유무선 네트워크 환경에서 실시간 그룹웨어를 위한 사용자 관심 영역에 기반한 뷰 동기화 프레임워크

(A User Interest-based View Synchronization Framework for Real-time Groupware over Wired and Wireless Networks)

최 미 진 † 조 은 영 †† 강 경 란 ††† 이 동 만 †††
(Mijin Choi) (Eunyoung Cho) (Kyungran Kang) (Dongman Lee)

요 약 무선 통신 환경의 발달은 유무선 통합 환경에서의 실시간 협동 작업을 지원하는 그룹웨어에 대한 요구를 증가시키고 있다. 그런데, 유무선 통합 환경에서 협동작업을 하기 위해서는 현격한 네트워크 대역폭의 차이에서 발생하는 데이타 전송 지연 시간의 차이를 극복하는 뷰 동기화 기법이 중요하다. 본 논문에서는 완화된 WYSIWIS를 활용하여, 사용자의 관심 영역에 기반한 뷰 동기화 기법을 제안한다. 무선 사용자는 단말기의 작은 화면 때문에 공유 작업 영역 중 일부분만을 보게 되므로 이 관심 영역에서 발생되는 이벤트를 우선적으로 전송함으로써 관심 영역 내에서의 뷰 동기화를 가능하게 한다. 본 논문에서는 ns-2를 사용한 시뮬레이션을 통하여 제안하는 기법의 성능을 평가하였으며, 사용자 관심 영역 내에서 무선 사용자들이 유선 사용자들과 실시간 협동 작업을 할 수 있을 만큼 낮은 메시지 전송 지연 시간을 유지하는 것을 보였다. 또한, 다양한 실시간 그룹웨어 시스템 개발에 활용될 수 있도록 제안된 뷰 동기화 기법을 기존의 그룹웨어 개발 환경으로 사용되는 Habanero를 활용하여 프레임워크의 형태로 구현하였다. 사용자를 위한 클라이언트로 간단한 텍스트 에디터를 구현하였다.

키워드 : 뷰 동기화, 실시간 그룹웨어, 유무선 네트워크, 사용자 관심 영역

Abstract With the proliferation of a wireless support and the advances of mobile devices, there is a growing need of making it available for a wireless network as well. Low bandwidth of a wireless network is one of the major issues that should be considered for extending a synchronous collaboration system developed for a wired network to support a wireless network. We propose a priority-based view synchronization scheme, exploiting the relaxed synchronization. The proposed scheme leverages the fact that mobile users need not have the exactly same view as the users in a wired network since they usually have a limited screen size and thus focus only on part of a whole document. We evaluate the performance of a simple group editor with the propose scheme using ns-2. The performance results show that the proposed scheme keeps the message transmission delay low so that mobile users can have synchronous colloboration with wired users without signification delay. The proposed scheme is designed and implemented as a framework to support various kinds of realtime groupware applications, exploiting the existing groupware framework, Habanero. As a client program, we implemented a simple group text editor.

Key words : view synchronization, real-time groupware, wired and wireless networks, user interest

· 본 논문은 과학기술부가 주관하는 국가지정연구실 사업(과제번호 M1-0104-00-0130)의 지원으로 수행되었음
† 정 회 원 : (주)이오테크닉스 과장
mjchoi@eotechnics.com
†† 비 회 원 : 한국정보통신대학원대학교 전자상거래과정
dolkongs@icu.ac.kr
††† 정 회 원 : 한국정보통신대학원대학교 공학부 교수
korykang@icu.ac.kr
dlee@icu.ac.kr
논문접수 : 2003년 1월 13일
심사완료 : 2003년 4월 7일

1. 서 론

무선 통신 환경의 발달은 유무선 통합 환경에서 실시간 협동 작업을 지원하는 그룹웨어에 대한 요구를 증가시키고 있다. 실시간 협동 작업을 위해서는 유무선 네트워크 사용자들 사이에 일관된 뷰를 유지하는 것이 중요하다. 그러나 무선 네트워크 사용자 단말기의 프로세싱 파워가 낮고, 유선 네트워크 사용자에 비해 크게 낮은 네트워크 대역폭 때문에 발생하는 전송 지연 시간 차이로 인하여 유선 네트워크 사용자와 실시간으로 일치한 뷰를 유지하는 것이 어렵다. 유선 사용자들과 무선 사용자들 사이의 메시지 전송 지연 시간의 차이를 해소하기 위한 다양한 방법들이 제안되어 왔다. 네트워크의 트래픽의 정도에 따라 선택적으로 메시지를 전달함으로써 뷰 동기화 전략을 달리하는 방법[1], XML/XSL을 통해 3차원 이미지 정보를 2차원 이미지 정보로 전환함으로써 무선 사용자에게 전달되는 메시지의 크기를 줄여 네트워크의 트래픽을 줄이는 방법[2], 압축이나 변환을 통해 무선 사용자에게 전달되는 메시지의 크기를 줄이는 방법[3] 등이 있다. 이렇게 메시지 크기를 줄이는 것에 더불어 무선 단말기의 제한된 화면 크기의 특성을 활용하여 제한적인 작업 영역 안에서 유무선 사용자들 사이의 뷰를 실시간으로 일치시키는 것이 가능하다. 즉, 무선 사용자는 공유 작업 영역의 일부분만 볼 수 있으므로 무선 사용자가 전체 작업 영역 내에서 유선 사용자와 동일한 정보를 가질 필요가 없다는 점에 착안하여 '완화된 WYSIWIS(What You See Is What I See)' 개념[4]을 활용함으로써 앞서 제안한 방법들의 효과를 크게 향상시킬 수 있다.

본 논문에서는 유무선 통합 네트워크 환경에서 완화된 WYSIWIS 개념과 네트워크 가상환경에서의 인지 모델인 포커스(focus)/님버스(nimbus) 개념[5]을 활용하여 사용자 관심 영역에 기반한 뷰 동기화 기법을 제안한다. 사용자의 '포커스'를 '사용자의 화면에 보여지는 영역에 포함되는 개체들의 집합'으로, 님버스를 '임의의 사용자가 가장 최근에 작업한 개체'로 정의한다. 또한 님버스를 세분화하여, 개별 사용자별로 '자신이 가장 최근에 작업한 개체'를 '활성(active) 님버스'라고 하며 그 외의 님버스를 '비활성(inactive) 님버스'라고 한다. 그리고, 유무선 사용자 사이에 '프락시'를 위치시켜서 무선 사용자에게 전달되는 이벤트 메시지들에 대한 스케쥴링 기능을 담당하도록 하는데, 위에서 정의한 사용자 관심 영역에 따라 이벤트 메시지 전달 우선 순위를 달리하여 무선 사용자가 현재 관심 영역에 대해서는 유선 사용자와 동기화된 뷰를 유지할 수 있도록 한다. 본 논문에서 제안하는 기법을 평가하기 위해 네트워크 시뮬레이터 ns-2를 사용해 포커스와 포커스 이외의 영역에 속하는 개체들 사이의 전송 지연 시간을 측정하였다. 그 결과, 포커스의 이벤트 메시지 전송 지연 정도는 님버스 개수의 증가에 관계없이 현격히 낮은 것으로 나타났으며 포커스가 변경되는 경우에도 제안된 기법을 사용하지 않았을 때보다 이벤트 메시지 전송 지연 시간이 낮은 것으로 나타났다. 또한, 본 논문에서 제안하는 뷰 동기화 기법을 적용하여 실시간 그룹웨어를 위한 프레임워크를 구현하였다. 프레임워크는 유선/무선 클라이언트, 프락시, 서버로 구성되는데, 서버는 미국 National Center for Supercomputing Applications(NCSA)에 의해 개발된 실시간 협동 프레임워크인 Habanero[2]를 활용하였고, 프락시와 유선 클라이언트, 무선 클라이언트를 추가로 설계하고 구현하였다. 유선/무선 클라이언트의 사용자 인터페이스로는 간단한 텍스트 에디터를 구현하였다. 프락시와 클라이언트들을 설계하는데 있어서 응용 분야에 따라 크게 영향을 받는 부분을 클라이언트의 일부 기능 요소로 제한함으로써 추후 이 프레임워크를 활용하여 다양한 실시간 그룹웨어 시스템 개발이 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 기술하고, 3장에서는 포커스/님버스 모델을 기반으로 우선 순위에 근거한 뷰 동기화 기법, 이를 위한 시스템 구조 및 성능 평가에 대해 기술하였다. 4장에서는 실시간 그룹웨어를 위한 프레임워크의 구현 현황을 자세하게 소개하고 마지막으로 5장에서 결론 및 향후 계획에 대해 기술한다.

