KIPS Tr. Comp. and Comm. Sys.
Vol.5, No.9 pp.223~228 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2016.5.9.223

Development Migration Agent Server for Seamless Virtual Environment

Donghyun Won† · Dongun An†† · Seungjong Chung†††

ABSTRACT

Nowadays users of Virtual Environment are want to play with thousands of players in an evolving virtual world at the same time over the internet. So, the load of this kind of Virtual Environments is heavier than that of any other precedents. One of load balancing methods is map-partition to divide the load of entire system which is vulnerable to delay message between clients and servers. In this paper, we propose a Migration Agent application server architecture using to help migration of player character between field servers and to reduce response time between clients and field servers. Migration Agent is reduce Player Character's responds time as Cache Server, if Player Character move to another Field Server, Player Character need the synchronization process in the DBMS approach, to minimize response time by reducing the period for cross - Player Character Field Server to perform the role. Field Server by placing them in form of a stack existing form of grid, for load concentrated on a specific server.

Keywords : Migration Agent, Field Server, Player Character, Virtual Environments

Seamless 가상 환경을 위한 Migration Agent 서버 개발

원 동 현† · 안 동 언†† · 정 성 종†††

요 약

최근 가상 환경의 발전은 이용자들이 현실 세계를 그대로 반영한 가상 환경을 요구하도록 변화하고 있다. 하지만 현재 서버 및 네트워크 환경은 소수의 이용자가 소수의 그룹을 이루어 함께하는 형태로, 현실 세계를 그대로 반영하지 못하고 있다. 가장 큰 이유 중 하나는 이용자간 정보 동기화에 많은 부하가 발생하기 때문이며, 이러한 부하는 정보 전달 시간을 증가시키게 된다. 본 논문에서는 이러한 문제를 최소화하기 위한 Migration Agent 서버 시스템을 제안한다. Migration Agent 서버는 Field Server 이동시 Player Character의 동기화 과정에서 DBMS의 접근을 최소화하여 응답 시간을 줄여주기 위한 서버로 Field Server간 Player Character 이동을 위한 캐시 서버 역할을 수행한다. 그에 더해 스택형태로 Field Server를 배치함으로 기존 격자 형태로 인해 특정 서버에 부하가 집중되는 것을 보완하였다.

키워드 : Migration Agent, Field Server, Player Character, 가상 환경

1. 서 론

최근 가상 환경에 대한 발전은 기존 MMORPG와 같은 대규모 이용자(PC: Player Character)가 이용하는 게임 환경을 넘어, 현실 세계를 그대로 반영한 증강 현실에 대한 요구로 이어 지고 있다[1]. 통신 기술과 IoT의 발전에 힘입어 현실 세계를 그대로 가상으로 옮겨 오는 것도 가능한 일이 되고 있다[2]. 가상 환경은 한정된 게임 공간 개념이 아닌 지속적으로 확장 가능해야 하며 많은 PC가 동시에 이용 가능해야 한다. 이때 개개의 Field Server가 관리하는 위치와는 관계없이 사용자는 개개의 Field Server들로 구성된 가상 환경을 하나의 지역으로 인식하게 하고 분산된 Field Server들이 구성한 가상 환경을 자유롭게 이동할 수 있어야 한다. 이 방식은 PC에게 넓은 환경을 제공한다는 장점과 함께, Field Server들 간의 영역들이 서로 인접하고 있는 경계 부분에서의 처리가 매우 복잡하다는 단점을 갖는다[3]. 이러한 문제를 해결하기 위해서 [4, 5]에서는 특정 서버에 사용자가 집중되는 경우를 위한 부하 분산 시스템을 제안한다. [6]에서는 부하 분산을 위해 Cloud Server 환경을 제안하고 있으며 이러한 방식은 쉽게 부하를 분산할 수 있다는 장점 및 뛰어난 확장성으로 각광 받고 있다. 하지만 Cloud Server 환경 역시 가상 환경 구성에 따라 응답 시간은 달라질 수 있으며 DBMS에 대한 접속 증가로 인한 성능 저하가

