급 환자 이송 트랜잭션의 실행에서 승인되어질 수 있는 상태 A는 (그림 4)와 같다.

A = { (S, -, S, S, S) }∪
{ (-, S, S, -, S) }

(그림 4) 〈예 1〉의 승인되어질 수 있는 상태 집합

응급 환자 이송 트랜잭션의 실행 상태 X가 승인되어질 수 있는 상태 A의 값 중 하나의 값에 도달된 경우, 응급 환자 이송 시스템은 이동 유연 트랜잭션을 성공적으로 수행한 것으로 간주하게 된다. 'S'는 해당되는 부트랜잭션의 실행이 성공적으로 수행되었음을 의미하며, '-'는 'S'이외의 값을 갖게 됨을 의미한다.

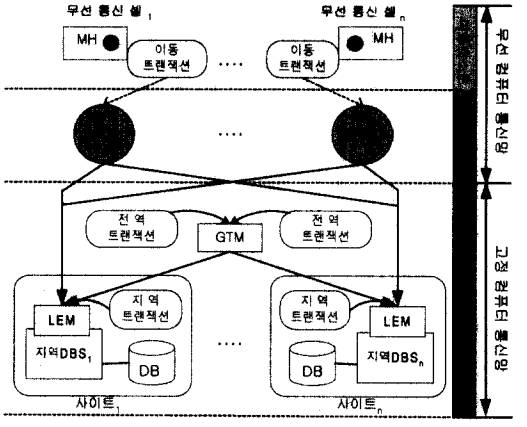
4. 이동 유연 트랜잭션 모델의 실행 제어 알고리즘

4.1 이동 이질 멀티데이타베이스 시스템

이동 이질 멀티데이타베이스 시스템은 (그림 5)와 같이 무선 컴퓨터 통신망내에 이동 호스트(Mobile Host: MH)계층의 이동 유연 트랜잭션, 이동지원국 계층의 이동 트랜잭션 관리자(Mobile Transaction Manager: MTM), 그리고 고정 컴퓨터 통신망 상의 이질 멀티데이타베이스 시스템으로 구성된다.

무선 컴퓨터 통신망 상의 이동 호스트는 이동 사용자의 요구 사항을 <정의 1>에 따라 이동 유연 트랜잭션으로 제기함으로써 고정 컴퓨터 통신망 상의 데이타를 액세스한다. 이동 호스트에서 생성된 이동 유연 트랜잭션은 이동 호스트가 존재하는 셀 내의 이동지원국으로 전달된다.

이동지원국 계층에서 이동지원국은 이동 호스트와 고정 컴퓨터 통신간의 무선 연결을 지원하기 위한 하



(그림 5) 이동 이질 멀티데이타베이스 시스템의 구조

드웨어적, 소프트웨어적 구성 요소를 포함하며, 한 셀 내에 하나씩 존재하게 된다. 분산 환경에서 트랜잭션은 동시성 제어, 중복 관리, 그리고 원자적 완료 기능을 가진 시스템에 의해 조정된다. 이와 다르게 이동 컴퓨팅 환경에서 이동 트랜잭션 실행은 전적으로 시스템에 의해 조정되지 않고, 이동 호스트의 이동성 및 사용자의 응답에 의존함으로 기존의 트랜잭션과 의미론적으로 다르다. 따라서 이동 이질 멀티데이타베이스 시스템에서는 이동 유연 트랜잭션을 위해 이동지원국 계층에 이동 트랜잭션 관리자를 두게 된다. 이동 트랜잭션 관리자는 이동 유연 트랜잭션의 실행을 위해 부트랜잭션의 실행 상태를 관리한다. 또한 이동 트랜잭션 관리자는 분산 환경의 트랜잭션 관리자와는 달리 핸드오버 및 위치 기반 연산의 지원 등을 통하여 이동 호스트의 이동성을 지원하고, 이동 컴퓨팅 환경의 제약 조건들을 최소화하기 위해 부분 실행의 결과 및 분할 연산을 허용케 한다. 이동 트랜잭션 관리자는 이동 호스트로부터 제기받은 이동 유연 트랜잭션을 실행하기 위해 기존의 트랜잭션 관리자가 제공하는 Begin-Transaction, commit, abort 연산뿐만 아니라, 이동 유연 트랜잭션의 실행 상태 정보의 관리 및 split-transaction, join-transaction, compensate 연산을 시스템적으로 지원한다. 이동 호스트에 의해 이동 유연 트랜잭션 생성 후, 이동지원국에 전달된 이동 유연 트랜잭션은 이동 컴퓨팅 환경의 특성상 잦은 접속 단절과 이동성이 존재하는 이동 호스트를 대신하여 이동 트랜잭션 조정자에 의해 제어된다. 이동 유연 트랜잭션은 이동 트랜잭션 관리자를 통

해 이질 멀티데이타베이스 시스템에 제기되기 때문에, 이동 유연 트랜잭션에 의한 데이타 액세스의 요구는 기존의 고정 컴퓨터 통신망상의 전역 트랜잭션의 요구처럼 수용되어진다. 이동 트랜잭션 관리자는 외부적인 요건으로 인한 취소 요구, 또는 핸드오버의 발생을 이동지원국에게 통보받고, 핸드오버 발생시 트랜잭션의 실행을 위해 새로운 셀의 이동 트랜잭션 관리자와 상호 작용하게 한다. 이동 유연 트랜잭션이 제기된 이동지원국내의 이동 트랜잭션 관리자와 핸드오버 발생 이후 새로운 셀의 이동 트랜잭션 관리자들은 참여자(participant)로서 트랜잭션의 실행에 참여한다. 이동 유연 트랜잭션의 실행 중 승인되어질 수 있는 상태에 도달하거나, 더 이상 실행 가능한 부트랜잭션이 존재하지 않은 경우, 해당 셀의 이동 트랜잭션 관리자가 조정자(coordinator)로서 실행 중인 이동 유연 트랜잭션을 완료하거나 철회한다.