2. 관련 연구

Whalen과 Black은 무선 네트워크에서 동기화된 협동 작업을 지원하는데 있어 제한된 대역폭으로 인한 긴 반응 시간과 패킷 충돌 및 손실로 인한 높은 에러율을 문제점으로 들고 있다[1]. 그리고 문서 개요, 점차적인 갱신, 동시성 제어, 문서 백업 및 그룹 인지 정보의 제한적 제공이라는 측면에서 여러 가지 전략을 제시하고 있다. 특히, 그들은 Calliope이라는 텍스트 기반의 그룹 에디터를 사용한 실험에서 그룹 참가자들에 대한 인지 정보의 크기가 무선 네트워크의 트래픽을 증가시키는 주요 요인임을 지적하였다. 이에 대한 해결 방법으로써 네트워크의 트래픽에 따라 인지 정보의 양을 줄이는 방법을 제시하고 있다. 인지 정보의 양을 조절하는데 있어서 사용자의 관심 영역에 대한 고려가 반영된다면 보다 높은 효과를 얻을 수 있을 것이지만, 이 방법에서는 이러

한 고려가 포함되지 않았다.

DISICPLE은 데이타 중심의 접근 방법에 기반하여 동기화된 협동 작업을 제공한다[6]. XML/XSL 데이타 변형을 통해 무선 단말기의 응용 프로그램에서 3차원 이미지를 2차원으로 변형하여 표시하는데 초점을 두고 있다[2]. DISCIPLE은 실시간 협동 작업을 수행하는 무선 사용자의 동적인 관심 변화를 고려하지 않았으며 이를 활용한 완화된 동기화의 가능성을 배제하였다.

Garnet은 Grid 시스템을 기반으로 하는 협동 프레임워크로 원래 PDA와 같은 경량의 클라이언트를 지원하지 않았으나 이후 경량급 클라이언트를 지원하기 위해 사용자에 대한 정보와 특정 장비에 대한 사양을 가지고 있으면서 적합한 프로토콜을 제공하는 어댑터를 개발하였다[3]. 어댑터는 저장된 무선 사용자의 정보에 근거하여 전달할 메시지를 압축하거나 변환하는 역할을 수행한다. 이런 경우 사용자의 관심 영역에 대한 고려가 반영되지 않으면, 사용자의 현재 관심 영역이 아닌 영역에서 발생한 이벤트 메시지를 압축하고 변환하는데 소요되는 지연 시간으로 인해 관심 영역에 속하는 이벤트 메시지가 늦게 전달되거나 전달되지 못하여 전체 협동 작업 진행에 장애가 발생할 수 있다.

Rodden은 네트워크 가상 환경에서의 인지 모델[5]의 하나인 포커스/님버스 개념을 컴퓨터를 사용한 협동 작업에 적용할 것을 제안하며, 다양한 협동 작업 응용에서의 포커스/님버스 개념을 도식화하여 제시하고 있다[7]. Wegner 등은 Rodden의 모델을 활용한 인지 서버(awareness server)를 제안하며, 다중 사용자가 참여하는 공유 문서 검토 작업을 지원하는 시스템을 구현하였다[8]. 인지 서버는 사용자들의 포커스/님버스 정보를 저장하고 있으면서 개별 사용자의 포커스 내에 포함되는 타 클라이언트의 님버스를 표시하며, 사용자의 포커스가 변경되면 변경된 포커스 내에 표시되어야 할 정보들을 해당 사용자에게 전달하는 역할을 수행한다. 그러나, 메시지 전달에 있어서는 포커스와 포커스 외부의 정보들에 대해서만 구별하고 있어 사용자 관심 영역에 대한 구별이 제한적이다.

3. 사용자 관심 영역에 근거한 뷰 동기화

3.1 실시간 협동 작업에서의 뷰 동기화를 위한 고려사항

실시간 그룹웨어 시스템에서 모든 참여자들이 동일한 뷰를 유지하기 위해서는 참여자들에게서 발생하는 이벤트들의 동기화가 중요하다. 그런데, 유무선 사용자들 간에 동기화를 이루기 위해서는 유선 사용자에 대해 구별되는 무선 사용자의 다음과 같은 특성들이 고려되어야 한다. 첫째, 무선 사용자는 유선 사용자에 비해 작은 단말기 화면을 가진다. 둘째, 무선 네트워크는 가용 대역폭이 작아 동일한 이벤트에 대해서도 유선 사용자에 비해 전송 지연 시간이 길다.

무선 사용자의 작은 단말기 화면은 각 참여자가 공유 작업 영역에 대해 일시적으로 다른 뷰를 허용하는 '완화된 WYSIWIS'를 활용할 수 있도록 하여 유무선 네트워크 환경의 차이로 인한 뷰 동기화의 어려움을 유연하게 해결할 수 있게 한다. 무선 사용자에 대해서는 현재 자신의 단말기 화면을 통해 보고 있는 영역에서 발생된 이벤트 메시지를 먼저 수신할 수 있게 함으로써, 전체 공유 작업 영역에 대한 동기화가 아닌 개별 관심 영역에서의 뷰 동기화를 이룰 수 있다. 그러나, 사용자의 관심 영역은 작업이 진행되면서 계속해서 바뀔 수 있으므로 사용자 관심 영역이 변경되는 시점에서의 뷰 동기화에 필요한 지연 시간을 줄이기 위해, 현재의 관심 영역이 아닌 영역에서 발생하는 이벤트 메시지들도 관심 영역의 이벤트 메시지보다는 낮은 비율이지만 사용자들이 지속적으로 수신할 수 있도록 한다.

3.2 사용자 관심 영역에 따른 이벤트 우선 순위 설정

개별 사용자의 관심 영역을 표현하는데 네트워크 가상 환경에서 제안된 포커스/님버스 개념을 활용한다. 특정 사용자의 '포커스'는 '사용자 화면에 보여지는 영역에 포함되는 개체들의 집합'으로 정의되며 님버스는 '임의의 사용자가 가장 최근에 작업한 개체'로 정의된다. 응용 분야에 따라 동시성 제어가 중요한 경우에는 '다른 사용자에 대한 배타적인 접근 권한이 보장되는 개체'로 정의될 수도 있다. 님버스는 개별 사용자의 관심도에 따라 다시 두 가지 종류로 구분된다. 첫째, '활성 님버스'로서, 해당 사용자가 가장 최근에 작업했거나 현재 작업하고 있는 개체로 정의된다. 둘째, '비활성 님버스'로서, 활성 님버스 외의 님버스들이 해당된다. 그림 1은 그룹 에디터에 적용된 포커스/님버스 모델을 보여준다. 전체 공유 작업 영역이 그림 1의 d)에서 보여지고 있는데, 사용자의 화면 크기가 다르고 작업 영역이 다르므로, a), b), c)에서 보이는 바와 같이, 각 사용자는 서로 다른 포커스를 갖게 된다. 그리고, 그림 1에서 기울임체의 개체가 개별 사용자의 활성 님버스를 나타내고 있는데, 자신의 활성 님버스가 다른 사용자에게는 비활성 님버스로 간주될 수 있다. 그리고, 어떤 사용자가 현재 공유 작업 영역에서 작업을 하고 있지 않다면, 이전에 작업했던 영역이 현재의 활성 님버스이고 동시에 다른 사용자가 이 영역에 대해 편집 작업을 하고 있을 수 있으므로, 동시에 둘 이상의 사용자가 하나의 님버스를 활성 님버

