□ 기술해설 □

Collaborative Virtual Environment에서의 동시성 제어와 기술

한국과학기술원 김창한·양재헌*

1. Collaborative Virtual Environment

가상환경(virtual reality, 이하 VR)이란 실제와 유사한 가상의 세계를 사용자에게 만들어 줌으로써 사용자로 하여금 실재감을 느끼도록 하는 기술이다. 가상환경은 3D 그래픽 등 관련 기술의 진보에 힘입어 빠른 발전을 하고 있으며 일부는 상용화되기도 하였다. 초기의 가상환경은 단일 사용자가 사용하는 단일 시스템으로 구성되었으나, 망(network) 기술의 발달에 따라 지리적으로 멀리 떨어진 다수의 사용자들에게 단일한 가상환경을 제공하고 동시에 상호작용할 수 있는 환경에 대한 연구가 활발히 진행중에 있다.

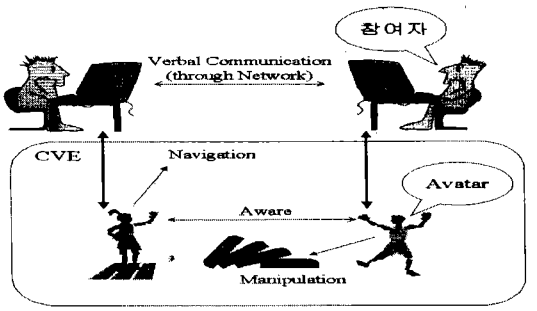


그림 1 Collaborative Virtual Environment 시스템

Collaborative Virtual Environment(이하 CVE) 시스템은 지리적으로 떨어져 있는 다수의 참여자에게 공유된 가상환경을 제공하여 그 속에서 상호작용을 통한 협업을 가능케 하는 시스템이다. CVE 시스템은 VR기술을 통해 현실에 가까운 공유 공간을 제공함으로써 기존의 컴퓨터 기반의 협업 시스템(Computer Supported Cooperative Work, 이하 CSCW)보다 현실감있고 효과적인 상호작용을 가능케 한다.

CVE 시스템의 참여자는, 그림 1에서 보는 바와 같이 크게 세 가지 채널을 통해 상호작용을 하게 된다. 참여자는 첫째, text나 audio를 사용한 대화를 통해 상호작용한다. 이러한 대화 채널은 CVE에만 고유한 채널은 아니나 협업을 보조하는 중요한 채널이 된다. 참여자는 대화를 통해 공동의 목표를 협의하며 작업을 계획하거나 조정할 수 있다.

둘째, 참여자는 다른 참여자의 행동을 관찰할 수 있으며, 한 참여자의 행동은 모든 참여자에게 알려진다. 이러한 CVE의 특징을 awareness라 한다. 기존의 CSCW 시스템이 shared text나 2D drawing space 등의 작업 공간을 제공한데 비해 CVE는 3차원의 실세계와 유사한 공간을 제공할 수 있기 때문에 높은 수준의 awareness를 보장할 수 있다. CVE는 이러한 높은 수준의 awareness를 통해 가상 세계에서의 효과적인 협업을 가능하게 한다. Avatar는 이러한 가상공간에서 참여자를 대리하는 객체를 말한다. Avatar는 실제 사람의 형상을 할 수도 있으며 단순한 형태의 로봇, 동물 등과 같이 다양하게 표현될 수 있다. CVE에서는 이 avatar를 통해 자신의 행동을 다른 참여자에게 알리게 된다. Avatar를 통한 awareness는 기존의 CSCW에서 제공하였던 커서를 통한 것보다 더욱 효과적이며 사실적일 수 있다. 예를 들면 표정이나 손짓 등을 통해 참여자의 행동을 알릴 수도 있다.