고정 컴퓨터 통신망상의 이질 멀티데이타베이스 시스템 계층은 고정 호스트 및 이동 호스트에게 정보 서비스를 제공하기 위한 시스템이다. 기존 지역 데이타베이스 시스템은 이동 컴퓨팅 환경을 지원하기 위해 이질 멀티데이타베이스 시스템으로 통합 가능하며, 통합된 환경에서 협력 실행을 위해 각 지역 데이타베이스 시스템에는 지역 실행 모니터(Local Execution Monitor: LEM)를 추가할 수 있음을 가정한다. 따라서, 각 사이트에서 운용되는 지역 실행 모니터는 이동 유연 트랜잭션의 실행 및 전역 트랜잭션의 실행에 참여한다. 또한, 이동 트랜잭션 관리자와 이질 멀티데이타베이스 시스템과의 인터페이스 및 전역 트랜잭션과 이질 멀티데이타베이스 시스템과의 인터페이스 역할을 담당한다. 이질 멀티데이타베이스 시스템에서 전역 트랜잭션 관리자는 여러 지역 데이타베이스 시스템에 분산 저장되어 있는 공유 데이타를 전역 단계에서 관리하는 작업을 담당한다. 전역 트랜잭션 관리자는 제기받은 전역 부트랜잭션 집합을 관련된 지역 데이타베이스 시스템으로 전달하고, 이들 실행이 전역적 단계에서 직렬화 가능성을 보장하면서 원자적으로 실행되도록 조정하는 조정자 역할을 수행한다. 또한, 전역 트랜잭션 관리자는 고정 컴퓨터 통신망상의 전역 트랜잭션들 사이에서 발생할 수 있는 교착 상태 처리 및 회복 역할을 담당하게 된다. 각 사이트내에서 처리되는 지역 트랜잭션은 지역 트랜잭션 관리자에 의해 지역적 단계에서 직렬화 가능성을 보장하면서 다른 지역 트랜잭션들과 동시적으로 실행되어진다.

4.2 이동 유연 트랜잭션의 실행 제어 알고리즘

이동 유연 트랜잭션의 실행 제어 알고리즘은 고정 컴퓨터 통신망상의 이질 멀티데이타베이스 시스템상의 유연 트랜잭션 모델을 위한 실행 제어 알고리즘을 기반으로 한다[5]. 유연 트랜잭션 모델을 위한 실행 제어 알고리즘은 이질 멀티데이타베이스 시스템을 위한 트랜잭션 처리에 효율성을 갖는다[15]. 그러나, 이동 유연 트랜잭션은 유연 트랜잭션의 실행 제어 알고리즘과는 다르게 이동 컴퓨팅 환경에서 지녀야 할 트랜잭션 모델의 특성 즉, 이동 트랜잭션의 실행시 위치 기반 실행, 핸드오버의 지원, 부분 실행의 결과 허용 등을 지원해야 한다. 본 절에서는 유연 트랜잭션의 실행 제어 알고리즘을 이동 컴퓨팅 환경하에서 이동성을 지원하도록 확장한 이동 유연 트랜잭션의 실행 제어 알고리즘에 대해 기술한다.

이동 유연 트랜잭션의 실행을 위해 이동 트랜잭션 관리자는 각각의 부트랜잭션에 관하여 <정의 2>에 따른 실행 상태 정보 X를 관리한다. 이동 유연 트랜잭션을 구성하는 부트랜잭션 t_i는 실행이 제기되지 않고 (x_i='N'), t_i에 대해 성공 종속성을 갖는 부트랜잭션들이 성공적으로 수행되고($\forall t_k \in S(t_k <_s t_i)$, x_k='S'), t_i에 대해 실패 종속성을 갖는 부트랜잭션들이 실패되어야 하며($\forall t_k \in F(t_k <_f t_i)$, x_k='F'), 외부종속성을 나타내는 외부 술어의 값이 참($\forall t_i \in \Pi$, $\Pi(t_i)$='true')을 만족되면 실행 가능(executable)하다.

이동 유연 트랜잭션 T는 <알고리즘 1>의 실행 제어 알고리즘을 갖는다. 이동 유연 트랜잭션을 구성하는 부트랜잭션들은 보상 불가능(NC)한 트랜잭션일 경우 <알고리즘 2>에 따라 실행되며, 보상 가능(C)한 트랜잭션일 경우 <알고리즘 3>에 따라 실행된다.

<알고리즘 1>의 실행 규칙에 따르면, 이동 유연 트랜잭션은 부분적으로 순서화된 부트랜잭션들의 집합 사이에서 부트랜잭션을 동시 실행함으로써 병렬성을 얻는다. 부트랜잭션은 성공 종속성, 실패 종속성 또는 외부 종속성의 만족 여부에 따라 스케줄되어진다. 부트랜잭션은 실행 결과에 따라 부트랜잭션의 실행 상태가 'E'에서 적합한 실행 상태 'F' 또는 'S'로 변환한다. 부트랜잭션의 실행을 끝마친 후에는 실행 상태 변수가 변경되어짐에 따라, 승인되어질 수 있는 상태 A에 도달했는지 여부를 검사한다. 승인되어질 수 있는

<알고리즘 1> 이동 유연 트랜잭션 T의 실행 제어 알고리즘

```
procedure execute_Mobile_Flex_Transaction(in : T)
/* T = ( M, S, F, Π, H, J, G ) : 이동 유연 트랜잭션  */
/* A : 승인되어질 수 있는 상태(an acceptable state set) */
{  X = ( N, N, ... , N);          /* 실행 상태 변수 X 초기화 */
   find executable_ti;            /* 실행 가능한 부트랜잭션 찾기 */
   do
      concurrently executable_ti do {
         xi = 'E';                /* 동시적으로 실행 가능한 ti를 실행 */
         if (ti == NC) then
            execute through <알고리즘 2>;
         else                     /* 부트랜잭션 ti = C */
            execute through <알고리즘 3>;
         }
      change xi in X to 'S' or 'F' based on execution result;
                                  /* ti 상태 변경 */
      if (X∈A) then               /* A 상태에 도달 */
         commit T using A and exit;
      find executable_ti;
   while(executable_ti = Ø)

   if ((doesn't_have_A) || (executable_ti = Ø)) then
      abort T and exit;           /* T 실행이 실패로 끝남 */
}
```

상태에 도달하지 않은 경우, 실행이 제기되지 않은 부트랜잭션(x_i='N')중에서 실행 가능한 부트랜잭션을 조사하여 실행 가능한 부트랜잭션을 스케줄한다. 부트랜잭션들의 스케줄은 A의 값 중 한 개의 값을 갖거나, 실행 중인 부트랜잭션(x_i='E')이 없고, 더 이상 실행 가능한 부트랜잭션이 없으면서 A에 도달하지 못할 때까지 계속 되어진다. 부트랜잭션들의 실행 중에 A에 도달하면, 조정자 역할을 하는 이동 트랜잭션 조정자는 이동 유연 트랜잭션에 대해 실행을 완료하고, A에 도달하지 못한 경우 이동 유연 트랜잭션에 대해 철회를 실시한다.