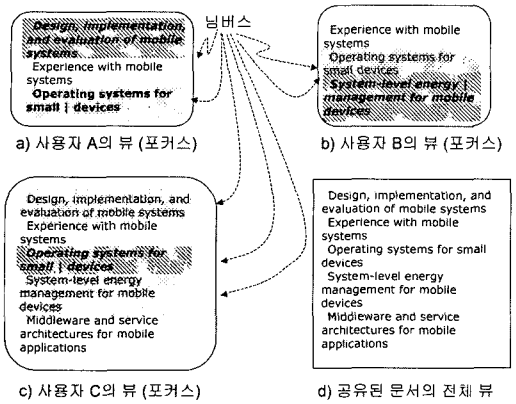


그림 1 그룹 에디터에 적용한 포커스/님버스 모델

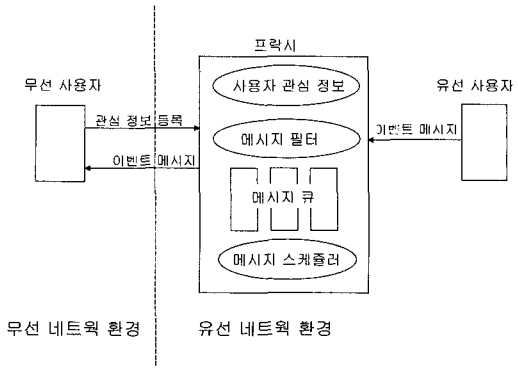


그림 2 사용자 관심 영역에 근거한 뷰 동기화 시스템 모델

스로 간주할 수 있다.

임의의 이벤트가 수신할 사용자의 어느 관심 영역에서 발생했는지에 따라 메시지의 전달 우선 순위를 달리 설정한다. 우선 순위의 종류는 응용 분야에 따라 다양하게 고려될 수 있다. 작게는 '높음'과 '낮음'의 두 가지 우선 순위를 부여할 수도 있으며, '높음', '중간', '낮음' 등으로 세 가지 우선 순위로 고려할 수 있고 더 다양한 단계의 우선 순위를 고려할 수 있다. 두 가지 우선 순위의 경우는, 사용자의 포커스에 속하는 이벤트는 '높음', 그 외의 영역에서 발생한 이벤트는 '낮음'의 우선 순위가 설정된다. 세 가지 우선 순위의 경우는, 수신자의 활성 님버스에서 발생한 이벤트는 '높음', 수신자의 포커스 내에 속하는 비활성 님버스에서 발생한 이벤트는 '중간', 그 외의 영역에서 발생한 이벤트는 '낮음'으로 우선 순위가 부여된다.

3.3 우선 순위에 근거한 이벤트 메시지 스케쥴링

실제 무선 클라이언트로의 이벤트 메시지가 전달되기 위해, 3.2절에서 설명한 우선 순위에 근거하여 메시지 스케쥴링 작업이 이루어진다. 유무선 클라이언트 사이에 프락시가 위치하여 이벤트 메시지 전달의 중개자 역할을 수행한다. 프락시는 유선 사용자로부터 전달되는 이벤트 메시지의 손실을 최소화하기 위해 유선 네트워크 상에 위치하도록 하며, 무선 사용자와의 효과적인 메시지 교환을 위해 무선 네트워크에 근접하게 위치하도록 한다. 그림 2에서는 프락시를 이용한 뷰 동기화 시스템 모델을 보이고 있다. 프락시는 무선 사용자의 관심 영역 정보를 유지하고 있고, 3.2절에서 기술하고 있는 기준에 따라 이벤트 메시지들에 대해 우선 순위를 부여하고, 우선 순위에 따라 이벤트 큐에 저장한다. 정기적으로 메시지 큐 내에 이벤트 메시지의 유무를 조사하고 가장 높은 우선 순위를 가진 큐에 저장된 이벤트 메시지들부터 우선적으로 무선 사용자에게 전송한다. 그리고, 프락시에서 사용자의 관심 정보를 획득하는데 소요되는 처리 부담을 최소화하기 위해, 무선 사용자의 자신의 관심 영역을 명시적으로 등록하도록 한다.

사용자의 관심 영역은 공유 작업이 진행되면서 변경될 수 있다. 비활성 님버스로 간주되었던 님버스가 활성 님버스로 변경될 수도 있으며, 포커스가 변경되어 포커스 밖에 위치하는 것으로 간주되던 님버스가 포커스 안에 포함되는 경우도 있다. 사용자의 관심 영역이 변경되면 해당 사용자가 변경된 관심 영역에서의 공유 작업을 진행하기에 앞서 타 사용자와의 동기화가 우선 이루어져야 한다. 즉, 신규 영역에서 관심 영역 변경 전에 발생한 이벤트 메시지들이 사용자에게 신속하게 전달될 수 있어야 한다. 이 때 일시에 전송해야 하는 이벤트 메시지의 수가 많으면 뷰 동기화에 걸리는 시간이 길어지고 전체 공유 작업에 장애를 유발할 수 있다. 이러한 문제 상황을 방지하기 위해, 운영 체제나 네트워크 라우터 장비 등에서 사용되는 스케쥴링 알고리즘의 한 종류인 weighted round robin[9]에서 사용하는 '가중치'를 사용한다. 여기서의 '가중치'는 일시에 전송 가능한 이벤트 메시지의 수를 의미한다. 각 메시지 큐마다 우선 순위의 순서대로 가중치를 할당한다. 가장 높은 우선 순위를 갖는 메시지 큐에 대해서는 가장 많은 가중치를 부여하며, 가장 낮은 우선 순위를 갖는 메시지 큐에는 가장 작은 가중치를 부여한다. 결과적으로 사용자의 관심도가 가장 높은 활성 님버스에 속하는 이벤트 메시지가 단위 시간 내에 보다 많이 전달될 수 있도록 하면서, 더불어 활성 님버스 혹은 포커스 변경 시에 발생하는 뷰 동기화의 부담을 줄일 수 있도록 한다.

3.4 뷰 동기화 기법 성능 평가

구체적인 구현에 앞서 네트워크 시뮬레이터 ns 2를 사용하여 뷰 동기화 기법의 성능을 평가한다. 시뮬레이션을 위한 네트워크 토폴로지는 그림 3과 같으며, 토폴로지의 중앙에 프락시와 서버가 위치하고 그 양쪽으로 무선 클라이언트와 유선 클라이언트가 위치하도록 한다.

유선 클라이언트만이 이벤트를 생성하되 각 유선 클라이언트는 서로 다른 영역에서 작업하는 것을 가정한다. 이벤트 메시지의 크기를 128 bytes로 하고, 무선 클라이언트로의 회선 대역폭보다 많은 트래픽이 발생하도록 각 유선 클라이언트는 초당 4 개의 이벤트를 생성한다. 그리고, 실제 편집 작업 시에 연속적인 타이핑 후 짧은 휴식을 갖는 사용자 행동을 반영하기 위해 10 번의 연속적인 이벤트 발생과 3.5 초 동안 중지하는 과정을 반복한다. 무선 클라이언트는 포커스 내에 하나의 님버스를 갖는 것으로 가정하며, 포커스는 매 10 초마다 변경되는 것으로 한다. 이 시뮬레이션에서는 두 단계의 우선 순위를 사용하며, 유선 클라이언트에서 발생된 이벤트 메시지가 무선 클라이언트에게 전달되는 데까지 걸리는 전송 지연 시간을 측정한다. 참여하는 유선 클라이언트의 수를 증가시키면서, 즉 공유 작업 공간 내의 님버스 수를 증가시키면서, 동일한 조건에서 실험을 10 번씩 반복하여 메시지의 전송 지연 시간의 평균값을 계산한다.