셋째, 참여자는 공유 가상 공간을 조작함으

*종신회원

로써 상호작용한다. 다른 CSCW 시스템과 마찬가지로 CVE에서도 공유 객체를 조작하는 것으로 작업을 수행할 수 있다. 작업의 대상과 목표는 응용 프로그램에 따라 달라질 수 있으나, 3차원 가상 공간은 실세계와 거의 같거나, 실세계에서 불가능한 작업 대상과 환경을 제공할 수 있다는 장점을 가진다. 그러나 객체에 대한 공유와 동시적 조작을 통한 협업이 가능하려면 조작에 대한 반응시간이 짧아야 하고, 공유하는 가상환경과 객체에 대한 정보가 일관성있게 유지되어야 한다.

2. CVE의 요구사항

CVE의 일반적인 요구사항은 다른 참여자의 행동을 알려주는 awareness, 가상 세계의 일관성(consistency) 유지와 효율적인 상호작용을 위한 responsiveness로 정리될 수 있다.

2.1 Awareness

참여자는 공유된 가상세계를 돌아다니며 가상세계와 다른 참여자의 행동을 관찰할 수 있다. CVE 시스템은 참여자에게 가상세계와 다른 참여자의 변화를 보여 주어야 한다. CVE에서 참여자는 실시간으로 가상세계를 돌아다니며 가상세계에 대한 조작을 하므로 이러한 참여자의 모든 움직임을 다른 참여자에게 알리는 데는 망 대역폭이 많이 필요하다.

2.2 Consistency

CVE 시스템은 다수의 참여자가 동시에 접근 및 상호작용하는 특징 때문에 일관성이 요구된다. 이러한 일관성 유지는 일반적인 분산 공유 시스템에서도 요구되어 왔다. CVE 시스템에서의 일관성 문제가 다른 환경에서의 일관성 문제와 구별되는 특징은 실시간 상호작용을 요구한다는 점이다.

2.3 Responsiveness

CVE 시스템은 사람이 직접 참여하여 상호작용하는 시스템으로 responsiveness를 요구한다. 참여자가 내린 명령에 대한 반응 시간이 길어지면 사용자는 연속되는 작업의 흐름에 방해를 받게 되며 현실감을 떨어뜨리게 된다. CVE 시스템에서의 반응시간은 지역 시스템에서의 처리 시간에 더해서 공유 가상세계에 대한 다수 참여자의 동시적 접근을 처리하기 위한 시간이 더해지므로 responsiveness는 가상세계의 consistency 문제와 밀접한 관련이 있다.

2.4 CSCW 요구사항과의 비교

기존 CSCW 분야의 연구에서는 CSCW 시스템의 요구사항으로 WYSIWIS(What You See Is What I See)를 제안한 바 있다[1, 2]. WYSIWIS는 협업중인 모든 참여자에게 같은 view를 제공하고자 하는 것으로, 이를 통해 다른 참여자에 의한 작업의 과정을 파악하고 작업 결과를 공유할 수 있다.

WYSIWIS는 공유 객체의 일관성 유지와 조작에 대한 반응 시간을 모든 노드에서 같도록 하는 것으로 나누어지며 객체에 대한 조작 및 커서를 통한 낮은 수준의 awareness를 포함한다. 그러나 분산 환경에서 반응 시간을 일치시키는 것은 힘들 뿐만 아니라 오히려 협업을 방해할 수 있다. 따라서 많은 CSCW 시스템은 이 요구조건을 완화하여 결국 공유 객체에 대한 consistency, responsiveness와 어느 정도의 awareness를 목표로 설계되었다.