† 준 회 원 : 전북대학교 대학원 컴퓨터공학과 박사수료
†† 종신회원 : 전북대학교 대학원 컴퓨터공학과 교수
††† 정 회 원 : 전북대학교 대학원 컴퓨터공학과 명예교수
Manuscript Received : August 9, 2016
Accepted : August 29, 2016
* Corresponding Author : Dongun An(duan@chonbuk..ac.kr)

존재한다. 본 논문은 [6]과 유사한 가상 서버 환경에, 기존 격자형 Field Server 배치 대신 스택구조로 Field Server를 배치하고 DBMS의 접근을 최소화하기 위한 캐시 역할을 수행하는 Migration Agent 서버를 적용함으로 응답 시간을 최소화하였다.

2. 관련 연구

2.1 AOI

가상 환경 연구에서는 일반적으로 참여자 별 인식 영역을 AOI(Area Of Interest, 관심영역)[7]라고 하며, 게임 내에서는 PC 주변의 일정한 범위를 의미한다. AOI는 게임 진행 시 PC에게 흥미를 유발시킬 수 있는 지역으로, 다음과 같은 사건들이 플레이어 주변에서 발생한다.

- PC/NPC(Non Playable Character)의 상태 변화
- Object의 상태 변화
- 이벤트 지역의 상태변화
- 채팅(일반채팅)

PC의 AOI내의 위와 같은 사건이 발생하면 PC는 서버에게 발생한 사건에 대한 정보와 자신의 정보를 보내고 서버는 이러한 정보를 처리하여 요청한 플레이어와 주변의 PC들에게 갱신 정보를 보낸다.

MMORPG에서는 AOI를 적용함으로서 MMORPG에서의 네트워크 트래픽과 서버의 연산을 줄여 서버와 네트워크의 효율을 높일 수 있다.

2.2 Migration Server

Field Server들 간의 캐릭터의 이동을 전담하여 처리하는 서버이다. 기존에는 Field Server의 캐릭터의 이동 및 동기화 처리를 메인 서버에서 수행하였으나 이러한 Field Server들 간의 동기화 처리 부분을 별도의 캐릭터의 Field Server 간 이동 처리만을 담당하는 서버를 둔다. 이것을 Migration Server라고 한다[8].

이 서버는 Seamless게임 환경에 있어서의 네트워크 통신의 증대에 따른 지연 문제를 개선하기 위하여 별도의 Field Server 간 캐릭터의 이동을 전담하는 서버를 구성한다.

이 서버는 해당 영역에서 발생하는 캐릭터들의 메시지를 처리하고 캐릭터들이 다른 Field Server로 자유롭게 이동할 수 있도록 Field Server 간 데이터 전송 및 이동에 대한 동기화 처리를 수행한다.

Migration Server는 모든 Field Server에 대한 정보를 가지고 있다. 그래서 한 Field Server에서 다른 Field Server로 이동할 때 Migration Server는 이동하려는 캐릭터의 정보를 등록하고 실제 이동이 발생할 시에 이에 대한 정보를 이동하려는 Field Server에 등록하여 준다.

3. Migration Agent 서버 개발

3.1 개발환경

테스트를 위한 가상 서버는 인텔 제온 E3-1280V (8M캐시, 3.7GHz) 16GB RAM으로 구성된 워크스테이션에 Ubuntu 16.04 LTS 버전을 설치하고 가상서버로 KVM을 적용하였다. 서버 구성은 Table 1과 같다.

각 Field Server는 PC의 정보를 저장하기 위한 DBMS가 설치되어 있다. 사용된 DBMS 시스템은 MySql Community Server 5.7 버전이 설치되어 있으며 PC가 로그인/아웃 및 Field Server 이전시 PC 정보를 저장하도록 설정하였다.

Table 1. Server composition

서버	OS	Core	Ram	SSD
1	Ubuntu 16.04	2	2G	8G
2	Ubuntu 16.04	2	2G	8G
3	Ubuntu 16.04	2	2G	8G
4	Ubuntu 16.04	2	2G	8G
5	Ubuntu 16.04	2	2G	8G

3.2 Field Server

[6]은 Fig. 1과 같은 격자형태의 서버 환경을 구성한다. 격자형태의 Field Server 배치는 Field Server 인접 지역을 넓게 한다. 인접 지역이 넓을수록 더 많은 데이터 동기화가 필요하고 이는 응답 시간 저하로 이어진다.