그림 4는 무선 클라이언트 W1에서 측정한 포커스에서의 평균 메시지 전송 지연 시간을 보인다. 결과의 'Nofiltering'은 본 논문에서 제안하는 기법을 적용하지 않은 경우를 나타내며, 전체 공유 문서 내에 존재하는 님버스의 수가 증가함에 따라 지연 속도는 급격히 증가함을 보여준다. 그림 4(a)에서 보는 바와 같이 님버스의 수가 8인 경우 평균 전송 지연 시간이 1.8 초에 이른다. 'Filtering'은 사용자의 관심도에 따라 메시지의 우선 순위를 부여한 경우를 나타내며 평균 전송 지연 시간은 0.3 초를 밑돌며 님버스의 수 증가에도 큰 차이를 보이지 않는다. 그림 4(b)는 무선 클라이언트가 포커스를 변경한 경우 변경된 포커스에 해당하는 이벤트 메시지 중 프락시에서 전달되지 않고 큐에 저장되어 있던 메시지들이 모두 해당 무선 클라이언트에게 전달되는데 걸리는 평균 전송 지연 시간을 나타낸다. 새로운 포커스에 속한 이벤트들은 프락시에서 낮은 우선 순위로 처리되어 큐에 저장되어 있는 시간이 길었기 때문에 a)에서 관찰된 것에 비해 높은 지연 시간을 갖는다. 그럼에도 불구하고, 가중치를 사용한 메시지 스케줄링을 하기 때문에 제안된 뷰 동기화 기법을 사용하지 않았을 때보다 낮은 지연 시간을 보인다.

결론적으로 사용자 관심 영역에 근거한 뷰 동기화 기법은 사용자 관심도가 높은 영역에 대해서는 낮은 전송

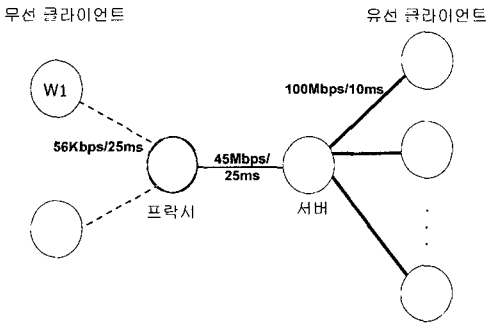
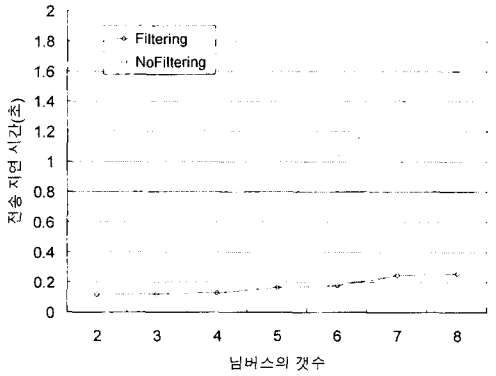
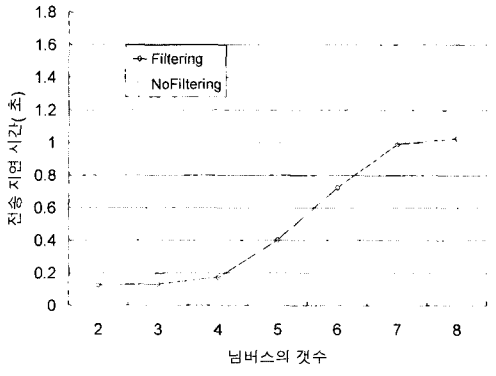


그림 3 시뮬레이션을 위한 네트워크 토폴로지



(a) 무선 클라이언트 W1의 포커스에서의 평균 메시지 전송 지연 시간



(b) 무선 클라이언트 W1에서 포커스 변경 시 뷰 동기화되는 동안의 평균 메시지 전송 지연 시간

그림 4 뷰 동기화 기법 성능 평가 결과

지연 시간을 보장함으로써 유무선 사용자 사이의 실시간 협동작업을 가능하게 하고, 사용자의 관심 영역이 변경되어도 일시적으로 뷰 동기화를 위해 시간이 지체될 수 있지만 뷰 동기화 기법을 사용하지 않는 경우보다 실시간 협동 작업을 효과적으로 유지할 수 있게 한다.

4. 뷰 동기화 기법을 지원하는 실시간 그룹웨어 프레임워크 구현

4.1 프레임워크 구조

NCSA에서 개발한 협동 작업 응용 프로그램 개발 프레임워크인 Habanero[2]를 활용하여, 본 논문에서 제안하는 뷰 동기화 기법을 지원하는 유무선 통합 환경을 위한 실시간 그룹웨어의 프레임워크를 구현하였으며 다음과 같이 구성되었다. 응용에 영향을 주는 정보를 클라이언트에서 처리하도록 하였으며, 클라이언트 내에서도 일반적으로 사용될 수 있는 기능 요소들을 확보할 수 있도록 설계하였다. 또한 시스템 구성 요소들 사이에 교환하는 메시지를 정의하는데 있어서도, 사용자 관심 영역을 등록하거나 무선 사용자 정보를 관리하기 위한 메시지들을 구별함으로써 본 논문에서 개발한 프레임워크를 참조로 하여 뷰 동기화 기법을 적용한 응용 분야를 확대해 나갈 수 있도록 한다. 그리고, 사용자의 관심 영역 정보를 관심 영역 내에 포함된 님버스 식별자의 집합 형태로 표현하며, 개별 이벤트 메시지가 속한 님버스 식별자와 님버스 내에서의 위치 정보를 클라이언트에서 파악하여 프락시에게 전달하여, 프락시는 이벤트 메시지 내에 포함된 님버스 식별자만을 사용하여 이벤트의 위치 정보를 파악할 수 있도록 하여, 프락시가 무선 클라이언트의 특성에 무관하게 메시지의 우선 순위를 부여할 수 있게 한다.

프레임워크의 전체 구조는 그림 5와 같이 서버, 프락시, 유선/무선 클라이언트로 구성된다. 서버는 Habanero

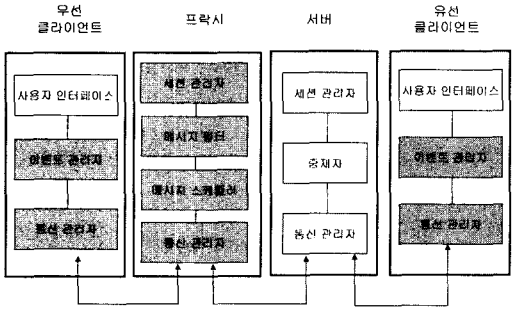


그림 5 프레임워크의 구조 및 기능 요소

에서 제공하는 것을 활용하였으며, 유선 클라이언트와 무선 클라이언트, 프락시에 대하여 본 연구에서 기능 요소들을 설계하고 구현하였다. 그림 5에서 짙은 색으로 표시된 부분이 뷰 동기화 기법을 지원하기 위한 기능 요소들이며, 본 프레임워크를 활용하여 추가적인 응용 개발에 있어서 재사용이 가능한 기능 요소들이다. 구성 요소들의 기능을 간략히 살펴보면 다음과 같다. 첫째, '서버'는 '세션 관리자'와 '중재자', '통신 관리자'로 구성되며, 진행 중이거나 예약된 세션 정보를 유지하고 사용자에서 발생하는 이벤트들의 동시성 제어를 위해 이벤트 메시지에 일련 번호를 부여하는 역할을 수행한다. 둘째, '프락시'는 본 논문에서 제시하는 동기화 기법을 지원하는 핵심 요소로서, '세션 관리자'와 '메시지 필터', '메시지 스케줄러', 그리고 '통신 관리자'로 구성되며 각 기능 요소의 세부 기능은 다음과 같다.