기존의 CSCW 시스템은 주로 2D drawing tool을 대상으로 하였기 때문에 view를 통일하는 것으로 CSCW 시스템의 요구사항을 표현할 수 있었으나 CVE의 요구사항은 이것으로 개념화 될 수 없다. CVE에서 각 참여자는 다른 참여자와 정확히 같은 view를 가지지 않으며 오히려 각자의 시점에서 가상 세계를 바라보게 된다. CSCW나 CVE는 모두 consistency를 요구하나 CVE는 view의 consistency가 아닌 가상 세계의 consistency를 요구하는 것이다. 또한 기존의 CSCW 시스템에서는 참여자의 행동에 대한 awareness가 공유 객체에 대한 조작이나 커서 정도였기에 WYSIWIS로 표현될 수 있었으나 CVE 에서는 별도로 avatar에 의한 높은 수준의 awareness를 요구하게 된다. 따라서, awareness, consistency, responsiveness의 세 가지 요구사항은 WYSIWIS를 포괄

하는 일반적인 요구사항이라 할 수 있다.

3. CVE에서의 동시성 제어 기법

동시성 제어(Concurrency Control) 기법은 공유 객체에 대한 동시적 조작을 처리하는 기법이다. CVE 시스템에서의 동시성 제어는 일관성의 유지와 빠른 반응 시간을 모두 요구하는 특징을 가진다. CVE에서 동시적인 조작에 의해 일관성이 유지되지 않는 경우의 예가 그림 2에 있다. 서로 다른 노드에서 발생한 event 1, 2는 전송과정의 지연시간으로 인해 각 노드에 전달되는 순서가 달라진다. 그 결과로 그림 2와 같이 참여자마다 서로 다른 가상세계를 보게되면 협업이 어렵게 된다. 본 장에서는 CVE에서 동시적 접근에 의해 발생하는 이러한 문제들을 해결하는 동시성 제어 기법과 동시성 제어 기능의 분산 방법에 대해 설명한다.

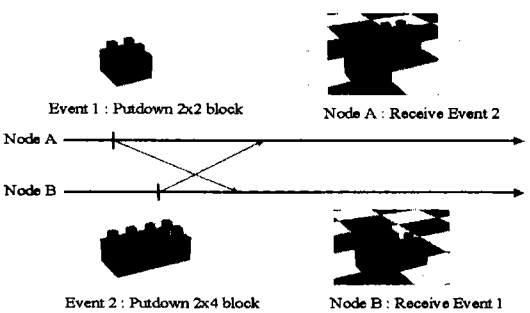


그림 2 CVE에서 동시적 조작에 의한 불일치

3.1 동시성 제어 기법

자주 이용되는 동시성 제어 기법으로는 locking과 ordering의 기법이 있다. Locking은 동일한 객체에 대한 접근을 한 순간에 한 참여자 이하로 제한하는 기법이며 ordering은 각 참여자가 발생시킨 event를 모든 참여 노드에서 같은 순서로 처리하도록 함으로써 일관성을 유지하는 기법이다.

Locking은 공유된 데이타에 대한 접근을 한 명 이하로 제한함으로써 데이타의 일관성을 유지하는 기법이다. CVE 시스템에서는 가상세계를 공유 데이타로 보고 이 공유 데이타에 대한 여러 참여자의 접근을 제어함으로써 일관성을 유지한다. 일반적으로 locking은 구현이 쉽다고 알려져 있으나 동시적 접근에 대해 하나의 참여자에게만 조작을 허용하는 단점을 가진다. 조작 요청이 거부된 참여자에게는 어떤 방식으로든 거부되었다는 사실을 알려 주어야 한다. Locking을 이용하여 효율적인 동시성을 제공하기 위해서는 적절한 lock의 단위(granularity)를 고려해야 한다. Lock의 단위가 너무 작다면 많은 lock 요구가 발생하게 되며 반대로 lock의 단위가 너무 크다면 가상세계의 조작에 대한 많은 요청이 거부되게 된다. 이러한 locking을 통한 동시성 제어 기법은 동시적인 조작을 허용하지 않아도 좋은 객체기반의 시스템에 유용하며 BrickNet, DIVE 등의 CVE 시스템에서 사용되었다[3, 4].