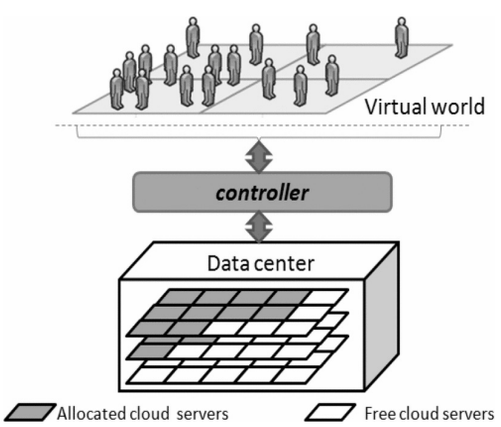


Fig. 1. Grid base Virtual world

만약 9개의 Field Server로 이루어진 격자 형태의 가상 환경이라면 정 중앙에 위치한 Field Server는 다른 Field Server들에 비해 2배나 많은 인접 지역을 가지게 되고, 중앙에 위치한 PC들의 응답 시간은 다른 Field Server에 위치한 이용자보다 느려질 수 밖에 없다.

Fig. 3과 Fig. 4는 격자형 배치와 스택형 배치의 Field Server 부하를 검증하기 위하여 총 16개(가로4*세로4)의 Field Server를 설정하고 AOI 범위를 달리하여 PC에게 정보를 제공하기 위해 발생한 Field Server 부하를 나타낸다.

가상 환경은 가로 세로 10Km로 설정했으며 한 Cell당

50cm로 설정하였다. PC는 20,000개를 고르게 배치하였으며 이 때 서버에 발생하는 부하를 측정하였다. AOI 30cell로 설정하였다.

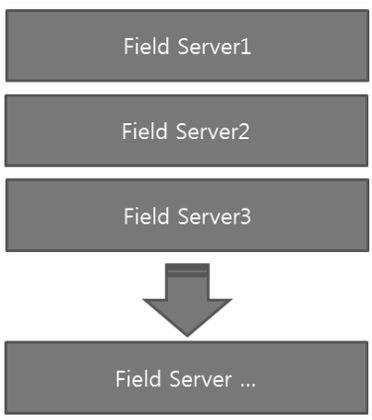


Fig. 2. Stack base Field Server composition

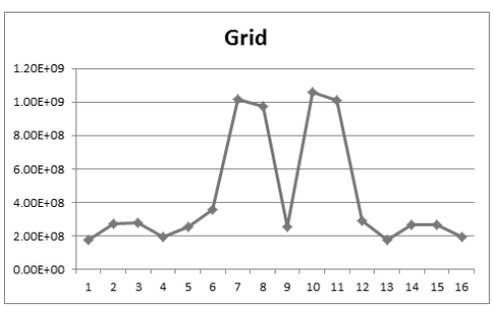


Fig. 3. Grid base Field Server Load

Fig. 3은 격자 방식이 인접 지역 (AC)에 위치한 PC 를 동기화하기 위해 특정 서버에 부하가 집중된 것을 보여준다. Fig. 4는 스택형 구조가 그리드 구조에 비해 비교적 균등한 부하 분산을 나타내고 있음을 보여준다.

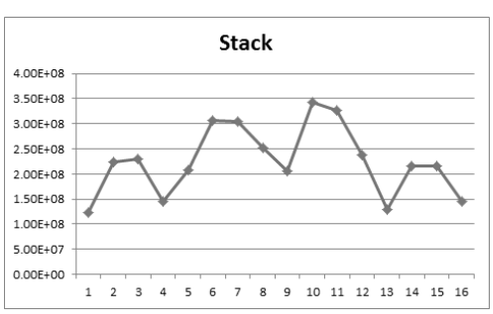


Fig. 4. Stack base Field Server Load

성능 비교를 위해 J. Lui, M. Chan [11]의 Equation (1)을 이용하였다.

$$C_p = W_1 C_P^W + W_2 C_P^L \quad (1)$$

스택형 구조의 경우 맨 윗단과 아랫단의 서버에 비해 중앙에 위치한 Field Server들에 상대적으로 많은 부하를 발생시키는 것을 볼 수 있으나 그리드 구조에 비해 그 차이가 적었다.