- 세션 관리자: 세션 정보와 각 세션에 참여한 무선 클라이언트들을 식별할 수 있는 정보를 관리한다. 세션 정보에 대해서는 서버와 동기를 유지한다.
- 메시지 필터: 개별 무선 클라이언트의 포커스와 활성 님버스 정보, 그리고 공유 작업 문서 내에 존재하는 님버스 목록을 유지 관리한다. 개별 이벤트 메시지에 대해 이벤트 메시지 내에 표시된 님버스 식별자와 포커스 내에 등록된 님버스 목록을 비교하고, 활성 님버스의 식별자와 비교하여 해당 이벤트의 우선 순위를 결정하여 해당 메시지 큐에 메시지를 저장한다.
- 메시지 스케줄러: 메시지 큐에 저장된 메시지들의 전송 순서를 각 큐의 가중치에 따라 조절하는 역할을 한다.
- 통신 관리자: 서버와 무선 클라이언트로부터 메시지를 송수신하는 역할을 수행한다.

마지막으로, '유선 클라이언트'와 '무선 클라이언트'는 그림 5에서 보이는 바와 같이, '사용자 인터페이스'와 '이벤트 관리자', '통신 관리자'로 구성되며 각 기능 요소의 세부 기능은 다음과 같다.

- 사용자 인터페이스: 미리 정해진 사용자 행동에 따라 발생된 이벤트를 이벤트 관리자에게 전달한다. 또한 님버스들을 화면 내의 다른 개체들과 구별할 수 있도록 다른 효과를 표시하는 등의 간단한 인지 기능도 제공한다.
- 이벤트 관리자: 사용자 인터페이스로부터 전달된 이벤트를 받아 메시지로 변형하여 통신 관리자를 통해 전달하거나 다른 클라이언트에 의해 발생된 이벤트 메시지를 처리하여 필요에 따라 사용자 인터

페이스가 화면에 표시할 수 있도록 한다.

- 통신 관리자: 이벤트 관리자가 전송 요청한 메시지를 프락시에게 전송하거나 프락시로부터 전달된 메시지를 이벤트 관리자에게 전달한다.

그림 6은 프레임워크 내 구성 요소들 사이에 교환하는 메시지의 흐름을 보이고 있는데, 이를 네 가지 과정으로 나누어 설명할 수 있다. 첫째, 무선 클라이언트가 세션에 참가하는 과정이다. 프락시는 서버에게 요청하여 *SessionInfo*를 통해 현재 진행 중인 세션 정보를 획득한다. 세션에 참여하기 전에 무선 클라이언트는 서버에 *SessionInfoRequest*를 보냄으로써 현재 진행 중인 세션의 정보를 요구한다(단계 1). 그 응답으로 클라이언트는 현재 진행 중인 세션의 목록과 참여자 정보를 포함한 *SessionInfoResponse*를 수신하게 된다(단계 2). 무선 클라이언트는 자신이 참여할 세션을 결정하여 *JoinSession*을 프락시에게 전송한다. 프락시는 이를 서버에서 이해할 수 있는 *NewMemberJoin*으로 변형하여 세션에 새로운 참가자가 있음을 알리고, 그 응답으로 서버로부터 *RunningApplicationList*를 수신하는데 이 메시지에 세션 내에서 사용되는 응용 프로그램의 목록이 포함된다. 서버는 *NewMemberNotification*으로 다른 유선 클라이언트에게 새로운 참가자에 대한 정보를 알린다(단계 3-7). 무선 클라이언트가 세션 내 응용 프로그램 중에서 사용하고자 하는 응용 프로그램을 결정하여 *SelectApplication*으로 프락시에게 알리면, 프락시로부터 *Initiation*을 받게 되는데, 이 메시지에는 초기 윈도우에 표시해야 할 컨텐츠와 공유 작업 공간 내에 존재하는 전체 님버스 목록이 포함된다(단계 8-9). 둘째, 공유 작업 도중 사용자의 포커스가 변경되는 과정이다. 응용 프로그램이 시작할 때나, 사용자의 커서가 이동되거나 윈도우 스크롤링 정지 등으로 인해 포커스가 변경될 때마다 무선 클라이언트는 이동된 포커스 내에 포함되는 님버스 목록을 전달하는 *Subscribe*를 프락시에게 전송한다(단계 10). 이 때 프락시가 해당 무선 클라이언트의 관심 영역 정보를 확보하게 된다.

셋째, 공유 작업 공간 내 특정 개체를 수정하는 과정이다. 무선 클라이언트가 자신이 작업하는 영역을 등록하고 선택한 객체에 대한 배타적인 접근 권한을 받기 위해 프락시에게 *ClaimNimbusRequest*를 송신한다. 프락시는 이를 서버에서 관리하는 동시성 제어 기법에 대응하도록 변환하여 서버에게 배타적인 접근 권한을 요청하게 되고, 프락시는 서버로부터 수신한 *LockResponse*에 반영된 접근 권한 허용 여부를 *ClaimNimbusResponse*에 실어서 무선 클라이언트에게 송신한다. 그리고, 이 때 타 무선 클라이언트들에게도 전송하여 님버스 목록을 갱신할 수 있도록 한다(단계 11-15). 사용

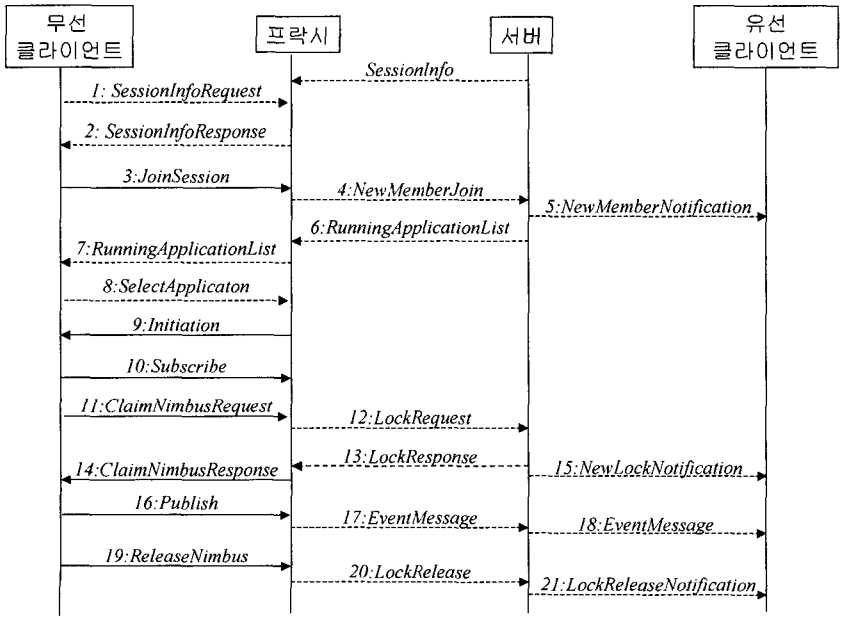


그림 6 프레임워크 내 메시지 흐름도

자에 의해 님버스에서 변경이 일어나면 무선 클라이언트가 *Publish*를 프락시에게 전달하고 프락시는 이를 *EventMessage*의 형식으로 변환하여 서버에게 전달한다. 그리고, 서버는 해당 메시지에 일련 번호를 할당하여 다시 프락시에게 전달하고, 또한 다른 유선 클라이언트들에게 동일한 *EventMessage*를 전달한다(단계 16-18). 그림 6에서는 표시되지 않았지만, 프락시는 수신된 *EventMessage*의 원래 송신자인 무선 클라이언트를 제외한 나머지 무선 클라이언트들에게 *EventMessage*를 전달한다. 이 때 모든 무선 클라이언트들에게 동시에 메시지를 전달하는 것이 아니고, 본 논문에서 제안하는 메시지 스케줄링 기법에 따라 서로 다른 시점에 무선 클라이언트들에게 전달한다.