Event ordering은 분산시스템에서 각 참여자가 발생시킨 event를 모든 참여자에 같은 순서로 전달 또는 처리될 수 있도록 함으로써 궁극적으로 일관성을 유지할 수 있도록 하는 방법이다. Event ordering은 causality 및 순서의 엄격성을 기준으로 다양한 수준으로 제공될 수 있으며, 순서를 엄격하게 제한하여 유지할수록 지연시간이 길어지는 trade-off의 관계에 있다[5].

Ordering은 메시지에 timestamp를 붙여 보내는 방법으로 구현된다. 이 timestamp는 실제의 시간일 필요는 없으며 메시지의 순서화를 위해 필요한 정보를 이용하게 된다. 일반적으로 많이 사용하는 방법은 vector timestamp를 이용하는 경우와 logical timestamp를 이용하는 방법이 있다. Ordering 기법은 동시적인 명령을 허용하여 높은 동시성을 보장하나 구현이 복잡하고 지연시간이 길기 때문에 많은 CVE 시스템에서 사용되지 못하였다.

3.2 동시성 제어 기능의 분산

동시성 제어 기법은 동시성의 제어를 누가 하는가에 따라 크게 두 가지로 나누어 볼 수 있다. 서버 기반의 동시성 제어 모델은 동시성을 제어하는 서버를 두고 동시성 제어 서버가 전체 참여자의 조작을 제어하는 모델이다. 서버 기반의 동시성 제어 모델은 구현이 쉽다는 장점을 가지나 모든 메시지가 서버를 매개함으

로 인해 메시지 병목(bottleneck) 현상을 발생시킨다. 분산 동시성 제어 모델은 전체 가상세계의 동시성을 제어하는 단일한 서버없이 각 참여자간의 통신을 통해 동시성을 제어하는 모델이다. 각 참여자의 가상 세계에 대한 조작은 전체 참여자에게 전송되며 부가적인 프로토콜을 이용하여 동시적인 접근에 대한 제어를 국지적으로 하게 된다. 동시성 제어 기법과 그 기능분산을 기준으로 기존의 시스템을 분석해 보면 다음과 같다.

dVS[6]는 서버-클라이언트 구조하에서 locking을 이용하여 일관성을 유지하는 대표적인 시스템이다. 각 클라이언트는 서버에 lock을 요구하여 허가를 얻은 후에 객체를 조작할 수 있다. 서버는 이미 다른 참여자에게 lock을 준 객체에 대한 요구는 거부하며 lock을 가진 클라이언트에 의해 수정된 객체는 그 객체를 복제해 가지고 있는 모든 클라이언트에게 multicast된다. 이러한 시스템에는 dVS 이외에도 BrickNet[3], VISTEL[7], AVIARY[8], Spline[9] 등이 존재한다. 서버-클라이언트 구조에서 서버에 의해 lock을 관리할 경우에는 lock을 얻기 위한 망에서의 지연시간뿐만 아니라 서버로의 병목현상이 발생하여 생기는 지연시간이 추가로 발생하게 되는 단점이 있다.

객체의 lock을 관리하는 단일한 서버가 없는 구조에서는 distributed locking을 사용하기도 한다. Distributed locking은 모든 참여자에게 lock을 요구하여 획득한 후 자신이 객체를 조작하는 동안 다른 참여자가 같은 객체를 조작하지 못하도록 하는 방법이다. DIVE[4]는 distributed locking을 구현한 대표적인 시스템이다. 초기의 DIVE는 event ordering을 보장하는 ISIS protocol에 의해 일관성을 보장하였으나 최근의 DIVE는 distributed locking을 이용하고 있다. DIVE에서는 모든 참여자들에게 lock을 요구해서 모두의 허가를 받을 경우에만 가상 세계의 객체를 조작할 수 있다. 이러한 distributed locking은 모두에게서 lock에 대한 허가를 얻음으로 인해 모든 참여 노드가 lock 관리의 부담을 갖는 단점을 가지게 된다.

