韓國IT서비스學會誌
第10卷 第3號
2011年 9月

분산 환경에서의 협업을 위한 네트워크 동기화 기법*

송정욱** · 권용무***

Network Synchronization for Collaborative Work in Distributed Environment*

Jungwook Song** · Yong-Moo Kwon***

■ Abstract ■

In the every day life of the people, the Internet is widely used. Currently over 1.9 billion people have one or more email addresses and over 600 million people use the Facebook. People are collaborating via the Internet more and more. When people are collaborating through the Internet, the differences of the message delivery delay are the biggest problem that disturbs the collaborative work over the network. To solve the differences of the message delivery delay, we introduce the delay-gap method. An experimental code have been implemented and the efficiency of the delay-gap is presented through the results from the experiment that have many participants.

Keyword : Internet Traffic, QoS, Media/Event Synchronization, Networked Collaboration

논문투고일 : 2011년 01월 21일 논문수정완료일 : 2011년 02월 22일 논문게재확정일 : 2011년 03월 15일

* 본 연구는 KIST "실감형 차세대 웹 플랫폼 개발" 과제 및 "Tangible 소셜 미디어 플랫폼 기술개발" 과제의 지원으로 수행되었습니다.

** KIST 영상미디어연구센터 Postdoc

*** KIST 영상미디어연구센터 책임연구원, 교신저자

1. 서 론

컴퓨터 네트워크의 발달로 우리의 생활은 나날이 편리해지고 있다. 인터넷 도입 초기에는 이메일과 검색 등이 서비스의 전부였지만, 지금은 전자상거래, 온라인 뱅킹, 게임 등 거의 일상생활의 전반을 인터넷에서 해결할 수 있다. 이제는 소통도 네트워크를 통하는 것이 더욱 편리할 정도이다. 심지어 어떤 연구자들은 구글(Google)[10] 검색엔진에서 자료를 수집하여 통계연구를 하는 경우도 있다.

네트워크는 공동 작업을 하기에 매우 좋은 도구다. 시차 혹은 거리상의 한계를 네트워크를 통해 쉽게 극복할 수 있다. 네트워크를 통해 심지어는 지구 반대편에 있는 동료와도 쉽게 협업이 가능하고, 멀리 떨어진 장소에 있는 연주자들이 함께 실시간으로 공연을 할 수도 있다. 하지만, 이런 경우에 사용되는 네트워크는 특별하게 특정 목적을 위해 사용되는 닫힌 네트워크이다.

인터넷과 같은 개방된 네트워크를 통해 실시간 협업 하기란 쉽지 않다. 개방된 네트워크란 누구나 쉽게 접근할 수 있고, 안정성이 보장되지 않는 네트워크를 말한다. 가장 큰 문제점은 각 사용자 사이에 이벤트 메시지를 전달하는데 걸리는 시간이 차이가 있기 때문이다.

본 논문에서는 메시지 전달 시간의 차이를 극복할 수 있는 효과적인 방법을 제안한다. 제 2장은 연구 동기, 제 3장은 네트워크 동기화를 위한 알고리즘 제안, 제 4장은 시스템 구현, 제 5장은 성능 평가에 대해 기술하고, 제 6장에서 결론 및 향후 과제를 제시한다.

2. 연구 동기

네트워크를 이용한 실시간 협업 시스템의 응용 시스템으로 네트워크를 통한 실시간 응원 서비스를 테스트베드(testbed)로 구현하였다. 구현된 시스템의 목표는 여러 사용자가 가상세계에서 구축된 가상 경기장에서 실시간으로 스트리밍 되는 야구 경기를 관람하면서 함께 응원하는 것을 도와주는 것이다. 가상 세계 서버 혹은 각 사용자에게서 여러 이벤트가 발생할 것이고, 발생된 이벤트는 다른 사용자들에게 전달이 되어야 하지만, 네트워크에서 각 사용자마다 메시지가 전달되는 시간에 차이가 있으므로 대부분의 사용자는 각기 다른 시간에 이벤트를 전달 받게 된다. 원활한 협업 응원을 하기 위해서는 이벤트가 도착하는 시각을 동기화 할 필요가 있고 이를 위해 네트워크 동기화 기법을 제안하게 되었다.