Table 2는 격자 형태와 스택 형태의 배치에 따른 Cp [11] 차이를 보여준다.

Table 2. Grid and Stack Qulity Function Cp

구분	평균	표준 편차	분산
Grid	4.40E+08	3.46E+08	1.19E+17
Stack	2.26E+08	6.9E+07	4.8E+15

본 논문에서는 특정 Field Server에 부하가 집중되는 문제를 해결하고자 Fig. 2와 같은 스택형 가상 세계 배치를 구성하였다.

3.3 Migration Agent 서버

Field Server 프로그램은 자바로 작성되었으며 멀티쓰레드 방식으로 구성되어 동시에 PC 100개의 접속을 처리하도록 구성하였다. 본 논문에서 구성한 Field Server 성능으로는 100개 이상의 PC 접속을 동시에 처리하게 되었을 때 PC대기 시간이 평균 30초 이상 증가하여 실험이 불가능하였다. 여러 테스트를 거쳐 최적 값으로 PC 100개 동시 접속을 처리하도록 하였다.

가상 환경에 접속하는 PC는 외부 네트워크에서 접속하도록 설정했으며 PC는 동시 최고 1,000개가 각 Field Server에 접속하도록 하였다.

Migration Agent 서버는 Field Server 간 PC의 이동시 가장 문제가 되는 DB 동기화를 위한 서버이다. 각각의 Field Server는 PC의 상태에 따라 정보를 저장하고 PC의 정보를 전달하기 전에 PC의 정보를 DB에 저장한다. 이 과정은 상대적으로 많은 시간이 소요되기 때문에 일반적인 게임 서버에서는 로그인/아웃이나 특별한 경우를 제외하고는 DBMS에 대한 접근을 최소화 한다. Migration Agent 서버는 소켓 통신을 이용해서 PC의 AOI 내 인접 Field Server의 PC 정보를 DBMS 접속 없이 전달할 수 있는 캐시 서버 역할을 수행한다.

Migration Agent 서버는 Field Server메모리에 있는 PC 정보를 소켓통신을 통해 전달 받고 PC가 AOI 범위 내 인접한 Field Server의 인접 지역으로 이동시 PC 정보를 Migration Agent 서버로 전송한다. Field Server는 PC가 다른 Field Server 내의 PC와 이벤트 발생 시 인접 Field Server로부터 정보를 요청하는 대신 Migration Agent서버에게 정보를 요청한다.

Migration Agent를 이용하면 Field Server는 인접 서버에 정보 전달을 위해 DBMS에 접근하지 않아도 되기 때문에 PC에 응답 시간을 줄일 수 있다.

4. 실험 및 결과

4.1 가상 환경 구성

실험을 위한 가상 환경은 가로 세로 10Km 중소형 도시 크기를 기준으로 하였으며 Cell당 50Cm를 할당하였다. 하나의 Cell에 PC 수는 1로 제한하였으며 다른 PC가 존재하면 이동할 수 없도록 하였다. AOI는 사람의 음성이 0.1초 이내 도달 가능한 거리인 34.3m를 기준으로, PC를 중심으로 상하좌우 70 Cell을 할당하였다. PC의 배치는 가상 환경에 고르게 분포하도록 하였으며 PC의 수는 2015년 기준 국내에서 가장 인구밀도가 높은 경기도에 해당하는 12,398명[9]을 기준으로 하여 약 0.1배에서 10배에 해당하는 수를 배치하였다. 실험에 적용된 PC 값은 1,000, 10,000, 100,000이다.

Fig. 5와 Fig. 6은 Migration Agent 시스템 적용 여부에 따른 가상 세계 구성을 보여준다.