Distributed locking의 단점을 극복하기 위한 방법으로 token을 사용하는 방법이 있다. 가상환경의 객체 단위 또는 특정한 lock의 단위에 token을 배정하고 그 token을 가진 참여자에게 객체를 수정할 권한을 주는 방법이다. 이러한 token을 기반으로 하는 기법은 lock관리의 부담을 분산시키며, 여러 참여자에게서 권한을 얻어야 하는 부담을 없애줄 수 있다.

서버 기반의 ordering은 클라이언트가 전송한 메시지를 단순히 서버가 순서를 매겨 모든 참여자에게 재전송 하는 방법으로 구현된다. CoCAD[10]는 서버에 기반 하여 ordering을 하는 대표적인 시스템이다 CoCAD 시스템은 일종의 Group CAD 시스템으로서 단일 서버를 통하여 공유 데이타에 가해지는 조작들을 순서화 하는 방법을 택하였다. 각 참여자의 조작들은 단일 서버로 전해지고, 단일 서버에서 순서가 정해진 후, 다시 모든 참여자에게 같은 순서로 전송되어서, 각각의 참여자 노드에서 처리된다. 서버에 기반 하여 event order를 보장하는 기법은 서버에 도착하는 순으로 순서화를 하게 되면 분산 ordering보다 지연시간을 줄일 수 있는 장점이 있다.

Peer-to-peer 구조하에서 event ordering을 보장하는 기법은 서버-클라이언트 구조하에서 보다 구현이 복잡하여 실제의 다중 참여자 가상현실 시스템에서는 사용한 예가 많지 않다. 초기 DIVE는 vector timestamp를 이용한 ISIS라는 프로토콜을 사용하였으나 ISIS는 ordering을 위한 지연시간이 큰 단점을 가지고 있다.

군사용 전장 시뮬레이션을 목적으로 만들어진 NPSNET은 HLA의 ordering 기준으로 보자면 가장 수준이 낮은 receive order만을 허용한다. Receive order는 각 노드에서 자신에게 전송되어진 순서대로 event를 처리하는 방식으로, 실제로는 의미있는 순서를 보장하지 않는다. 따라서 NPSNET에서는 가상 세계의 공유 객체를 조작할 수 없으며 오직 avatar의 이동과 포탄의 발사와 같은 조작만이 허용된다.

최근에 개발중인 DMSO의 HLA는 분산 환경에서 다양한 ordering을 제공하고자 한다. HLA에서는 각 참여자의 logical timestamp를

찍은 메시지를 전달함으로써 각 참여자가 메시지에 찍힌 시간 순으로 event를 처리하게 되며 이것의 응용을 통하여 ordering을 보장하려고 한다. 그러나 이 방법은 각 참여자가 순서를 보장하기 위하여 다른 참여자의 logical time을 알아야 하기 때문에 지연시간이 길어지고 지연시간의 bound를 정해주지 못한다. 따라서 실시간 가상 현실보다는 시뮬레이션 분야에 적합하며 실제로 HLA는 그러한 목적으로 개발중이다.

4. 낙관적 동시성 제어 기법

3장에서 살펴 본 동시성 제어 기법들은, 일관성을 위협하는 동시적 조작이 일어나는 것을 최악의 경우로 취급하여 이를 방지하고자 하기 때문에 비관적 동시성 제어(pessimistic concurrency control) 기법이라고도 불린다. 기존의 비관적 동시성 제어 기법은 참여자가 내린 명령의 수행이 일관성의 보장을 위해 지연되는 단점을 가지기 때문에 사람이 직접 상호작용하는 실시간 상호작용 시스템에서 사용자의 자연스런 작업을 방해하게 된다[1].