넷째, 활성 님버스를 변경하는 과정이다. 활성 님버스의 변경은 포커스의 변경과 더불어 발생할 수 있고, 포커스는 고정된 상태에서 사용자가 커서를 움직여서 새로운 개체 등을 선택하는 경우에 발생한다. 활성 님버스를 변경하기 위해서는 우선 기존의 활성 님버스를 해지하는 과정이 진행되어야 한다. 무선 클라이언트의 관심 영역 정보를 수정하도록 프락시에게 통지하기 위해 활성 님버스의 해지가 필요하고 또한 해당 님버스에 대해 배타적인 접근 권한이 확보되어 있는 경우 이를 해지하기 위해서 필요하다. 활성 님버스의 해지 과정은 무선 클라이언트가 *ReleaseNimbus*를 프락시에게 전송하는 것으로 시작된다. 프락시는 해당 무선 클라이언트의 활성 님버스 정보를 수정하고, *LockRelease*를 서버에 전송한다. 서버는 다시 유선 클라이언트들에게 *LockReleaseNotification*을 전송하여 배타적인 접근 권한이 해지되었음을 통지한다(단계 19-21). 그림 6에 표시되지 않았지만, 나머지 무선 클라이언트들도 프락시가 전달하는 *ReleaseNimbus*를 수신함으로써 활성 님버스 해지를 인지하게 되고, 각자가 유지하고 있는 님버스 목록을 수정한다. 새로운 활성 님버스의 설정은 앞서 설명한 세 번째 과정에 따라 진행된다.

표 1 뷰 동기화 기법을 지원하기 위한 메시지

메시지 명칭	변수	설명
JoinSession	• SessionID : 세션 식별자 • ClientID : 무선클라이언트 식별자	• 방향: 무선 클라이언트 → 프락시 • 용도: 세션에 참가 요청
Initiation	• Contents: 사용자 화면에 표시될 데이타 • NimbusList: 사용자의 포커스 내에 포함되는 님버스 식별자 목록	• 방향: 프락시 → 무선 클라이언트 • 용도: 응용프로그램 초기화를 위한 컨텐츠와 님버스 목록 전달
ClaimNimbusRequest	• SessionID: 세션 식별자 • ClientID: 무선클라이언트 식별자 • PrevNimbusID: 공유 작업 공간 내에서 신규 님버스보다 앞에 위치한 님버스 식별자 • Length: 신규 님버스의 길이	• 방향: 무선 클라이언트 → 프락시 • 용도: 선택한 개체를 활성 님버스 등록, 접근 권한 요청
ClaimNimbusResponse	• SessionID: 세션 식별자 • NewNimbusID: 신규 님버스 식별자	• 방향: 프락시 → 무선 클라이언트 • 용도: ClaimNimbusRequest에 대한 응답 메시지
ReleaseNimbus	• SessionID: 세션 식별자 • NimbusID: 해지하고자 하는 님버스의 식별자	• 방향: 무선 클라이언트 → 프락시, 프락시 → 무선 클라이언트 • 용도: 활성 님버스 해지, 접근 권한 해지에 대한 요청 및 타 무선 클라이언트의 님버스 해지 통지
Publish	• SessionID: 세션 식별자 • NimbusID: 이벤트가 발생한 님버스 식별자 • TargetApp: 이벤트가 발생한 응용 프로그램 • EventPosition: 님버스 내에서의 이벤트 위치 • Contents: 이벤트의 내용	• 방향: 무선 클라이언트 → 프락시 • 용도: 활성 님버스에서 발생한 변경 정보 메시지
Subscribe	• SessionID: 세션 식별자 • ClientID: 무선 클라이언트 식별자 • NimbusList: 새로운 포커스에 속하는 님버스 식별자 목록	• 방향: 무선 클라이언트 → 프락시 • 용도: 포커스 내의 님버스 목록 전달
LeaveSession	• SessionID: 세션 식별자 • ClientID: 무선 클라이언트 식별자	• 방향: 무선 클라이언트 → 프락시 • 용도: 세션에 탈퇴 통지

그림 6에서 언급된 메시지들 중 본 논문에서 제안하는 뷰 동기화 기법 지원을 위해 주요하게 사용되는 메시지들과 프락시가 무선 클라이언트의 멤버쉽 관리를 위해 필요한 메시지들의 종류와 메시지 내에 함께 전달되는 정보, 그리고 해당 메시지의 송신자와 수신자를 정리하여 표 1에서 자세하게 기술하고 있다.

4.2 프락시 구현

프락시는 Microsoft Windows 환경에서 Sun 사의 JDK version 1.3을 사용하여 개발되었고, 그림 5에서 보이는 바와 같이, '세션 관리자'와 '메시지 필터', '메시지 스케쥴러', 그리고 '통신 관리자'의 네 가지 기능 요소들을 갖는다. '세션 관리자'는 서버의 세션 정보와 동기를 유지하고 무선 클라이언트에 대한 정보를 저장하며, ProxySession과 ProxyManager로 구성된다. ProxySession은 서버에서 유지하는 것과 같은 세션 정보를 복제하여 가진다. ProxyManager는 무선 클라이언트의 세션 가입 요청을 수신하여, 세션 내 참여 무선 클라이언트의 IP 주소, 접속 포트 번호 등과 같은 식별자로서 사용될 수 있는 정보들을 유지한다. 그리고, 무선 클라이언트의 세션 가입 요청을 서버에게 전달한다. '메시지 필터'는 개별 무선 클라이언트에 대응하는 NimbusManager와 NimbusQueue들로 구성된다. NimbusManager는 무선 클라이언트의 포커스 내에 포함되는 님버스들의 식별자 목록과 활성 님버스의 식별자를 유지한다. 무선 클라이언트가 전송하는 *Subscribe* 메시지에 의해 해당 사용자의 포커스 정보 내에 포함되는 님버스 식별자들이 변경되고, *ClaimNimbusRequest*, *ReleaseNimus* 등의 메시지들에 의해 활성 님버스 정보가 변경된다. NimbusQueue는 우선 순위의 종류만큼 만들어진다. 개별 NimbusQueue에는 해당 우선 순위와 가중치가 지정된다. 본 프레임워크에서는 높음, 중간, 낮음의 세 가지 우선 순위를 지정하고, 각 우선 순위에 대해 10, 5, 2의 가중치를 부여한다. '메시지 스케쥴러'에는 IOHandler가 속하며, NimbusQueue를 주기적으로 점검하여 가중치에 따라 연속적으로 메시지들을 전송한다. '통신 관리자'는 서버와 무선 클라이언트로부터 메시지를 송수신하는 역할을 수행하는데, IOHandler, ProxyCommManager, ProxyServerListener로 구성된다. 무선 클라이언트들과의 메시지 송수신을 위해 IOHandler를 사용하고, ProxyServerListener는 서버와의 메시지 송수신을 위해 사용한다. ProxyCommManager는 개별 무선 클라이언트들을 대상으로 만들어지는 IOHandler를 총괄하는 역할을 수행하고, 프락시 내 다른 기능 요소들에게 메시지를 전달하는 역할을 담당한다.