2.1 메시지 지연

현재 우리가 사용하고 있는 인터넷은 매우 안정적으로 운영되고 있다. 하지만 메시지의 전달 과정이나 서비스 품질 등을 사용자가 제어하지는 못한다. 특히, 최종 사용자(end user)는 메시지를 보내거나 메시지의 도착을 기다리는 동작 이외에는 네트워크에 대한 어떠한 제어 권한도 없다. 현재의 인터넷은 최선 노력 네트워크이기 때문에 보낸 메시지가 실제 도착하기 이전에는 지연시간을 예측하기 어렵다. 그러나 호스트 사이의 메시지 왕복 시간(round trip time) 혹은 단방향 지연시간을 측정할 수 있는 방법이 있으며[9, 13, 14], 측정된 값을 이용하여 메시지 전달 지연시간을 예측할 수 있다.

유니캐스트(unicast) 네트워크에서는 서버와 여러 클라이언트가 연결되어 있고 클라이언트들이 서버를 통해 협력 작업을 하고 있을 때, 일반적으로 하나의 클라이언트에서 발생한 이벤트가 서버를 통해서 동시에 다른 모든 클라이언트로 전달이 된다. 유니캐스트 통신 방법에서 물리적으로 전송하는 시각의 차이가 발생할 수밖에 없으므로 절대적으로 동시는 없다. 그러나 각각의 메시지가 클라이언트에 전달되는 과정에 발생하는 전송 지연의 크기에 비하면 무시할 수 있을 만큼 작은 차이이므로 동시라는 표현을 사용한다.

문제점은 송신자가 보낸 메시지가 수신자에 전

달이 되는데 걸리는 시간인 메시지 지연이다. 서버에서 메시지를 여러 클라이언트로 동시에 전송하더라도 각각의 메시지가 전달되는데 소요되는 시간이 다르므로 대부분의 메시지는 클라이언트에 각각 다른 시각에 도달하게 된다. 멀티캐스트(multi-cast) 환경에서도 마찬가지로 각각 멀티캐스트 클라이언트까지 도달하는 트리의 깊이가 다를 수 있으므로 메시지 지연 시간이 달라질 수 있다.

2.2 이벤트 동기화(Event Synchronization)

이벤트 동기화란 서버에서 전송된 이벤트 메시지의 프레젠테이션 시각 동기화를 의미한다. 즉, 서버에서 전송된 이벤트 메시지가 클라이언트로 전송되고 클라이언트 응용 계층에서 표시되는 시각을 프레젠테이션 시각이라 할 때, 여러 클라이언트에서 동일한 이벤트에 대한 프레젠테이션 시각을 동기화 하는 것을 이벤트 동기화로 정의한다. 하지만 메시지 지연 시간에 비해 이벤트 메시지가 네트워크를 통해 클라이언트에 도착한 뒤 응용계층까지의 지연 시간은 상대적으로 매우 짧은 시간이므로 이벤트 메시지의 도착 시각을 프레젠테이션 시각으로 정의해도 문제가 되지 않는다.

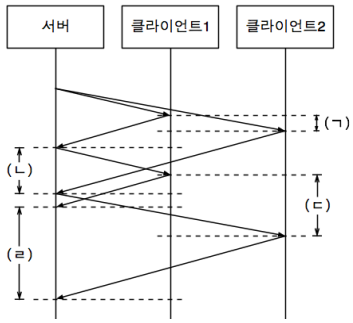
하나의 이벤트에 대해 사용자의 반응은 프레젠테이션 시각에 의존하기 때문에 여러 사용자 사이의 프레젠테이션 시각 동기화는 매우 중요하다. 즉, 서버에서 발생한 이벤트가 각 클라이언트로 전달되는 시각이 동기화 되어야만 사용자의 반응도 동기화 될 수 있다.