반면에 낙관적 동시성 제어(optimistic concurrency control) 기법은 참여자의 명령을 즉시 수행하고 참여자가 다른 작업을 하는 동안 불일치를 수정하여 실시간 상호작용을 보장하게 된다. 하지만 이러한 낙관적 동시성 제어 기법은 참여자의 명령들이 잦은 충돌을 일으킬 경우 오히려 참여자의 작업에 혼란을 줄 수 있다. 기계들에 의해서만 수행되는 일반적인 분산시스템과는 달리 CSCW 시스템과 같이 사람이 직접 참여하는 분산 시스템에서는 참여자 스스로가 충돌을 회피하려는 경향이 존재하기 때문에 낙관적 동시성 제어가 더욱 적합할 수 있다[1]. 특히, CVE 시스템의 경우 CSCW 시스템보다 높은 수준의 awareness를 제공하기 때문에 참여자의 충돌 회피를 도울 수 있다.

그러나, 이러한 낙관적 동시성 제어 기법은 충돌 검사와 복구를 하기가 어려운 단점이 있다. 많은 CSCW 시스템들은 실시간 상호작용을 보장하기 위하여 낙관적 동시성 제어 기법을 구현하였다. Ellis는 group text editor에서 vector timestamp에 의한 충돌 검사와 transform을 이용한 충돌 복구를 제안하였다[11]. 객체간 종속관계가 없는 group 2D graphic tool에서는 object 기반의 lock이나 충돌 검사를 수행할 수 있다[12, 13].

4.1 낙관적 Locking

낙관적 locking 기법에서는 각 참여자가 객체를 조작하고자 할 때, lock을 얻으리라고 가정하고 지역적으로 조작을 한 후 실제로 lock을 얻으면 그 조작의 결과를 모든 참여자에게 알린다. Lock을 얻는데 실패하면 조작은 취소되며 조작하기 전의 상태로 복구된다. 만약 여러 객체에 대한 연속적인 조작을 수행할 때 최초의 조작에 대한 lock을 얻는데 실패하면 이에 종속된 모든 조작이 취소되어야 한다. 낙관적 locking에서의 이러한 연속적인 취소 동작은 구현이 어려울 뿐만 아니라 참여자에게 혼란을 줄 수 있다.

비관적 locking 기법에서와 마찬가지로 lock의 단위를 설정하는 것이 문제가 된다. 시스템의 관점에서 볼 때, lock의 단위가 크다면 작은 회수의 lock 요구를 통해 많은 작업을 할 수 있으나 lock 요구에 대한 충돌이 빈번해져 조작이 무효화(invalidate)될 확률이 높아지며, 반대로 lock의 단위가 작다면 동시적인 조작이 용이한 반면 lock에 대한 요구가 빈번해지고 한 조작을 위해 여러 개의 lock을 얻어야 하는 경우가 발생한다[14]. 이러한 lock의 단위는 응용프로그램의 특성에 맞게 적용되어야 하며, 동시성을 높이기 위해 lock의 단위를 변형 가능하도록 구현한 경우도 있다[15].

4.2 낙관적 Ordering

비관적 ordering은 각 참여자가 발생시킨 조작 명령 메시지의 수행을 각 메시지의 순서가 보장될 때까지 지연시킨다. 메시지의 순서가 보장된다는 것은 메시지가 가진 timestamp 보다 작은 timestamp를 가진 메시지가 발생되지 않는다는 것이 보장되는 것이며, 이는 각 참여자의 timestamp를 모두 조사함으로써 알 수 있다. 따라서 비관적 ordering은 명령을 수행하는데 걸리는 시간이 길어지게 된다.