<알고리즘 2>는 이동 유연 트랜잭션에서 보상 불가능(NC)한 부트랜잭션 t_i에 대한 실행 제어 알고리즘을 나타낸 것이다.

스케줄된 부트랜잭션이 보상 불가능일 경우 부트랜잭션 t_i는 <알고리즘 2>에 따라 실행된다. 부트랜잭션의 실행 중에 핸드오버 발생시, 부트랜잭션의 핸드오버 제어 규칙 H을 <알고리즘 4>에 적용하여 핸드오버를 처리한다. 보상 불가능한 부트랜잭션은 성공적으로 수행을 마친 경우 prepared-to-commit 상태가 되고 (x_i='S'), 실행이 올바르다고 판단되면 부트랜잭션이 조정자 역할을 하는 이동 트랜잭션 관리자에게 "YES"

<알고리즘 2> 보상 불가능(NC)한 부트랜잭션 t_i의 실행 제어 알고리즘

```
procedure execute_Non_Compensable_subtransaction(in : ti)
{  do
      execute operations;
      if (handover_occurs) then    /* information from MSS */
         handover_control(H(ti));  /* <알고리즘 4>이용 */
   while(last_operation);

   if (execution = OK) then { /* prepared_to_commit 상태 */
      send "YES" vote to coordinator_MTM; /* xi = 'S' */
      receive MESSAGE from coordinator_MTM;
                              /* receive the coordinator decision */
      if (MESSAGE = "COMMIT") then
         commit ti;           /* ti ∈ aj, xi = 'S' */
      else
         abort ti;            /* ti ∈ aj, xi = '-' */
   }
   else
      send "NO" vote to MTM;  /* xi = 'F' */
 }
```

로 투표한다. 이동 유연 트랜잭션을 완료하기 위해 이동 트랜잭션 관리자로부터 받은 응답 메시지가 "COMMIT"이면 부트랜잭션을 완료하고, 아니면 부트랜잭션을 철회한다.

<알고리즘 3>은 이동 유연 트랜잭션에서 보상 가능(C)한 부트랜잭션 t_i에 대한 실행 알고리즘을 나타낸 것이다.

<알고리즘 3> 보상 가능(C)한 부트랜잭션 t_i의 실행 제어 알고리즘

```
procedure execute_Compensable_subtransaction(in : ti)
{ do
      execute operations;
      if (handover_occurs) then              /* information from MSS */
         handover_control(H(ti));            /* <알고리즘 4>이용 */
   while(last_operation);

   if (H(ti) = 'split_resume' or 'split_restart') then {
      if (join-transaction(tid, tij) ≡ ti) then
         execution = OK;  /* join tid and tij is equivalent to original ti;  */
      else
         execution = NOT_OK and compensate(tid);
   }

   if (execution = OK) then
      commit ti and send "YES" vote to coordinator_MTM;  /* xi = 'S' */
   else
      abort ti and send "NO" vote to coordinator_MTM;    /* xi = 'F' */

   receive MESSAGE from coordinator_MTM; /* receive the coordinator decision */
   if (MESSAGE = "COMMIT") then
      no actions;              /* ti ∈ aj, xi = 'S' */
   else
      compensate ti;           /* ti ∈ aj, xi = '-' */
}
```

스케줄된 부트랜잭션이 보상 가능하면, 부트랜잭션 t_i의 실행은 <알고리즘 3>에 따라 실행되어진다. 실행 중에 핸드오버 발생시 핸드오버 제어 규칙에 따라 <알고리즘 4>를 적용하여 핸드오버를 처리한다. 부트랜잭션이 분할 수행되면, 그 트랜잭션을 구성하는 마지막 연산의 수행 후, 결합 연산을 수행한다. 보상 가능한 부트랜잭션은 prepared-to-commit 상태를 거치지 않고, 완료나 철회를 결정할 수 있다. 실행의 결과가 올바르면, 트랜잭션을 완료하고 조정자 역할을 하는 이동 트랜잭션 관리자에게 "YES"로 투표를 한다(x_i='S'). 실행의 결과가 올바르지 않으면, 트랜잭션의 실행 결과를 철회하며 이동 트랜잭션 관리자에게 "NO"로 투표하게 되고(x_i='F'), 이전 셀에서 완료된 연산들에 대해서는 보상 트랜잭션을 수행한다.

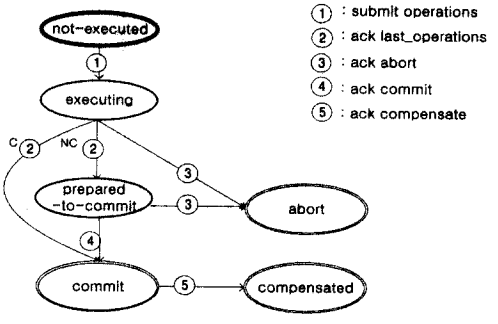
<알고리즘 4>는 이동 유연 트랜잭션에서 핸드오버 제어 규칙 H에 따른 핸드오버 제어 알고리즘을 나타낸 것이다.

<알고리즘 4> 핸드오버 제어 알고리즘

```
procedure handover_control(H(ti))
{  /* H(ti) : 부트랜잭션 ti의 핸드오버 제어 규칙의 값 */
   switch(H(ti)) {          /* check handover_control_rule */
     case 'restart'         : abort ti and retry ti at new cell;
     case 'continue'        : continue;
     case 'split_resume' or 'split_restart' :
        if (J(ti) = 'user') then {
           receive user_response; /* information from MSS */
           if (user_response = "OK") then
              commit ti and send "YES" vote to coordinator_MTM;
           }
        else {
              split-transaction;
           if ('split_resume') then
              commit(tif) and execute(tii); /* xi = 'E' */
           else
              commit(tif) and execute(ti); /* xi = 'E' */
        }
   }
}
```