만약 프레젠테이션 시각이 동기화 되지 않으면 사용자의 반응 시각이 달라질 것이고, 여러 사용자의 반응 메시지가 전달되는 시간이 각각 다르므로 상대적으로 큰 메시지 지연 값을 갖고 있는 사용자는 서버와의 상호작용에서 계속해서 늦어지는 음의 되돌림(minus feedback) 현상이 발생한다. 또한, 이전에 설명한 메시지 지연 때문에 이벤트 동기화는 달성하기가 매우 어렵다.

2.3 반응 동기화(Response Synchronization)

반응 동기화란 사용자가 전송한 반응 메시지가 서버로 도착하는 시각을 동기화 하는 것을 의미한다. 네트워크를 통한 실시간 협업 응용에서는 이러한 반응 동기화가 반드시 필요할 것이다. 하지만 이벤트 동기화와 마찬가지로 메시지 지연 때문에 반응 동기화 또한 달성하기가 매우 어렵다.

각 메시지의 전송 시간의 차이 때문에 이벤트의 프레젠테이션 시각이 달라지고, 프레젠테이션 시각의 차이로 인해 각 사용자의 반응 시각이 달라진다. 따라서 사용자 오류(human error)를 무시하면 가장 큰 문제는 서로 다른 메시지의 전송 지연이다. [그림 1]은 서버와 클라이언트 사이에서 메시지 전달 지연과 음의 되돌림 현상을 보여준다. 서버에서 클라이언트 1과 클라이언트 2로 메시지를 동시에 전송 했지만, 다른 전송 지연 값으로 인해 각각 다른 시각에 도착한다. 각 클라이언트는 메시지가 도착하자마자 응답을 보냈음에도 응답 메시지가 서버에 도착하는 시각은 더 크게 차이가 생긴다. 이러한 메시지의 도착 시각의 차이는 (ㄱ), (ㄴ), (ㄷ), (ㄹ)로 메시지를 교환할수록 누적되어 점점 커진다.



[그림 1] 메시지 지연과 음의 되돌림

3. 관련 연구

메시지 동기화를 위해서 로컬 및 분산 시스템간의 클럭 동기화 기법들이 요구된다. 클럭동기화 기법으로는 타임 서버를 사용하는 기법과, 일반적인

글로벌 인터넷 환경에서의 클럭동기화 기법이 있다[11]. 타임 서버를 기반으로 한 기법으로는 Cristian 알고리듬[1], Berkeley 알고리듬[2]이 발표되어 있다. 인터넷 환경에서의 클럭동기화 기법으로는 NTP(Network Time Protocol)[12]가 있으며, 인터넷 클럭동기화 표준기법으로서 2010년 4월 NTP v4(RFC 5905)[8]가 발표되었다. NTPv4의 경우 동기화 정확도가 퍼블릭 인터넷에서 10ms 수준, LAN 환경에서 200us 수준을 제공한다.

한편 서브 ms 단위의 고정밀 동기화를 위해 LAN 환경을 대상으로 하여 PTP(Precision Time Protocol)가 2002년 PTP 버전1 IEEE-1588-2002, 2008년 PTP 버전2 IEEE 1588-2008로 발표되었다[7].

메시지 동기화 방법으로는 시스템간의 클럭동기화 후에 각 시스템에서 보내는 메시지에 전송시각에 대한 타임스탬프(Time Stamp)를 함께 전송함에 의해 동기화를 수행하는 방법이 적용되고 있다. 게임 및 분산가상환경에서의 메시지 동기화를 위한 기법으로서 메시지 처리 대기 시간(waiting time) 결정 기법으로서 네트워크 지연 최대 시간을 적용하는 방법[5], 네트워크 상태에 따라 동적으로 대기 시간을 결정하는 방법[3], 메시지 발생 확률 기반 대기시간 결정 방법[6], 지연시간 순위에 따라 결정하는 방법[6] 들이 제안되었다. 이들 기법은 기본적으로 메시지 동기화를 위해 타임스탬프를 메시지 전송시 함께 전송하여 시스템 클럭 기준 시각으로 사용하고 있다.

4. 메시지 동기화 기법 알고리듬 제안

본 논문에서 제안하는 메시지 동기화 기법의 주요 특징은 다음과 같다.