그림 7은 무선 클라이언트에서 발생된 메시지를 프락시에 처리하는 과정과 무선 클라이언트에게 메시지를 전달하기 전의 처리 과정을 보여준다. 그림 내의 화살표는 객체의 메소드 호출을 의미하며, 화살표 위에 표시된 것은 호출되는 메소드 명칭이다. 다만, 프락시 외부로메시지가 송신되거나 외부로부터 메시지가 전달되는 경우

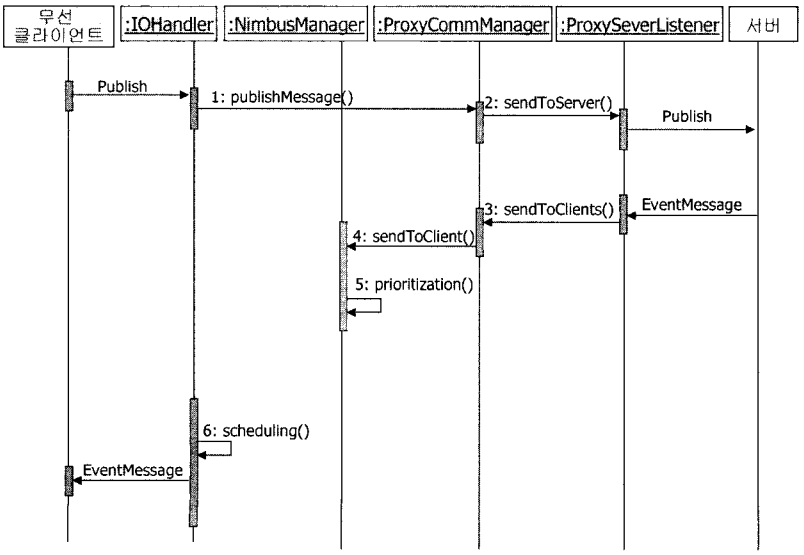


그림 7 이벤트 메시지 전송 및 필터링 과정

에는, 해당 메시지의 명칭을 적고 있다. 통신 관리자 내 IOHandler가 무선 클라이언트로부터 메시지를 받아서 ProxyCommManager에게 전달한다(단계 1). ProxyCommManager는 메시지를 서버로 전송하기 위해 ProxySeverListener에게 요청한다(단계 2). 서버로부터 전송된 메시지는 ProxySeverListener를 통해 ProxyCommManager에게 전달되고, 등록된 개별 무선 클라이언트 메시지들을 위해 만들어진 이벤트 필터 내의 NimbusManager에게 전달된다(단계 3-4). NimbusManager는 NimbusQueue 목록을 관리하며 이벤트 메시지 내에 표시된 님버스 식별자에 따라 우선 순위를 결정하고 해당 NimbusQueue에 메시지를 저장한다(단계 5). IOHandler는 주기적으로 NimbusQueue들을 점검하여 저장되어 있는 메시지들에 대해 통해 무선 클라이언트에 전송한다(단계 6).

4.3 클라이언트 구현

그림 5에서 보이는 바와 같이 클라이언트는 유선 클라이언트와 무선 클라이언트로 나뉘어진다. 무선 클라이언트는 Microsoft 사의 Windows CE version 2.1에서 Personal Java를 사용하여 개발되었고, 유선 클라이언트는 무선 클라이언트와 유사한 구조를 갖는데, 개발 환경이 Microsoft 사의 Windows 2000에서 Java SDK v1.3에서 개발되었다는 것과 윈도우의 크기가 무선 클라이언트에 비해 크다는 점이 다르다. 다만, 본 논문의 주제가 무선 사용자를 위한 뷰 동기화 기법에 초점을 맞추고 있으므로, 이 절에서는 무선 클라이언트의 구현에 대해서 주로 기술하도록 한다.

무선 클라이언트는 '사용자 인터페이스'에 해당하는 AppGUI와 '이벤트 관리자'에 속하는 NimbusList, EventHandler, 그리고 '통신 관리자'에 속하는 CommHandler로 구성된다. 본 구현에서는 AppGUI로 간단한 텍스트 에디터를 구현하였다. AppGUI는 사용자의 키 입력, 윈도우 스크롤링 등 사용자의 행동에 따라 발생되는 이벤트를 EventHandler에게 전달한다. 그리고, 사용자의 작업을 돕기 위해 윈도우 내에서 님버스에 해당하는 부분의 색을 달리하여 구별할 수 있도록 하는 기능을 제공한다. NimbusList은 사용자의 활성 님버스 외에도 공유 작업 공간에 존재하는 님버스 전체 목록을 유지한다. 이는 사용자에 의해 발생하는 이벤트나 외부로부터 수신된 이벤트의 정확한 위치 정보를 계산하기 위한 자료로써 사용된다. EventHandler는 AppGUI로부터 전달된 이벤트를 받아 CommHandler를 통해 프락시에게 전달하고, CommHandler를 통해 프락시로부터 수신된 다른 클라이언트의 이벤트들에 대해서 AppGUI를 통해 화면에 표시될 수 있도록 한다. CommHandler는 프락시로 이벤트를 포함한 메시지를 전달하거나 프락시로부터 메시지를 수신하는 역할을 수행한다.

그림 8은 본 논문에서 구현한 무선 클라이언트 내의 이벤트와 메시지의 처리 순서를 보여 준다. 그림 내의 화살표는 객체의 메소드 호출을 의미하며, 화살표 위에 표시된 이름은 호출되는 메소드 명칭을 나타낸다. 무선 클라이언트에서의 처리 과정은 크게 네 가지로 나누어 볼 수 있다. 첫째, 사용자가 윈도우에 표시된 개체를 선택하는 경우이다. AppGUI는 커서의 위치 값으로부터 라인의 범위를 계산해 내서 입력 이벤트를 EventHandler에게 전달한다. 이벤트워 위치 정보를 계산하는 것은 응용 분야에 따라 달라질 수 있다. EventHandler는 선택된 객체가 다른 사용자에 의해 작업 중인 객체인지의 여부를 NimbusList에게 요청하여 확인한다(단계 1-2). 만약 다른 사용자에 의해 작업 중이지 않다면 이전의 활성 님버스를 해지하고 NimbusList의 님버스 식별자 목록에 새로운 님버스를 등록한다. 님버스를 해지하고 설정하는 과정은 4.2 절에서 설명하고 있는 과정을 따라 진행된다(단계 3-6). 둘째, 사용자가 AppGUI를 사용하여 선택한 개체를 수정하는 경우이다. 본 논문의 구현에서는 키를 입력하는 경우가 해당하고, 이러한 이벤트 역시 프레임워크가 적용되는 응용 분야에 따라 달라질 수 있다. 키 입력이 발생하면 AppGUI가 EventHandler에게 키 입력 이벤트를 전달한다. EventHandler는 NimbusList에게 요청하여 해당 님버스 식별자와 님버스 내에서 발생 이벤트의 상대적 위치값을 계산하여 이 님버스 식별자와 상대 위치 정보를 포함한 *Publish* 메시지를 CommHandler를 통해 프락시에게 전송한다(단계 7-9). 셋째, 타 클라이언트로부터 발생된 이벤트가 전달되는 경우이다. CommHandler로부터 수신된 다른 클라이언트의 이벤트에 대해 이벤트 내에 포함된 님버스 식별자와 위치 상대값을 NimbusList에게 전달하여 발생한 이벤트의 위치를 계산해 내고, 이에 따라 화면에 표시하도록 AppGUI에게 요청한다(단계 10-12). 넷째, 사용자가 포커스를 변경하는 경우이다. 본 논문의 AppGUI를 사용하는 경우에는, 윈도우를 위 아래로 스크롤링하다가 멈추는 경우 등이 해당한다. 이때, 포커스가 변경되면, AppGUI가 변경된 포커스 위치 정보와 해당 이벤트를 EventHandler에게 통지한다. EventHandler는 NimbusList에서 해당 포커스 내의 님버스 목록을 재계산한 후 CommHandler을 통해 *Subscribe* 메시지를 프락시에게 전송한다(단계 13-15).