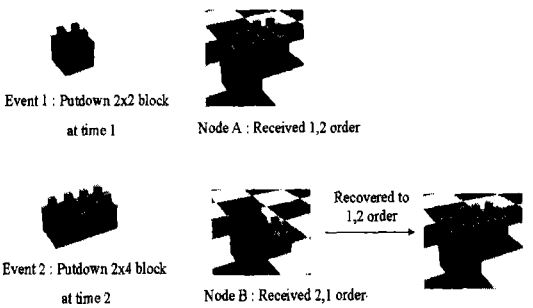


그림 3 낙관적 ordering에서의 수행 과정

낙관적 ordering은 메시지 순서의 보장 없이 메시지를 즉시 처리한다. 만약 순서가 어긋난 메시지가 도착한다면 그 메시지보다 늦은 time-stamp를 가진 메시지들 중 이미 처리된 메시지들을 모두 undo하고, 새로운 메시지를 수행한 후, 다시 나머지 메시지들을 redo함으로써 메시지의 올바른 순서를 복구한다. 그림 3에서 노드 B의 경우 늦게 도착한 event 1이 더 작은 timestamp를 갖고 있기 때문에 event 2를 undo하고, event 1을 수행한 후에, redo하여 노드 A와 같은 상태로 복구한다. 그러나 이러한 단순한 복구방법은 시간순서대로 수행되지 않은 모든 메시지의 처리를 복구하기 때문에 그에 드는 비용이 높을 뿐만 아니라 복구하지 않아도 결과가 같은 메시지들의 복구로 인하여 사용자의 혼란을 더욱 가중시키게 된다. 따라서 낙관적 ordering을 효율적으로 구현하기 위해서는, 순서가 뒤바뀌어 수행되면 서로 다른 결과가 되는 경우에만 복구를 해주는 알고리즘이 필요하게 된다. Group 2D drawing tool의 경우에는 객체간의 종속관계가 없기 때문에 한 객체에 대한 조작은 다른 객체에 영향을 주지 않는다. 따라서 같은 객체에 대한 조작만 순서가 지켜지면 된다[12]. 그러나 일반적인 CVE 시스템에서의 객체는 서로간에 종속관계를 가질 수 있기 때문에 이 종속관계를 검사하여 메시지간 충돌을 밝혀 내야 한다.

낙관적 ordering은 구현하기가 쉽지 않고 사용자에게 혼란을 가져다줄 수 있다는 단점이 있으나 빠른 반응시간으로 실시간 상호작용을 가능케 해 준다. 비관적 ordering의 경우 바른 순서가 보장될 때까지 시스템이 기다려야 하기 때문에 연속적인 동작을 자연스럽게 수행할 수 없으나 낙관적 ordering의 경우에는 순서의 보장 없이 즉시 수행하기 때문에 연속적인 조작을 짧은 시간에 자연스럽게 할 수 있다.

5. 맺는 말

본 논문은 새로운 컴퓨터 기반의 협업 환경으로 부각되고 있는 Collaborative Virtual Environment에서의 동시성 제어 문제에 대하여 간략히 기술하였다. CVE는 높은 수준의 awareness와 현실감 있는 공유 공간을 제공할 수 있기 때문에 효과적인 협업 시스템을 제공할 수 있을 것으로 예측되고 있다.

현재 CVE 시스템에서의 핵심적인 과제는 기존의 VR기술의 과제와 더불어 지리적으로 떨어진 참여자간의 객체 공유와 동시적 조작을 효율적으로 해결하는 것이다. 이러한 동시성 제어 문제는 기존의 CSCW 분야를 비롯한 분산시스템에서 많이 연구되어 왔으나 CVE 시스템에서의 동시성 제어는 공유 데이타의 양이 많다는 점, 조작이 복잡하며 객체간 종속관계가 있다는 점, 일관성 유지 및 빠른 반응시간을 동시에 요구한다는 점에서 차이가 있다.

CSCW 분야의 연구에서는 빠른 반응시간을 제공하면서도 일관성을 유지하기 위한 기법으로 낙관적 동시성 제어기법을 적용한 예가 많으며, 이러한 접근법은 CVE에서도 유효할 것으로 보인다. 그러나 CVE에서 낙관적 동시성 제어 기법을 적용하기 위해서는 객체간 종속관계가 있는 조건에서의 충돌감지와 복구를 효율적으로 할 수 있는 기법이 개발되어야 한다.