(1) 네트워크 지연 측정시, 기존 관련 기법들에서 사용하는 RTT에 기반으로 하나, 동기화 방법 측면에서는 타임스탬프를 사용하지 않고 네트워크 이벤트 서버와 각각의 클라이언트간의 RTT 정보만을 기반으로 메시지를 동기화 하는 기법을 제안한다.

(2) 메시지 지연 문제를 효과적으로 해결하기 위해 지연-간격(delay-gap) 방법을 도입하였다. 즉 메시지의 프레젠테이션 시각을 동기화하기 위해 서버에서 여러 클라이언트로 메시지를 전송할 때 서버와 클라이언트 네트워크 지연시간을 반영하여 임의의 간격으로 전송한다.

(3) 서버에서 클라이언트로 전송하는 메시지의 지연 시간 관련하여 시스템 응답 성능을 고려하여 지연 시간 임계치(DT : Delay Threshold)를 사용한다.

4.1 메시지 지연 측정

메시지 동기화를 위해 먼저 메시지 지연을 측정할 필요가 있다. 두 호스트 (ㄱ), (ㄴ)이 서로 메시지를 주고받는다면 (ㄱ)에서 (ㄴ) 방향, (ㄴ)에서 (ㄱ) 방향의 두 메시지 지연 값이 있다. 두 메시지 지연 값이 같을 수 있지만, 네트워크 연결 특성에 따라 일반적으로 두 값은 다를 수 있다. 한쪽 방향의 메시지 지연을 단방향 지연이라 한다. 본 논문에서는 메시지 왕복 시간을 둘로 나누어서 메시지 지연 값으로 사용하기로 한다. 측정된 메시지 지연은 지연-간격을 계산하기 위해 사용되며 주기적으로 측정하여 통계 값을 이용한다.

4.2 지연-간격 계산

[그림 2]는 각 클라이언트의 지연-간격을 구하는 알고리듬이다. 지연-간격 MD(Message Delay)를 계산하기 위해서 클라이언트 목록은 메시지 지연 값의 역순으로 미리 정렬되어 있어야 한다. (i)번째 지연-간격 DG (i)는 (i)번째 항목의 메시지 지연 값에서 (i+1)번째 메시지 지연 값을 뺌으로써 구할 수 있다.

서버는 클라이언트 목록에 있는 순서대로 각 클라이언트에 메시지를 전송하면서 각 클라이언트 사이에 지연-간격에 기반한 강제적인 지연 시간 DF(Delay Force)를 준다. 따라서 (n)번째 클라이

언트의 지연-간격의 총 합은 $\sum_{i=1}^{n} DG(i)$이다. 클라이언트의 수가 너무 많을 때, 지연-간격의 총 합이 너무 큰 경우가 발생할 수 있다. 이런 경우에 지연-간격의 총 합의 임계값 DT를 정해 줄 수 있으며, [그림 2]의 알고리듬에서 DT를 적용하여 지연-간격을 조정하고 있다.

```
SORT DESC client list BY MD

sum_DG = 0
FOR  i = 1 TO n-1
       DG(i) = MD(i)-MD(i+1)
       sum_DG = sum_DG+DG(i)
END

i =1
WHILE (sum_DG > DT) AND (i < n)
       sum_DG = sum_DG-DG(i)
       DG(i) = 0
       i = i+1
END
```

[그림 2] 지연-간격 계산 알고리듬

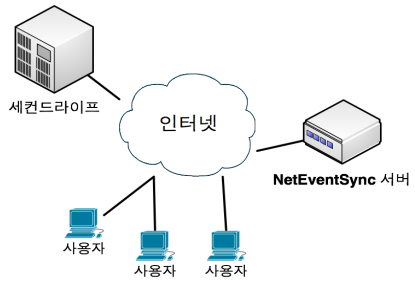
5. 시스템 구현

지연-간격 방법 메시지 동기화 기법을 시험하기 위해 가상 세계 상에서 네트워크를 이용한 실시간 협업 시스템에 적용해 보았다. 전체 실험은 가상 세계에서 여러 사람이 공동의 목표를 가지고 협업을 할 때, 제공하는 여러 설정에 따른 공동 경험(co-experience)의 느낌을 평가하기위한 실험이었으며, 공동 경험의 느낌을 더욱 배가하기 위해 지연-간격 방법의 이벤트 메시지 동기화 기법을 적용 하였다. 전체 실험은 “네트워크 상의 협업 그룹 응원 기술”[4]을 평가하기 위해 수행 되었다.