Fig. 5와 Fig. 6 모두 5대의 서버를 이용하며 Fig. 6의 경우 4대가 가상 서버를 구성하며 한 대는 Migration Agent 시스템으로 사용하였다. Migration Agent 시스템이 담당하는 구역은 가로 20,000Cell, 세로 140Cell * 3 이다.

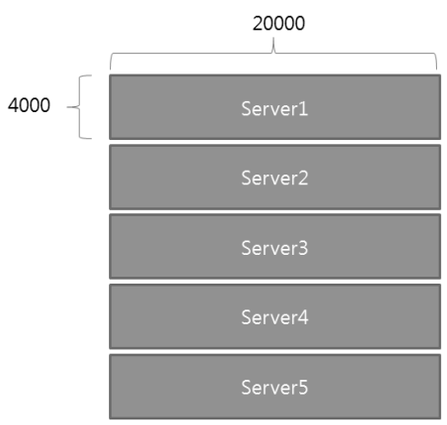


Fig. 5. Virtual World without Migration Agent

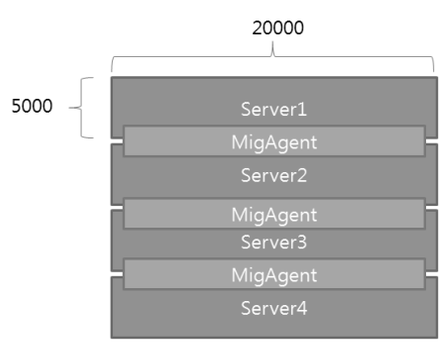


Fig. 6. Virtual World with Migration Agent

4.2 실험 결과

Table 3과 Table 4는 Migration Agent 적용 여부에 따른 평균 응답시간을 보여준다. 예상과 같이 Migration Agent는 평균 응답시간을 줄여주었다. 실제로 지연이 발생 했을 때 Migration Agent 서버 응답 시간을 확인해보기 위해 PC가 지연을 인식하는 시간인 0.3초[10] 이상 응답 시간이 걸린 케이스들에 대한 결과를 추출해보았다.

Table 3. Without Migration Agent, Response time(sec)

PC	평균	표준 편차	분산
1,000	1.087	0.506	0.256
10,000	1.065	0.323	0.104
100,000	1.296	1.519	2.308

Table 4. With Migration Agent, Response time(sec)

PC	평균	표준 편차	분산
1,000	0.234	0.927	0.860
10,000	0.265	0.974	0.950
100,000	0.251	1.114	1.241

Table 5~Table 7은 Migration Agent 적용 여부에 따른 객관적인 성능 비교를 위해 0.3초 이상의 응답 시간을 보인 같은 수의 샘플을 무작위 추출한 결과이다.

Migration Agent 적용 여부와 관계없이 PC가 Field Server를 이동할 때 평균 응답 시간이 PC 숫자의 증가에 따라 증가하고 있음을 보여주고 있다.

Table 5~Table 6은 Migration Agent 적용에 따른 서버 응답 시간을 보여준다. Table 6을 통해 확인해볼 수 있는 것처럼 Migration Agent를 도입할 때 전반적인 응답 시간 감소 효과를 볼 수 있었다. Table 7은 PC 수가 100,000인 경우 평균 응답 시간이 증가하여 성능이 7.54% 감소하는 모습을 보여 주었는데 PC의 동기화를 위해 [6]의 Fig. 7과 같이 각 Field Server의 기본 응답 시간이 증가와 그로 인한 대기시간이 증가하기 때문이다.

하지만 전체 성능을 비교해보면 Migration Agent 서버가 전체 성능을 개선시키고 있다는 점을 분명히 확인할 수 있다.