이동 유연 트랜잭션의 실행 중에 핸드오버 발생시, 이동지원국은 이동 트랜잭션 관리자에게 핸드오버의 발생을 알려주게 된다. 핸드오버의 발생을 통보받은 이동 트랜잭션 관리자는 이동 유연 트랜잭션 모델의 핸드오버 제어 규칙 H에 따라, 핸드오버 절차를 처리한다. 핸드오버 제어 규칙이 'restart'일 경우, 이전 셀에서의 실행을 철회하고 재시작되도록 스케줄되어진다. 핸드오버 제어 규칙이 'continue'일 경우, 이전 셀에서 실행을 지속하게 된다. 핸드오버 제어 규칙이 'split_resume' 또는 'split_restart'일 경우에는, 승인되어질 수 있는 결합 규칙 J를 이용하여 부트랜잭션의 부분 실행의 결과가 허용 가능한지 여부를 판단한다. J의 값이 'user'인 경우, 이동 사용자의 응답을 받아 "OK"이면 부분 실행의 결과만으로 완료 연산을 수행할 수 있도록 한다(x_i='S'). 그렇지 않을 경우, 분할 연산을 수행한다. 핸드오버의 제어 규칙이 'split_resume'인 경우 이전 셀에서의 수행에 대해서는 완료하고 새로운 셀에서 완료 시점 이후부터 실행을 지속하고, 'split_restart'인 경우 이전 셀에서의 수행을 완료하고 새로운 셀에서 실행을 재시작한다. 새로운 셀의 이동 트랜잭션 관리자가 분할 연산되어 실행되는 부트랜잭션을 위해 조정자 역할을 수행하며, 이전 셀의 이동 트랜잭션 관리자는 참여자로서 조정자와 상호 작용한다. 분할 연산의 수행시 실행 상태는 변하지 않는다(x_i='E').

이동 유연 트랜잭션을 구성하는 부트랜잭션 t_i의 실행 상태에 따른 상태 전이도(state diagram)는 (그림 6)과 같다. 상태 전이도에서 굵은 선은 초기 상태를 표시하며, 최종 상태는 이중 타원으로 표시되었다.



(그림 6) 부트랜잭션(t_i)의 상태 전이도

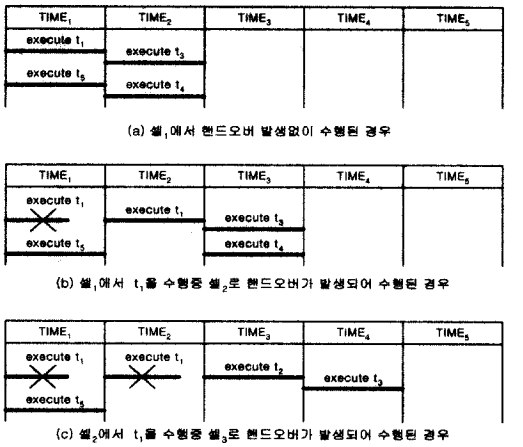
이동 유연 트랜잭션을 구성하는 부트랜잭션은 보상 가능 여부에 관계없이 모두 초기 상태로 not-executed(x_i='N') 상태가 된 후 <알고리즘 1>에 따라 스케줄되어, 실행 상태(x_i='E')가 된다. 부트랜잭션 중 보상 불가능(NC)한 부트랜잭션은 마지막 연산의 실행 후에 prepared-to-commit(x_i='S') 상태가 되거나, 실행 중에 고장 또는 승인되어질 수 있는 상태 A의 값에 따라 abort(x_i='F') 상태가 되어진다. 이와 다르게 보상 가능(C)한 부트랜잭션은 최종 상태로 commit(x_i='S') 상태나, 실행 중에 고장 또는 A의 값에 따라 abort(x_i='F') 상태 또는 compensated(x_i='F') 상태를 지닐 수 있다.

이동 유연 트랜잭션을 위한 실행 완료 및 철회는 다음과 같이 이루어진다. T를 n개의 부트랜잭션 $\{t_1,t_2,\ldots,t_n\}$을 갖는 이동 유연 트랜잭션, T의 실행 상태 $X=(x_1,x_2,\ldots,x_n)$, 승인되어질 수 있는 상태 $A=\{a_1,a_2,\ldots,a_m\}$라고 하면, T는 승인되어질 수 있는 상태($x_i \in A$)에 도달하거나, 더 이상 실행 가능한 부트랜잭션이 존재하지 않을 때까지 실행되어진다. 이동 유연 트랜잭션이 A에 도달함으로써 T의 실행이 끝난 경우, 즉 a_j의 값을 갖는 경우 해당 셀의 이동 트랜잭션 관리자가 전역 조정자로서 이동 유연 트랜잭션의 완료를 담당한다. A에 도달된 경우, 실행 중인 부트랜잭션(x_i='E')과 a_j에서 필요로 하지 않은 완료된 부트랜잭션($t_i \in a_j$, x_i='-')이 존재하기도 한다. 승인되어질 수 있는 상태 a_j에 도달된 이동 유연 트랜잭션을 완료하기 위해, 이동 트랜잭션 관리자는 a_j에서 t_i의 값이 'S'의 값을 지니는 보상 불가능(NC)한 부트랜잭션 t_i에게 "COMMIT" 메시지를 보내고, 'S'의 값을 지니는 보상 가능(NC)한 부트랜잭션 t_i는 prepared-to-commit 상태를 거치지 않고 완료되어져 있으므로 부수적인 연산을 하지 않는다. 실행 중(x_i='E')인 부트랜잭션들은 모두 철회하며, a_j에서 필요로 하지 않은 완료된 부트랜잭션 즉, a_j에서 x_i의 값이 '-'인 완료된 보상 가능(C)한 부트랜잭션에 대해서는 대응되는 보상 트랜잭션을 제기함으로써 보상을 실시한다.

이와 다르게 실행 중인 부트랜잭션(x_i='E')이 없고, 더 이상 실행 가능한 부트랜잭션이 없으면서 A에 도달하지 못하면 수행 중인 이동 유연 트랜잭션에 대해 철회를 실시한다. 이동 트랜잭션 관리자는 prepared-to-commit 상태(x_i='S')에서 전역 결정을 기다리는 보상 불가능(NC)한 부트랜잭션 t_i에게 "ABORT" 메시지를 보낸다. 그리고, 완료된 보상 가능(C)한 부트랜잭션에 대해서는 대응되는 보상 트랜잭션을 제기함으로써 보상을 실시한다.

(그림 7)은 이동 유연 트랜잭션 모델의 실행 제어 알고리즘에 따라 <예 1>의 응급 환자 이송 시스템을 위한 이동 유연 트랜잭션 모델의 실행을 보여준다. 하나의 부트랜잭션은 한 단위의 실행 시간에 실행된다고 가정하였다.