실험을 위한 가상세계 플랫폼으로 세컨드라이프(SecondLife)를 사용 하였으며, 공동 경험의 정도를 평가하기 위해 가상 야구 경기장, 야구팀의 유니폼, 응원도구 등 세컨드라이프의 여러 장치를 이용하였다.

5.1 NetEventSync 서버

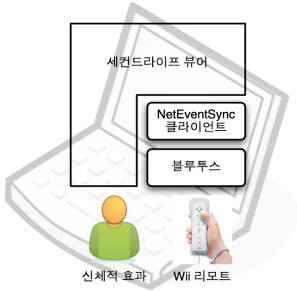
[그림 3]은 전체적인 실험 구성을 보여준다. 세컨드라이프 서비스는 기업에서 상업적인 목적으로 운영되고 있으므로 어떠한 변경도 적용할 수 없기 때문에 사용자 접속 응용프로그램인 세컨드라이프 뷰어를 수정하였다. NetEventSync 서버는 주기적으로 사용자들로부터 메시지 지연 값을 수집하고 지연-간격을 계산하고, 계산된 지연-간격을 다시 사용자에게 제공한다.



[그림 3] 전체 실험 구성

5.2 NetEventSync 클라이언트

[그림 4]는 사용자측 모듈 구성을 보여준다. 세컨드라이프 뷰어에 실험을 위해 몇 가지 기능을 추가 하였다. 신체적 효과나 위 리모트는 공동 경험 감과 인터페이스 향상을 위해 도입되었다.



[그림 4] 사용자측 모듈 구성

NetEventSync 클라이언트는 주기적으로 세컨드라이프 서버까지 메시지 지연을 측정하여 그 값을

NetEventSync 서버로 전송하고 계산된 지연-간격 값을 전달 받는다. 전달 받은 지연-간격 값은 세컨드라이프 뷰어에서 특정 이벤트를 서버로 전송할 때 적용된다.

6. 성능 평가 및 검토

실험은 한국의 프로야구 경기를 생방송으로 시청하면서 같은 팀을 응원하는 설정으로 수행 되었다. 총 40여 명의 실험 참가자가 각자의 집에서 네트워크를 통해 참여했다. 각 참가자는 실험을 위해 준비된 세컨드라이프 가상세계의 비공개 영역에 마련된 가상의 경기장에 모여서 실험에 참가하였다. [그림 5]는 실험이 이루어진 가상공간의 구성을 보여준다.

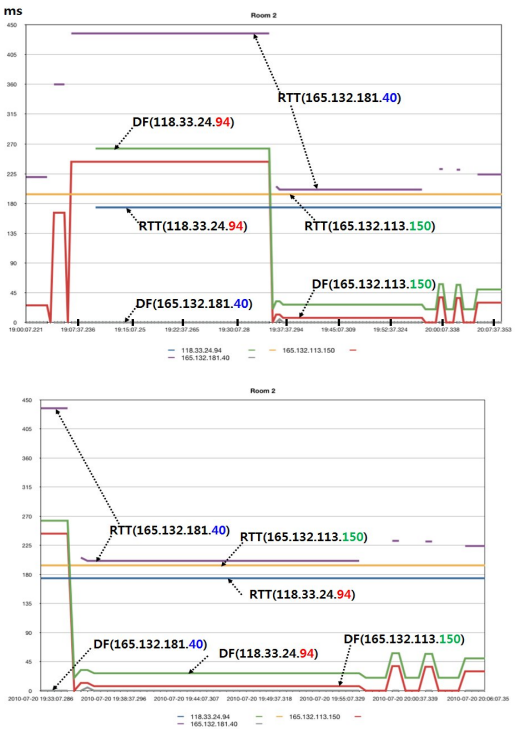
[그림 5] 가상 경기장에서 응원하는 참여자들

[그림 6] 실험 참가자 분포

[그림 6]은 실험 참가자들의 분포를 보여준다. 참가자들은 서울 경기지역에서 주로 참여하였고 대전에서 참여한 참가자도 있었다.