이러한 처리 과정에서 볼 수 있듯이 EventHandler와

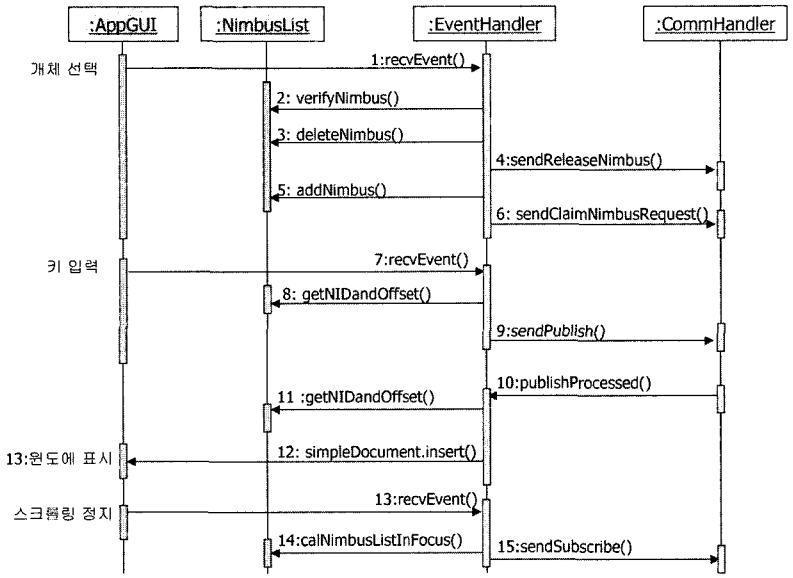


그림 8 무선 클라이언트의 내부 처리 순서도

CommHandler는 무선 클라이언트의 응용 분야에 독립적인 기능을 수행한다. 그룹 에디터가 아닌 다른 분야의 응용 시스템을 개발하고자 하는 경우 AppGUI와 NimbusList만을 개별 응용 분야의 특성에 맞게 수정하여 설계함으로써 본 논문에서 제안하는 기법을 활용한 클라이언트 개발이 가능하다.

5. 결론과 향후 계획

본 논문에서는 완화된 WYSIWIS와 포커스/님버스 개념을 활용하여 유무선 사용자 간에 효과적인 뷰 동기화 기법을 제시하였다. 클라이언트의 작업 영역은 사용자의 관심 정도에 따라 포커스와 활성 님버스, 그리고 비활성 님버스로 나뉘며 모든 이벤트 메시지들은 이 기준에 근거하여 전송 우선 순위가 결정된다. 이로써 무선 클라이언트는 다른 메시지보다 가장 높은 관심을 갖는 활성 님버스의 이벤트 메시지를 우선적으로 수신하게 하여, 활성 님버스나 포커스 영역에 있어 유선 클라이언트와 동일한 뷰를 유지하게 한다. 구현에 앞서 제안하는 뷰 동기화 기법의 성능을 평가하기 위해 네트워크 시뮬레이터를 활용하여 유무선 사용자 사이의 메시지 전송 지연 시간을 측정하였다. 시뮬레이션 결과에 따르면, 님버스의 증가나 포커스의 변경에도 불구하고 사용자 관심 영역에 근거한 뷰 동기화 방법이 무선 사용자의 관심 영역 내에서는 유무선 사용자들 사이의 실시간 동기화 작업을 가능하게 한다. 또한 제안된 뷰 동기화 기법을 적용하여 실시간 그룹웨어를 위한 프레임워크를 구현하였다. 이를 위해 유선 네트워크 환경에서 협동 작업을 위한 응용 프로그램 개발 환경으로 사용되는 Habanero를 활용하였으며 프락시와 유선/무선 클라이언트들을 추가로 구현하였다. 본 논문에서는 간단한 텍스트 에디터의 형식으로 클라이언트를 개발하였지만, 클라이언트의 '사용자 인터페이스'내의 구성 요소들을 개별 응용 분야에 맞게 수정함으로써 본 프레임워크를 활용하여 뷰 동기화 기법을 지원하는 실시간 그룹웨어 개발이 가능하다.

향후 과제로, 제안된 방법을 통해 전달된 메시지 전송 지연 시간을 실제 네트워크 환경에서 측정해 봄으로써 그 실용성을 검증할 예정이며, 무선 사용자의 수가 증가함에 따라 부가되는 프락시의 처리 부담을 최소화하기 위한 방법론을 추가로 고려하여 제안하는 기법의 확장성을 높일 예정이다.

참 고 문 헌

[1] Whalen, T. and Black, J. P., "Adaptive Groupware for Wireless Networks," Proc. of the second IEEE Workshop on Mobile Computing Systems and Applications(WMCSA 99), pp.20-29, Feb. 1999.

[2] Chabert, A., Grossman, E., Jackson, L. S., Pietrowiz, S. R. and Seguin, C., "Java object sharing in Habanero," Comm. ACM, Vol. 41, No. 6, pp.69 76, Jun. 1998.

[3] Ko, S., Kim, K., Lee, S. and Oh, S., "Status of Hand-Held Interface to Garnet Collaborative Environment," Proc. of the third International Conference on Internet Computing, Jun. 2002.
[4] Roth, J. and Unger, C., "Using handheld devices in synchronous collaborative scenarios," Proc. of the second International Symposium on Handheld and Ubiquitous Computing, pp. 187 199, Sep. 2000.
[5] Benford, S. and Fahlen, L., "A Spatial Model of Interaction in Large Virtual Environments," Proc. of Third European Conference on Computer Supported Cooperative Work (ECSCW 93), Sep. 1993.
[6] Marsic, I., "Adaptive Collaboration for Wired and Wireless Platforms," IEEE Internet Computing, Vol. 5, No. 4, pp. 26-35, Aug. 2001.
[7] Rodden, T., "Populating the Application: A Model of Awareness for Cooperative Applications," Proc. of ACM CSCW'96, pp. 87-96, Nov. 1996.
[8] Wegner, L., Ahmad, M., Schmidt, Ch., Frohlich, S. and Evers, W., "A Collaborative Infrastructure for Mobile and Wireless Systems," Proc. of NFS Workshop on an Infrastructure for Mobile and Wireless Systems, Oct. 2001.
[9] Katevenis, M., Sidiropoulos, S. and Courcoubetis, C., "Weighted round robin cell multiplexing in a general-purpose ATM switch chip," IEEE Journal on Selected Areas in Communications, Vol. 9, No. 8, pp.1265 -1279, Oct. 1991.

최 미 진

1990년 2월 영남대학교 컴퓨터공학과 졸업. 1993년 2월 영남대학교 컴퓨터공학과 전산기시스템 전공석사 졸업. 2001년 3월 한국정보통신대학원대학교 전자상거래 과정 입학. 2003년 2월 한국정보통신대학원대학교 전자상거래 과정 졸업 1999년 10월~현재 (주)이오테크닉스 과장. 관심분야는 전자상거래, 실시간 그룹 상호작용

조 은 영

1993년 2월 상명대학교 전자계산학과 졸업. 2001년 3월 한국정보통신대학원대학교 전자상거래 과정 입학. 2003년 2월 한국정보통신대학원대학교 전자상거래 과정 졸업. 관심분야는 전자상거래, 실시간 그룹 상호작용

강 경 란

1992년 2월 서울대학교 계산통계학과 졸업. 1994년 2월 한국과학기술원 석사 1999년 2월 한국과학기술원 박사. 1999년~2000년 한국전자통신연구원 선임연구원. 2000년~2002년 (주)디지탈웨이브 책임연구원. 2002년~현재 한국정보통신대학원대학교 연구조교수. 관심분야는 multicast, real-time collaboration, mobility

이 동 만

1982년 서울대학교 컴퓨터공학 학사 1984년 한국과학기술원 전산학 석사 1987년 한국과학기술원 전산학 박사 1987년~1988년 한국과학기술원 박사후 과정. 1988년~1997년 Hewlett-Packard 책임연구원. 1997년~현재 한국정보통신대학원대학교 부교수. 관심분야는 multicast protocol, group communication, distributed virtual environment, pervasive computing