(그림 7)의 (a)는 $셀_1$에서 핸드오버가 발생하지 않을 경우의 이동 유연 트랜잭션의 실행을 의미한다. <알고리즘 1>에 따라 실행 상태 X=(N,N,N,N,N)의 값을 지닌다. 부트랜잭션 t_1은 성공 종속성을 갖지 않고, 실패



(그림 7) 실행 제어 알고리즘을 이용한 <예 1>의 실행

종속성이 없으면서 위치 기반 외부 종속성이 참(true)이므로 실행 가능(executable)하다. 부트랜잭션 t_5는 성공 종속성, 실패 종속성 및 외부 종속성을 갖지 않으므로 실행 가능하다. $TIME_1$에서 보상 가능(C)한 부트랜잭션 t_1, t_5는 <알고리즘 3>을 이용하여 동시에 실행되어질 수 있으며, 실행 상태 X=(E,N,N,N,E)의 값을 지닌다. $TIME_1$ 이후 t_1과 t_5의 실행 결과는 실행 상태 변수값을 'E'에서 'S'로 변경함으로써, 실행 상태 X=(S,N,N,N,S)가 된다. $TIME_2$ 이후 t_1에 대해 성공 종속성을 갖는 t_3, t_4가 성공 종속성을 만족시킴으로써 실행 가능하다. $TIME_2$에서 t_3, t_4의 실행 또한 실행 상태 변수값을 'N'에서 'S'로 변경함으로써, 실행 상태 X=(S,N,S,S,S)가 된다. 변경된 실행 상태는 (그림 4)의 A의 한 값과 동일함으로 이동 유연 트랜잭션의 실행은 성공적으로 종료되었다고 할 수 있다.

(그림 7)의 (b)는 t_1과 t_5의 실행 중에 $셀_1$에서 $셀_2$로 핸드오버가 발생한 경우의 실행을 보여준다. t_1의 핸드오버 제어 규칙 H의 값은 'restart'값을 지니므로 $셀_1$에서의 실행을 철회하고, $셀_2$에서 실행을 재시작하게 된다. 이와 다르게 t_5는 핸드오버 제어 규칙의 값이 'continue'이므로 기존의 $셀_1$에서 실행을 지속하게 된다. 부트랜잭션 t_4는 성공 종속성을 만족시키지 못함으로 실행 가능하지 않다. t_1의 실행이 성공적으로 수행된 후에 t_3, t_4의 실행이 가능하다.

(그림 7)의 (c)는 $셀_2$에서 $셀_3$으로 핸드오버가 발생한 상태의 실행을 보여준다. $셀_3$에서는 t_1의 위치 기반

외부 종속성의 값이 거짓(false)이므로 t_1이 재시작되지 못함으로써, 실행 상태의 값을 'E'에서 'F'로 변경하게 된다. 부트랜잭션 t_1에 대해 실패 종속성을 갖는 t_2는 실패 종속성이 만족됨으로써 실행 가능하다. 부트랜잭션 t_2가 성공적으로 수행되면, 실행 상태 X=(F,S,N,N,S)의 값을 갖는다. 부트랜잭션 t_2에 대해 성공 종속성을 갖는 t_3의 성공적인 실행은 실행 상태 X=(F,S,S,N,S)의 값을 갖게 한다. 실행 상태가 (그림 4)의 A의 한 값과 동일함으로 트랜잭션의 수행이 성공적으로 종료되었다라고 한다.

(그림 7)에서 알 수 있듯이 본 논문에서 제안된 이동 유연 트랜잭션 모델의 실행 제어 알고리즘은 부트랜잭션간의 병렬적 수행을 허용한다. 또한, 이동 호스트의 위치에 따른 위치 기반 연산을 지원하며, 시스템 자체적으로 핸드오버가 발생할 경우 실행을 결정할 수 있는 구조 및 트랜잭션내의 부분 실행의 결과를 허용한다. 또한 실행의 최종 상태로 승인되어질 수 있는 여러 개의 상태를 갖고, 일부 부트랜잭션이 실패하더라도 대안적인 부트랜잭션의 실행을 통하여 고장이 발생하기 쉬운 이동 컴퓨팅 환경하에서 유연성을 갖음을 알 수 있다.

이동 유연 트랜잭션의 동시성 제어 및 완료는 고정 컴퓨터 통신망상의 이질 멀티데이타베이스 시스템상의 유연 트랜잭션 모델을 위한 전역 타임스탬프 알고리즘 및 이질적인 완료 규약[5]에 의존함으로써 지역 데이타베이스의 일관성을 유지한다.

이동 유연 트랜잭션 모델에서 이동지원국 계층의 이동 트랜잭션 관리자는 이동 유연 트랜잭션의 실행을 위한 다른 이동 트랜잭션 관리자와의 동시적 실행을 위한 제어를 담당한다. 유연 트랜잭션 모델에서는 전역 트랜잭션 동시성 제어 계층과 지역 트랜잭션 동시성 제어 계층을 두어 전역 동시성 관리자와 지역 동시성 관리자의 협력으로 동시성을 제어한다. 각각의 지역 동시성 관리자는 완료된 부트랜잭션과 지역 트랜잭션들을 위한 동시성 제어 역할을 담당하며, 전역 동시성 관리자는 모든 전역 트랜잭션의 동시성 실행을 관리한다. 이러한 접근 방식의 경우 사용되는 전역 타임스탬프 알고리즘(global timestamp ordering algorithm)[5]은 지역 자치성을 보장한다. 이동 유연 트랜잭션의 동시성 제어는 고정 컴퓨터 통신망상의 전역 타임스탬프 알고리즘에 의존함으로써 지역 데이타베이스 시스템의 자치성을 보장하고, 지역 데이타베이스의 일관성을 유지한다. 그러나, 유연 트랜잭션 모델과는 다르게 이동성을 갖는 이동 유연 트랜잭션을 위해 전역 타임스탬프 알고리즘은 확장되어야 하며, 확장된 전역 타임스탬프 알고리즘 및 정확성에 대한 검증이 향후 연구로서 진행되어야 한다.