6.1 강제 지연 시간

[그림 7]은 실험 결과의 일부를 보여준다. 시뮬레이션 실험과는 다르게 일부 자료가 누락이 되었다. 이는 NetEventSync 서버와 클라이언트 사이에 통신 프로토콜이 부하 경감을 위해 UDP로 구현되었기 때문인 것으로 보인다. [그림 7]은 세 개의 IP 주소 클라이언트 간의 RTT 차이에 따라, 가장 RTT가 큰 IP 주소 165.132.181. 40 클라이언트 대비 상대적인 강제 지연 시간 DF를 나타낸 것이다. 세컨드라이프 서버측은 어떠한 변경도 할 수 없었기 때문에 실제 측정하지는 못하였지만, 강제 지연 시간을 조정하여 이벤트 메시지를 전송했기 때문에 도착시각은 거의 비슷했을 것이다.



[그림 7] 메시지 지연을 따라 변화하는 강제 지연 시간 DF(Delay Force)

6.2 설문 조사

사용자의 공동 경험 느낌을 평가하기 위해 설문

조사를 도입하였다. 설문 문항 중에서 메시지 동기화와 관련된 문항은 다음과 같다.

- 순차성 : 파도타기 응원을 할 때, 실제 경기장에서 파도타기 응원을 하는 것과 같이 아바타들이 순차적으로 응원한다는 느낌을 받았다.
- 동시성 : 단체 점프 응원 동작 시에 다른 사용자들과 자신의 동작이 동시에 이루어 졌는가?
- 지연성 : 내가 위모트로 응원 버튼을 누르거나 응원 모션을 취해서 응원했을 때, 나의 아바타들은 딜레이(지연)없이 반응했다.

응답의 형태는 5점식 이었으며, 5는 가장 좋은 점수를 의미하고 1은 가장 나쁜 점수를 의미한다. 본 실험에서는 가상 경기장에서의 참여자들간의 연대감(solidarity)과 사용할 수 있는 인터랙션의 방법에 따라 4개의 가상 경기장을 만들고 참여자들을 4개의 그룹으로 나누어 각각의 경기장에서 응원하도록 하였다. 각 그룹은 7~10명의 사용자들로 구성되었으며 응답 결과는 그룹별로 통계처리 하고, 전체 그룹의 평균 통계자료를 제시하였다. 같은 설문을 5회 말 이후와 경기 종료 후 두 번에 걸쳐 응답 받았다.

〈표 1〉 설문 결과(1차)

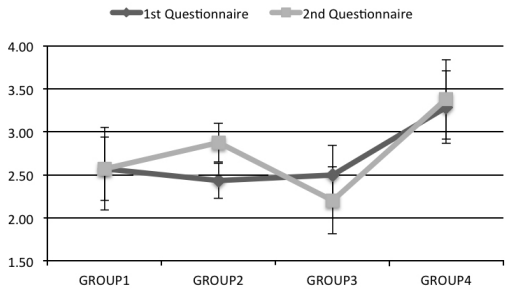
문항	그룹 1	그룹 2	그룹 3	그룹 4	평균
순차성	2.57	2.43	2.50	3.29	2.69
동시성	3.57	2.86	2.40	3.14	2.99
지연성	2.71	3.43	3.30	3.43	3.21

〈표 2〉 설문 결과(2차)

문항	그룹 1	그룹 2	그룹 3	그룹 4	평균
순차성	2.57	2.88	2.20	3.38	2.75
동시성	3.14	3.13	2.20	3.50	2.99
지연성	3.00	4.00	3.70	3.75	3.61