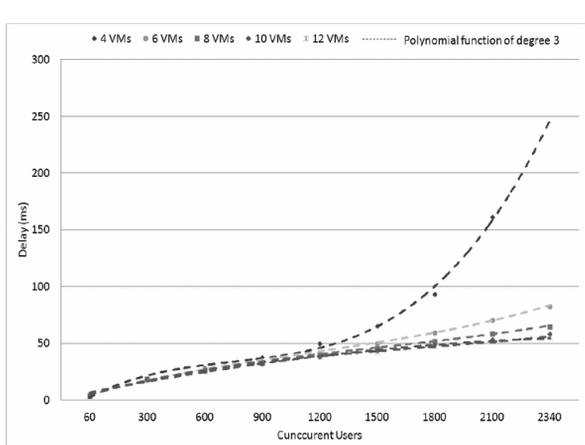


Fig. 7. Delay Time by number of PC [6]

Table 8~Table 10은 Migration Server를 적용 여부에 따른 전체 성능 비교 결과이다. Table 8은 Field Server가 DBMS를 거쳐 응답할 경우 대부분 0.3초 이상의 응답 시간이 걸린다는 것을 보여준다. Table 9은 캐시 역할을 수행하는 Migration Agent적용하는 경우 응답 지연 발생 비율이 약 7%로 줄어들게 됨을 보여준다. Table 10에서는 Migration Agent를 적용에 따른 지연 감소 비율을 보여준다. Migration Agent 평균 약 93%의 지연 감소 효과를 보여준다. 이러한 결과는 Table 7에서 보여주는 바와 같이 Migration Agent가 PC가 100,000 이상인 경우 Field Server 부하 증가로 인한 성능저하, 즉 Migration Agent 서버 응답 지연 가능성이 매우 낮다는 것을 보여준다. 하지만 이러한 현상도 충분이 발생이 가능하기 때문에 Field Server의 적정 부하 유지는 매우 중요한 사항이다.

Table 5. Without Migration Agent, delay time(sec)

PC	평균	표준 편차	분산
1,000	1.906	1.358	1.846
10,000	2.162	1.865	3.480
100,000	2.610	2.333	5.446

Table 6. With Migration Agent, delay time(sec)

PC	평균	표준 편차	분산
1,000	1.436	1.358	1.845
10,000	1.942	1.270	1.614
100,000	2.807	2.205	4.866

Table 7. Migration Agent and Non Migration Agent Compare

PC	평균	표준 편차	분산
1,000	24.65%	0	0.05%
10,000	10.17%	31.90%	53.62%
100,000	7.54% 감소	5.48%	10.65%

Table 8. Rate of Non Migration Agent System delay (response time over 0.3sec)

PC	지연발생	전체	비율
1,000	579,985	580,707	99.8%
10,000	4,990,158	4,994,163	99.9%
100,000	41,897,446	42,143,491	99.4%

Table 9. Rate of Migration Agent System delay (response time over 0.3sec)

PC	지연발생	전체	비율
1,000	33,147	491,505	6.7%
10,000	385,593	4,953,694	7.7%
100,000	3,501,255	42,567,820	8.2%

Table 10. Migration Agent System delay (response time over 0.3sec) reduce rate

PC	지연감소 비율
1,000	94.20%
10,000	92.27%
100,000	91.64%

5. 결 론

실험의 결과는 Migration Agent 서버가 Field Server 부하를 균등하게 유지하도록 도와주고 PC간 응답 시간 차이를 줄여 줌으로 PC와 Field Server 간 응답 시간에 따른 문제들을 개선하는 결과를 보여주었다. 또한 스택 구조의 Field Server는 특정 서버에 집중될 수 있는 부하를 분산시켰다.

앞으로 가상 세계에 대한 욕구는 계속 증가하게 될 것이다. VR 기기가 일상화되고 IoT 등의 인프라가 계속 진행될수록 가상 현실과 현실 세계의 차이는 계속 줄어들게 될 것이며, 이러한 현상이 진행될수록 이용자들은 가상 세계에서의 즉각적인 반응을 요구하게 될 것이다.

Migration Agent 서버는 Field Server 간 부하를 균등하게 하는데 도움이 되는 것을 확인하였다. 하지만 Field Server의 부하가 증가하는 경우 발생하게 되는 PC 응답 저하 문제는 해결하지 못하였다.