이동 유연 트랜잭션 모델에서 이동 트랜잭션 관리자는 또한 이동 유연 트랜잭션의 원자적 실행을 위해 이동 유연 트랜잭션 실행의 완료를 담당한다. 유연 트랜잭션을 위한 완료 규약은 2단계 완료 규약(two-phase commit protocol)을 변형시킨 이질적인 완료 규약(heterogeneous commit protocol)[5]을 기반으로 한다. 이질적인 완료 규약은 유연 트랜잭션을 구성하는 부트랜잭션의 유형에 따라 완료 규약을 다르게 한다. 즉, 부트랜잭션이 보상 불가능한 경우 2단계 완료 규약을 따르지만, 보상 가능할 경우, "simulating a prepared state"를 두어 보상 불가능한 부트랜잭션처럼 완료가 가능하게 한다. 또한 유연 트랜잭션의 실행의 완료시 승인되어질 수 있는 상태 A의 도달에 영향을 주는 부트랜잭션에 관하여만 완료함으로써 트랜잭션 실행시 고장(failure)에도 불구하고 이질 멀티데이타베이스 시스템내의 전역 트랜잭션의 실행시 트랜잭션 처리의 효율성을 갖는다. 이동 유연 트랜잭션의 완료 규약은 유연 트랜잭션의 이질적인 완료 규약에 의존한다. 그러나, 이동 컴퓨팅 환경의 특성을 지원하기 위해, 위치 기반적인 연산, 핸드 오버의 제어 및 부트랜잭션 레벨의 부분 실행의 결과로 인하여 이질적인 완료 규약이 향후 연구로서 더욱 확장되어야 한다.

5. 기존 트랜잭션 모델과의 비교

본 절에서는 이질 멀티데이타베이스 시스템을 위한 유연 트랜잭션 모델, 기존에 제안된 이동 트랜잭션 모델, 그리고 본 논문에서 제안된 이동 유연 트랜잭션과의 유연성의 지원 여부 및 이질성의 지원 여부를 기준으로 하여 비교하였으며, 비교 결과는 <표 2>과 같다.

〈표 2〉 이동 유연 트랜잭션 모델과 기존 트랜잭션 모델들과의 비교

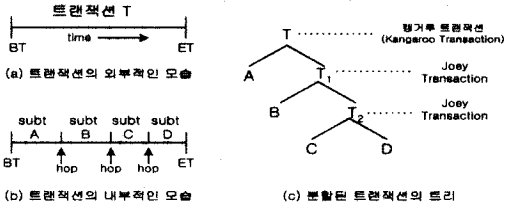
	유연 트랜잭션 모델	기존 이동 트랜잭션 모델	이동 유연 트랜잭션 모델
유연성	지 원	비지원	지 원
이질성	지 원	부분지원	지 원
병렬수행	지 원	비지원	지 원
이동성	비지원	지 원	지 원

유연 트랜잭션 모델은 이동성을 지원하지 않으므로 위치 기반적인 연산의 증대와 핸드 오버의 지원이 필요한 이동 컴퓨팅 환경하에서 적합하지 않다. 반면에 기존에 제안된 이동 트랜잭션 모델의 경우, 이동성을 지원하지만 유연성 및 이질성, 병렬 수행을 지원하지 않음으로써 고장이 발생하기 쉬운 이동 컴퓨팅 환경하에서 트랜잭션 처리의 효율성을 갖기 어렵다.

이동 컴퓨팅 환경의 여러 가지 문제점들을 해결하기 위해서 제안된 다양한 이동 트랜잭션 모델들과 본 논문에서 제안한 이동 유연 트랜잭션 모델의 특성을 좀 더 자세히 비교 분석한다. 분류는 2.1절에서 제시된 이동 컴퓨팅 환경에서 이동 트랜잭션이 가져야 할 특성을 기준으로 지원 여부를 표시하였으며, 분류 결과는 <표 3>와 같다.

이질적인 지역 데이타베이스 시스템의 액세스 지원과 지역 데이타베이스 시스템의 자치성 보장은 지원하는 데이타베이스 모델을 기준으로 하여 분류하였다. 이동 트랜잭션 모델이 이질적인 데이타베이스 시스템을 기반으로 하면 이질적인 지역 데이타베이스 시스템의 액세스를 지원 가능으로 분류하였으며, 멀티데이타베이스 시스템을 기반으로 하면 지역 데이타베이스 시스템의 자치성을 지원 가능으로 분류하였다. 지역 데이타베이스 시스템의 데이타 일관성 지원은 모델에 따라 다르다. 보상 트랜잭션을 이용하여 데이타 일관성을 지원하거나, 객체 의미론에 기반하여 지원하기도 하며, 기반하는 데이타베이스 시스템의 동시성 제어를 통하여 데이타의 일치성을 보장하기도 한다. 위치 기반 실행 및 핸드오버의 지원 가능 여부, 부분 실행의 결과 지원은 이동 트랜잭션 모델이 지원하는 특성에 따라 분류하였다. 부분 실행의 결과 지원은 이동 트랜잭션 모델이 보상 트랜잭션을 이용하여 부트랜잭션의 부분 실행의 결과를 지원함을 의미한다.

캥거루 트랜잭션(Kangaroo Transaction) 모델과 이동 유연 트랜잭션 모델은 2.1절에서 제시한 이동 트랜잭션이 가져야 할 특성에서 많은 유사점을 보인다. 이동 유연 트랜잭션 모델과 같이 캥거루 트랜잭션 모델에서 이동 사용자는 캥거루 트랜잭션을 제기하여 고정 컴퓨터 통신망상의 정보를 액세스한다. 캥거루 트랜잭션은 (그림 8)과 같이 여러 이동지원국에서 순차적으로 실행되는 연산의 집합으로 본다. 즉, 이동 트랜잭션의 실행은 시스템에 의해 조정되어지지 않고 이동 호스트의 이동성에 의존한다. 캥거루 트랜잭션은 이동 호스트가 이동함에 따라 분할 트랜잭션 개념에 따라 한 이동지원국내에서의 실행을 완료하며, 실행을 완료한 트랜잭션의 부분을 분할 기점(split-point)이라고 정의한다. 분할된 트랜잭션의 첫 부분은 부분적으로 완료되어지며, 나머지 부분은 새로운 이동지원국에서 부트랜잭션인 Joey 트랜잭션의 형태로서 실행을 지속함으로써 부분 완료되어진다.