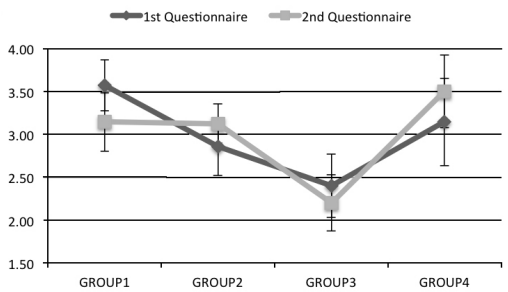
<표 1>과 <표 2>는 그룹별 설문 결과 및 전체 그룹의 평균 설문 결과를 보여준다. 두 결과는 “지연성” 문항을 제외하고 거의 같은 결과를 나타낸다. “지연성” 문항에서 차이는 실험 참가자들이 세컨드라이프 인터페이스에 익숙하지 못하고, 실험이 너무 오랜 시간 진행되었기 때문으로 생각된다.

[그림 8]은 “순차성” 문항 설문 결과의 변화를 보여주며, 큰 변화는 없다. 실험 관리자가 순차적 응원 동작을 이끌었지만 세컨드라이프 서버에서 참가자들에게 전달되는 이벤트 메시지 지연을 제어하지 못했다. 따라서 각 참가자의 프레젠테이션 시각이 달랐을 것이다. 결과적으로 각 사용자는 적절한 시각에 반응을 하지 못했을 것이다.



[그림 8] 순차성 문항 결과

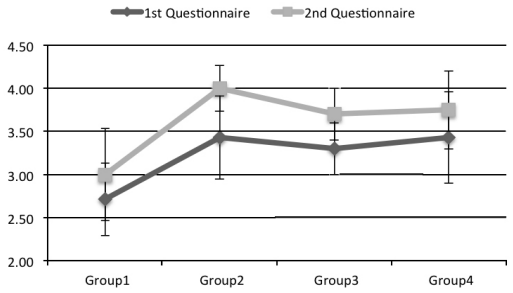
[그림 9]는 “동시성” 문항 설문 결과의 변화를 보여준다. 그룹 3을 제외하고 대부분의 참가자들은 응원 동작이 비교적 동시에 잘 이루어 졌다고 응답했다. 동시성 응원 동작 보다 순차성 응원 동작이 더 어려웠던 것으로 여겨진다.



[그림 9] 동시성 문항 결과

[그림 10]은 “지연성” 문항 설문 결과의 변화를 보여준다. 높은 점수는 참가자가 지연을 느끼지 못

했음을 의미한다. 2차에서 지연을 더 느끼지 못한 것은 실험을 하면서 점점 세컨드라이프에 익숙해 짐으로 나온 변화로 보인다. 결과는 측정된 평균 메시지 지연으로부터 계산된 지연-간격을 강제로 적용 했음에도 불구하고 실험 참가자들은 지연을 거의 느끼지 못했음을 알려준다.



[그림 10] 지연성 문항 결과

6.3 검토

본 논문에서 제안한 방법을 적용한 경우와 적용하지 않은 경우를 비교를 통한 본 논문에서 제안한 기법의 유효성 검토가 추가로 이루어지면 좋을 것으로 판단되나, 이에 대한 실험은 이루어지지 않아 본 논문에 제시 하지 못한 한계점을 가지고 있다.

7. 결론 및 향후 과제

서버와 클라이언트 혹은 사용자들 사이에 네트워크를 통해 지속적으로 상호 작용을 해야 하는 응용의 경우에 메시지 동기화는 서비스 품질을 결정하는 매우 중요한 요소가 될 것이다. 하지만 인터넷 자체의 불안정성 때문에 같은 출발지에서 다른 목적지로 가는 메시지를 같은 시각에 도착하도록 할 수 있는 방법은 없었다.

본 논문에서 여러 클라이언트로 동시에 보낸 메시지가 가능한 같은 시각에 도달할 수 있도록 하는 지연-간격 메시지 동기화 방법을 제안 하였다. 제안된 방법을 검증하기 위해 시뮬레이션 실험을 수행 하였으며, 또 실질적인 환경에서 효과를 검증하기 위한 실험도 수행하였다. 실험 결과에서 보여 지듯이 정밀한 하드웨어의 도움 없이 지연-간격 방법은 잘 작동할 수 있는 것으로 보인다. 또한 강제적으로 적용된 지연-간격을 실질적인 사용자는 잘 느끼지 못하는 것으로 보인다.