즉각적인 반응을 요구하는 상황에서는 0.3초 이내 응답이 이루어지지 않는 경우 이용자는 네트워크의 지연을 인지하게 된다. 향후 연구에서는 현실 세계와의 차이를 더욱 줄일 수 있도록 더 많은 PC를 지원하면서도 지연 발생 비율을 현재보다 더 개선한 Migration Agent 시스템을 개발하고자 한다.

References

[1] C. E. Dickerson, S. J. Clement, D. Webster, D. McKee, J. Xu, and D. Battersby, “A service oriented virtual environment for complex system analysis: preliminary report,” in *System of Systems Engineering Conference (SoSE), 2015 10th Date of Conference: 17-20*, pp.152-157, May, 2015.

[2] Seung-Wook Han, Hee-Suk Son, Byoung-Oh Kim, Seon-Yeong Han, and Dong-Man Lee, “Personal Genie: A Middleware System for Context-aware Spontaneous Interaction with Heterogeneous Smart IoT,” *Journal of KIISE*, Vol.39, No.4, pp.270-276, 2012.

[3] Jeremy R. Millar, Douglas D. Hodson, Gilbert L. Peterson, and Darryl K. Ahner, “Data Quality Challenges in Distributed Live-Virtual-Constructive Test Environments,” *Journal of Data and Information Quality*, Vol.7, Issue 1-2, June, 2016.

[4] Jong-Gwan Choi, Hye-Young Kim, and Won-Sik Woo, "A Study of a Game User Oriented Load Balancing Scheme on MMORPG," *Journal of Korea Game Society*, Vol.12, No.3, pp.69-76, 2012.

[5] Dongkee Won, Beobkyun Kim, Seungjong Chung, and Dongun An, "Design the time-interval based fairness partitioning method in DVE," *2007 International Symposium on Information Technology Convergence*, 23-24 Nov., 2007.

[6] Eya Dhib, Khaled Boussetta, Nawel Zangar, and Nabil Tabbane, "Modeling Cloud gaming experience for Massively Multiplayer Online Games," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, Issue Date: 9-12 Jan., 2016.

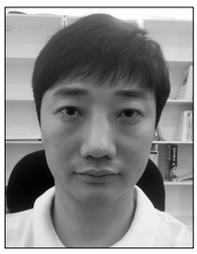
[7] Seok-Jong Yu, "A Study on an AOI Management in Virtual Environments Based on the Priority," *Journal of Korea Multimedia Society*, Vol.9, Issue 2, pp.189-196, 2006.

[8] SungWon Mun, "An Efficient Game Space Management Technique in Distributed Seamless Game Servers," Sogang University Graduate School of Information and Technology, 2004.

[9] Statics Korea [Internet], http://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1007.

[10] G. Johansson and K. Rumar, "Drivers' brake reaction times," *The Journal of the Human Factors and Ergonomics Society*, Vol.13, No.1, pp.23-27, Feb., 1971.

[11] J. Lui and M. Chan, "An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems," *IEEE Transaction on Parallel and Distributed Systems(2002)*, Vol.13, Issue 3, pp.193-211, Mar., 2002.



원 동 현

e-mail : dkwon@jbnu.ac.kr
2003년 전북대학교 컴퓨터과학과(공학사)
2006년 전북대학교 컴퓨터공학과(공학석사)
2009년~현 재 전북대학교 대학원
컴퓨터공학과 박사수료
관심분야 : DVE, 분산시스템, 그리드 컴퓨팅

안 동 언

e-mail : duan@jbnu.ac.kr
1981년 한양대학교 전자공학과(공학사)
1987년 한국과학기술원 전산학과(공학석사)
1995년 한국과학기술원 전산학과(공학박사)
1995년~현 재 전북대학교 대학원
컴퓨터공학과 교수
관심분야 : 정보검색, 한국어 정보처리

정 성 종

e-mail : sjchung@jbnu.ac.kr
1975년 한양대학교 전기공학과(공학사)
1981년 휴스턴대학교 전자공학과(공학석사)
1988년 충남대학교 전자공학과(공학박사)
1985년~2014년 전북대학교 대학원
컴퓨터공학과 교수
2014년~현 재 전북대학교 컴퓨터공학과 명예교수
관심분야 : 패턴인식 및 지능공학