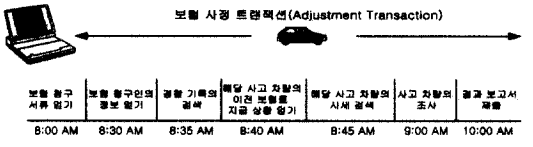


(그림 8) 캥거루 트랜잭션 모델

〈표 3〉 이동 유연 트랜잭션 모델과 기존 이동 트랜잭션 모델과의 비교

	이질 LDBS의 액세스를 지원	LDBS의 자치성	LDBS의 데이타 일관성	유연성의 제공	위치 기반 실행의 지원	핸드오버의 지원	부분 실행의 결과 지원
Reporting and Co-Transaction[11]	○	○	보상 트랜잭션	×	×	×	○
Kangaroo Transaction[12]	○	○	기존 설비에 의존	×	×	○	○
Clustering Model[4]	×	×	클러스터사이의 보장	×	×	×	×
Semantics Based Model[13]	○	○	객체의미론에 기반	×	×	×	×
MDSPTM[14]	○	○	기존 설비에 의존	×	×	×	×
이동 유연 트랜잭션 모델	○	○	기존 설비에 의존	○	○	○	○

이동 유연 트랜잭션 모델과 캥거루 트랜잭션 모델간에 보험 사정인의 업무 과정을 이동 트랜잭션으로 표현하여 비교한다. 이동 컴퓨팅 응용 영역의 예로서 자동차 보험 사정인의 업무과정은 많은 문헌에서 참조되며, 일상에서 이동 컴퓨팅 응용 서비스에 대한 하나의 비전으로서 제시되고 있다[2, 6, 12]. 이동 컴퓨팅 환경하에서 보험 사고 사정인이 자동차 보험을 처리하는 과정을 (그림 9)로 나타내었다.



(그림 9) 자동차 보험 사정인의 보험 사정 트랜잭션

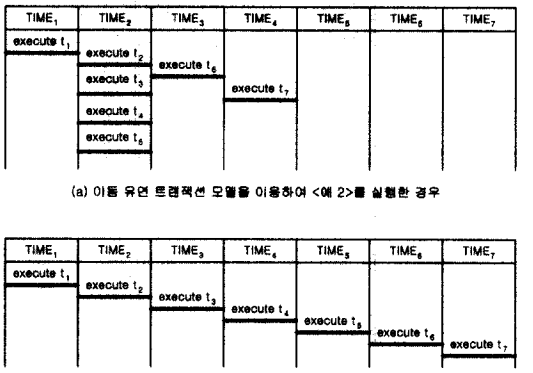
자동차 보험 사정인의 업무는 보험 사정인이 휴대하고 있는 이동 컴퓨터로의 보험 청구 서류가 전달되면서 시작된다. 사정인은 서류를 검토한 후에 필요한 정보들을 모으는 프로그램을 수행한다. 사정인이 사고 현장에 도착하여 검사하는 동안에 필요하다면 추가적인 정보를 수시로 얻을 수 있다. 또한 사고 현장으로 이동하면서 보험 처리에 필요한 정보를 직접 추가하거나 보완하면서 그 결과가 자동적으로 관련 컴퓨팅 서버의 내용이 변경하도록 한다. 마지막으로 입력한 결과를 취합하여 결과 보고서를 작성하면 트랜잭션이 완료하게 된다. 보험 사정 트랜잭션은 <예 2>와 같은 부트랜잭션들로 표현되어질 수 있다.

<예 2> 자동차 보험 사정인의 보험 사정 트랜잭션

t_1 : 보험 청구 서류 얻기
t_2 : 보험 청구인의 정보 얻기
t_3 : 경찰 기록의 검색
t_4 : 해당 사고 차량의 이전 보험료 지급 상황
t_5 : 해당 사고 차량의 시세 검색
t_6 : 사고 차량의 조사
t_7 : 결과 보고서 제출

보험 사정 트랜잭션을 이동 유연 트랜잭션과 캥거루 트랜잭션으로 실행하였을 때의 비교를 (그림 10)으로 나타내었다.

이동 유연 트랜잭션 모델을 이용하여 보험 사정 트



(a) 이동 유연 트랜잭션 모델을 이용하여 <예 2>를 실행한 경우

(b) 캥거루 트랜잭션 모델을 이용하여 <예 2>를 실행한 경우

(그림 10) 이동 트랜잭션 모델을 이용한 <예 2>의 실행 비교

랜잭션을 실행한 예를 (그림 10)의 (a)로서 표현하였다. 부트랜잭션 t_1의 보험 청구 서류를 얻은 후, 부트랜잭션 t_2, t_3, t_4, t_5는 서로간에 이질적인 지역 데이타베이스 시스템을 액세스함으로 동시적으로 실행되어진다. t_2, t_3, t_4, t_5를 실행하면서 사고 현장으로 이동하여 t_6을 성공적으로 실행 후, t_7을 수행함으로써 보험 사정 트랜잭션의 실행을 성공적으로 끝마친다.

캥거루 트랜잭션 모델을 이용하여 보험 사정 트랜잭션을 실행한 예를 (그림 10)의 (b)로서 표현하였다. 캥거루 트랜잭션 모델에서는 부트랜잭션간의 병렬성을 지원하지 않음으로 t_1, t_2, t_3, t_4, t_5, t_6, t_7이 순차적으로 수행됨으로써 이동 유연 트랜잭션 모델과 비교하여 실행 시간이 지연된다.

캥거루 트랜잭션 모델과 이동 유연 트랜잭션 모델간에는 많은 유사점이 존재한다. 그러나, 캥거루 트랜잭션 모델이 핸드오버를 지원하기 위해 트랜잭션을 보상 가능하도록 규정함에 반하여, 이동 유연 트랜잭션 모델은 이동 유연 트랜잭션을 구성하는 부트랜잭션이 보상 가능하거나 보상 불가능한 특성을 갖도록 한다. 또한 캥거루 트랜잭션 모델이 부트랜잭션간의 병렬성을 지원하지 않는데 반하여, 이동 유연 트랜잭션 모델은 부트랜잭션간의 병렬성을 지원함으로써 트랜잭션 처리의 효율성을 갖는다. 그리고 이동 유연 트랜잭션 모델에는 이동 호스트의 위치에 따라서 실행 결과값이 달라지는 위치 기반 실행, 핸드오버의 지원 및 트랜잭션 내의 부분 실행의 결과를 허용한다.

따라서 본 논문에서 제안한 이동 유연 트랜잭션 모

델은 트랜잭션의 실행이 오래 지속되고, 위치 기반 연산이 증가하며, 부분 실행의 결과 허용을 원하는 이동 컴퓨팅 응용 서비스에 폭넓게 이용되어질 수 있다.

6. 결 론

본 논문에서는 이질 멀티데이타베이스 시스템을 위한 유연 트랜잭션 모델의 확장을 통하여 이동 컴퓨팅 환경하에서 이동 이질 멀티데이타베이스 시스템을 위한 새로운 이동 트랜잭션 모델인 이동 유연 트랜잭션 모델 및 실행 제어 알고리즘을 제안하였다.