참고문헌

[1] Stefik, M., Bobrow, D.G., Foster, G., Lanning, S. and Tatar, D. (1987) "WYSIWIS revied : Early experiences with multiuser interfaces." ACM Transactions on Office Information Systems, April 1987.

[2] Greenberg, S. and Marwood, D., "Real time groupware as a distributed system : Concurrency control and its effect

on the interface." Proceedings of the ACM CSCW Conference on Computer Supported Cooperative Work, 1998.

[3] Gurminder Singh, Luis Serra, Willie Png, Audrey Wong, Hern Ng, "Bricknet: Sharing Object Behaviors on the Net", IEEE VRAIS, pp. 19~25, 1995.

[4] Christer Carlsson, Olof Hagsand, DIVE a Multi-User Virtual Reality System, in Proc. Of VRAIS 93, Seattle, September 1993.

[5] Richard M. Fujimoto, Richard M. Weatherly, "HLA Time Management and DIS", in Proc. Of Simulation Interoperability Workshop, 1996.

[6] division Ltd, "dVS Technical Overview, Vision 2.0.4 first edition", 1993.

[7] Ohya, Jun, Kitamura, Yasuichi, Yakemura, Haruo, et. al., "Real-time Reproduction of 3d Human Images in Virtual Space Teleconferencing." Proceedings of IEEE Virtual Reality International Symposium, pp. 408~414, september 1993.

[8] David N. Snowdon and Adrian J. West, "AVIARY : Design Issues for Future Large-scale Virtual Environments," PRESENCE, Vol 3, No 4, pp. 288~304, 1994.

[9] John W. Barrus, Richard C. Waters and David B. Anderson, "Locales and Beacons : Efficient and Precise Suport For Large Multi-User Virtual Environments," in Proc. Of VRAIS '96, 1996.

[10] Mark A. Gisi, Cristiano Sacchi, "Co-CAD: A Collaborative Mechanical CAD System," RESENCE, Vol 3, No. 4, pp. 341~350, Fall 1994.

[11] Ellis, C.A. and Giibbs, S.J, "Concurrency control in groupware systems." In Proceedings of the ACM SIGMOD International Conference on the Management of Data, pp. 399~407, Seattle, Washington, USA.

[12] Karsenty, A. and Beaudouin-Lafon, M., "An algorithm for distributed groupware applications." In Proceedings of the 13th International Conference on Distributed Computing Systems ICDCS '93, 1993.

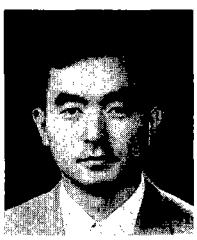
[13] Greenberg, S., Roseman, M., Webster, D. and Bohnet, R., "Human and technical factors of distributed group drawing tools." Interacting with Computers, 4(1), pp. 364~392, 1992.

[14] Bernstein, P., Goodman, N. and Hadzilacos, Concurrency control and recovery in database systems, Addison-Wesley, 1987.

[15] Newman-Wolfe, R. E. and Pelimuhandiram, H. K., "MACE : A Fine Grained Concurrent Editor." In Proceedings of the ACM COCS Conference on Organizational Computing Systems, pp. 240~254, 1991.

김 창 한

1997 한국과학기술원 전산학과 학사
1997~현재 한국과학기술원 전산학과 석사과정 재학중
관심분야 : 분산시스템, 운영체제, 분산가상현실
E-mail : kimch@cslovis.kaist.ac.kr



양 재 헌

1985 서울대학교 컴퓨터공학과 학사
1987 서울대학교 컴퓨터공학과 석사
1987~1989 한국통신전임연구원
1994 메릴랜드 주립대학 전산학 박사
1996 밀즈대학교 전산학과 조교수
1996~현재 한국과학기술원 전산학과 조교수
관심분야 : 분산시스템, 운영체제, 가상현실
E-mail : jhyang@cs.kaist.ac.kr