네트워크의 상태를 측정할 때 메시지 왕복 시간 이외에도 네트워크 지터와 같은 고려될 수 있는 여러 가지 인수가 있다. 지터나 네트워크 서비스 품질 지원 메커니즘이 있다면 메시지 동기화에 도움이 될 것이다. 메시지 동기화 방법에 대한 연구의 다음 단계로 실시간 동영상 스트리밍에서 메시지 왕복 시간과 네트워크 지터를 같이 적용하는 것을 고려하고 있다.

참 고 문 헌

[1] Cristian, F., “Probabilistic clock synchronization”, *Distributed Computing*, Vol.3, No.3 (1989), pp.146-158.

[2] Gusella, R. and S. Zatti, “The accuracy of the clock synchronization achieved by TEMPO in Berkeley UNIX 4.3BSD”, *IEEE Transactions on Software Engineering*, Vol.15, No.7(1989).

[3] Hong, E., D. Lee, E. Park, and K. Kang, “An efficient synchronization mechanism transmission adapting to heterogeneous delay in networked virtual environments”, *Proc. of SPIE/ACM MMCN*, (2003), pp.24-33.

[4] Lee, C., H. Choi, and Y.-M. Kwon, “Networked Collaborative Group Cheerleading Technology”, *Proc. International Symposium on Ubiquitous VR,* Gwangju, Korea, (2010), pp. 20-23.

[5] Lin, Y., K. Gou, and S. Pau, “Sync-MS : Synchronized message service for real-time multi-player distributed games”, *Proc. of the*

10th IEEE Int'l Conf. on Network Protocols, (2002), pp.155-164.

[6] Paik, D., C.-H. Yun, and J. Hwang, "Effective message synchronization methods for multiplayer online games with map", *Computers in Human Behavior*, Vol.24(2008), pp. 2477-2485.

[7] Precision Time Protocol, http://en.wikipedia.org/wiki/Precision_Time_Protocol.

[8] RFC5905, "Network Time Protocol Version 4 : Protocol and Algorithms Specification", *IETF RFC*, http://tools.ietf.org/html/rfc5905.

[9] "RFC792, INTERNET CONTROL MESSAGE PROTOCOL", *IETF RFC*, http://tools.ietf.org/html/rfc792.

[10] Search Engine, Google, http://www.google.com.

[11] WIKIPEDIA, "ClockSynchronization", http://en.wikipedia.org/wiki/Clock_synchronization.

[12] WIKIPEDIA, Network Time Protocol, http://en.wikipedia.org/wiki/Network_Time_Protocol.

[13] WIKIPEDIA, "Round-trip delay time", http://en.wikipedia.org/wiki/Round-trip_delay_time.

[14] WIKIPEDIA, "Ping", http://en.wikipedia.org/wiki/Ping.

◈ 저 자 소 개 ◈

송 정 욱 (swoogi@areum.kr)

건국대학교 컴퓨터공학과에서 이동 네트워크에서의 경로 최적화 분야로 박사를 받았다. 2010년 3월부터 2011년 2월까지 KIST 영상미디어연구센터에서 Postdoctor를 하였다. 주요 연구관심 분야는 네트워크 인터랙션, 시스템 네트워크, 모바일컴퓨팅, 미래인터넷, 클라우드컴퓨팅 등이다.

권 용 무 (ymk@kist.re.kr)

한양대학교 전자공학과에서 학사, 석사를 하고, 동 대학에서 멀티미디어 DB 분야로 박사를 받았다. 현재 KIST 영상미디어연구센터에 재직 중이며 주요 연구관심 분야로 가상현실/가상세계, 3차원 영상, HCI, 네트워크 협업, 웹 응용 서비스 등이며, IEEE TASE, IEEE TCE, Optical Engineering, 한국정보과학회, 대한전자공학회, 한국문화콘텐츠기술학회 등에 다수 논문을 실었다.