본 논문에서 제안된 이동 유연 트랜잭션 모델은 이질 멀티데이타베이스 시스템을 위한 트랜잭션의 특성과 이동 컴퓨팅 환경을 위한 트랜잭션의 특성을 반영하였다. 또한, 이동 컴퓨팅 환경하에서 각 사이트에서 자치적으로 운용되는 데이타베이스 시스템을 상호 협력이 가능하도록 통합하여 이동 이질 멀티데이타베이스 시스템으로 발전시키며, 이를 통하여 이동 사용자의 요구 사항을 지원하도록 이동 유연 트랜잭션의 실행 제어 알고리즘을 정의하였다.

본 논문에서 제안된 이동 유연 트랜잭션 모델은 이동 호스트의 현재 위치에 따라 실행 결과가 달라지도록 위치 기반 종속성을 갖는 연산을 지원한다. 또한 이동 컴퓨팅 환경하에서 통신량을 최적화시킬 수 있도록 핸드오버가 발생시 이동 유연 트랜잭션을 구성하는 부트랜잭션의 특성에 따라 실행을 결정할 수 있는 구조를 포함하였다. 그리고 이동 유연 트랜잭션에서는 이동 사용자에게 불완전한 정보를 제공하더라도 이동 사용자에게 필요한 정보 내용을 제공하면 해당 트랜잭션을 실행을 성공으로 간주할 수 있는 부분 실행을 허용케 한다. 또한, 이동 유연 트랜잭션은 이동 사용자에게 사용자의 요구 사항을 표현할 때 대안적인 부트랜잭션의 허용으로 실패에 대해 탄성을 갖는 유연성을 제공한다. 이러한 유연성 제공은 부트랜잭션간의 병렬성을 갖게 함으로써 이동 컴퓨팅 환경의 자원을 효율적으로 사용할 수 있게 해주기 때문에 이동 컴퓨터를 이용하여 보다 긴 접속 시간 및 넓은 응용 범위를 제공할 수 있게 한다.

향후 연구로서 본 논문에서 제안된 이동 유연 트랜잭션 모델의 실행시 고장이 발생하기 쉬운 이동 컴퓨팅 환경하에서 이동성을 효율적으로 지원하는 동시성 제어 및 완료에 관한 연구가 지속되어져야 한다.

참 고 문 헌

[1] B. Marsh, F. Douglis, and R. Caceres, "Systems Issues in Mobile Computing," Technical Report TR-50-93, *MITL*, Feb. 1993.

[2] E. Pitoura and B. Bhargava, "Dealing with Mobility : Issues and Research Challenges," Technical Report CSD-TR-93-070, *Purdue Univerity*, 1993.

[3] T. Imielinski and B. Bardrinath, "Mobile Wireless Computing : Solutions and Challenges in Data Management," *Communication of ACM*, Vol.37, No.10, Oct. 1994.

[4] E. Pitoura and B. Bhargava, "Maintaining consistency of data in mobile distributed environments," *Proceedings of 15th International Conference on Distributed Computing Systems*, 1995.

[5] A. Elmagarmid, Y. Leu, W. Litwin, and M. Rusinkiewicz, "A Multidatabase Transaction Model for INTERBASE," *In Proceedings of the International conference on VLDB*, Aug. 1990.

[6] A. Helal and M. Eich, "Supporting Mobile Transaction Processing in Database Systems," Technical Report TR-CSE-95-003, *University of Texas at Arlington*, 1995.

[7] R. Alonso and H. Korth, "Database System Issues in Nomadic Computing," *Proceedings of the 1993 ACM SIGMOD Conference on Management of Data*, pp.388-392, May 1993.

[8] A. Acharya, B. R. Badrinath, T. Imielinski, and J. Navas, "Towards mosaic like location dependent services," Technical Report, *Rutgers University*, May 1995.

[9] J. Grant, "Incomplete Information in a Relational Database," *Fundamental Information*, Vol.3, No.3, pp.363-378, 1980.

[10] J. Gray, "The Transaction Concept : Virtues and Limitations," *Proceedings of the International Conference on VLDB*, pp.144-154, Sep. 1981.

[11] P. K. Chrysanthis, "Transaction Processing in Mobile Computing Environment," *Proceedings of IEEE Workshop on Advances in Parallel and Distributed System*, pp.77-83, 1993.

[12] M. Dunham and A. Helal, "A Mobile Transaction Model that Captures both the Data and Movement Behaviour," *Mobile Networks and Application (MONET)*, Vol.2, No.2, 1997.

[13] W. D. Gary and P. K. Chrysnathis, "Supporting Semantics-based transaction processing in mobile database applications," *Proceedings of the 14th IEEE Symposium on Reliable Distributed Systems*, Sep. 1995.

[14] L. H. Yeo and A. Zaslavsky, "Layered Approach to Transaction Management in Multidatabase Systems," *The 5th International Hong Kong Computer Society Database Workshop: Next Generation Database Systems*, pp.179-189, 1994.

[15] A. K. Elmagarmid, Database Transaction Models for Advanced Applications, *Morgan Kaufmann Publishers*, 1992.

[16] V. P. Chrysanthis and K. Ramamritham, "ACTA : A Framework for Specifying and Reasoning about Transaction Structure and Behavior," *Proceedings of ACM SIGMOD*, May 1990.

[17] M. H. Dunham and V. Kumar, "Location Dependent Data and its Management in Mobile Database," *Proceedings of IEEE Computer Society, DEXA Workshop*, pp.414-419, 1998.

[18] C. Pu, G. Kaiser, and N. Hutchinson, "Split-Transactions for Open-Ended Activities," *Proceedings of the 14th VLDB conference*, 1988.

구 경 이

e-mail : g9721046@inhavision.inha.ac.kr

1997년 인하대학교 전자계산공학과 (공학사)

1999년 인하대학교 전자계산공학과 대학원(공학석사)

1999년 9월~현재 인하대학교 멀티데이타베이스 연구실 인턴연구원

관심분야 : 트랜잭션 관리, 이동 데이타베이스 시스템, 워크플로우 관리 시스템 등

김 유 성

e-mail : yskim@dragon.inha.ac.kr

1986년 인하대학교 전자계산학과 (이학사)

1988년 한국과학기술원 전산학과 (공학석사)

1992년 한국과학기술원 전산학과 (공학박사)

1990년~1992년 삼성전자 컴퓨터부문 주임 연구원

1996년~1997년 미국, 퍼듀대학교 전산학과 방문연구원

1992년~현재 인하대학교 전산공학과 조교수

1998년~현재 인하대학교 전산정보실 시스템 부장

현재 인하대학교 전자계산공학과 부교수

관심분야 : 멀티미디어 데이터베이스, 정보검색, 이동 데이터베이스, 트랜잭션 관리