Vol.7, No.11 pp.269~274 pISSN: 2287-5891 https://doi.org/10.3745/KTCCS.2018.7.11.269

Implementation and Performance Evaluation of Migration Agent for Seamless Virtual Environment System in Grid Computing Network

Dong Hyun Won† · Dong-Un An††

ABSTRACT

MMORPG is a role-playing game that tens of thousands of people access it online at the same time. Users connect to the server through the game client and play with their own characters. If the user moves to a management area of another server beyond the area managed by the server, the user information must be transmitted to the server to be moved. In an actual game, the user is required to synchronize the established and the transferred information. In this paper, we propose a migration agent server in the virtual systems. We implement a seamless virtual server using the grid method to experiment with seamless server architecture for virtual systems. We propose a method to minimize the delay and equalize the load when the user moves to another server region in the virtual environment. Migration Agent acts as a cache server to reduce response time, the response time was reduced by 50% in the case of 70,000 people.

Keywords : Migration Server, PC(Player Character), Seamless, Gird, AOI(Area of Interest), Cell

그리드 컴퓨팅 네트워크에서 Seamless 가상 환경 시스템 구축을 위한 마이그레이션 에이전트 구현 및 성능 평가

원 동 현† · 안 동 언††

요 약

MMORPG환경에서 이용자들은, 한정된 게임 월드보다 끊임없이 넓은 지형에서 플레이하기를 원하기 때문에 끊임없이 연속된 확장 가능한 공간을 제공해야 한다. 이때 사용자는 개개의 서버들로 구성된 게임 월드를 하나의 지역으로 인식하게 하고 분산 서버들이 구성한 게임 월드를 자유롭게 이동할 수 있어야 한다. 여러 대의 게임 서버를 분산 방식으로 구성하여 하나의 게임 월드를 제공하므로 개개 서버가 관리하는 영역을 벗어나 다른 서버의 관리지역으로 이동 시 이동에 따른 게임 캐릭터의 게임 관련 정보 등을 모두 전송해 주어야 한다. 서버 간의 정보 동기화를 위하여 복잡한 과정이 필요하고 이로 인한 응답 지연이 발생할 수 있다. 본 논문에서는 가상 환경을 구축하는 데 필요한 가상 환경 중 가상 환경 구축을 위한 마이그레이션 에이전트 서버를 제안한다. 마이그레이션 에이전트는 응답시간 단축을 위한 캐시 서버 역할을 수행하며, Player character수 70,000인 상황에서 응답시간을 50% 단축할 수 있었다.

키워드 : Migration Server, PC(Player Character), Seamless, Gird, AOI(Area of Interest), Cell

1. 서 론

가상 환경의 개념은 가상현실(Virtual Reality) [1]의 하위 개념으로, 가상현실이란 1989년 미국의 Jason Lancer가 처음 사용하면서 알려진 기술로, '컴퓨터가 만들어낸 감각 몰입이 이루어지는 가상 환경에서 이용자가 실시간으로 상호작용을 할 수 있는 인간-컴퓨터 인터페이스'이다[2, 3].

가상현실의 가장 대표적인 예는 World of Warcraft[4], 리니지[5]와 같은 MMORPG[6, 7]이며. 퀘이크[8]와 같은 1인칭 슈팅 게임도 기존 하나의 서버에서 플레이 하던 방식에서 다수가 하나의 가상현실을 즐기는 방식으로 변하고 있다. 서버 환경은 1인칭 슈팅게임과 MMORPG 방식 모두 끊임없이 넓은 지형에서 플레이하기를 원하는 욕구를 만족시키기 위해 변화하고 있다[9].

개개의 서버가 관리하는 지형 등 공간 위치와는 관계없이 사용자는 개개의 서버들로 구성된 게임 월드를 하나의 지역으로 인식하게 하고 분산 서버들이 구성한 게임 월드를 자유

† 준 회 원 : 전북대학교 컴퓨터공학과 공학박사
†† 종신회원 : 전북대학교 컴퓨터공학과 교수
Manuscript Received : April 4, 2018
First Revision : July 4, 2018
Accepted : August 27, 2018
* Corresponding Author : Dong-Un An(duan@jbnu.ac.kr)

룹게 이동할 수 있어야 한다[7].

사용자가 각각 분산된 영역을 담당하던 서버의 지역에서 다른 지역으로 이동하게 되므로 이동전의 서버에서 관리하던 사용자의 데이터는 필수적으로 이동한 서버로 전송되어야 하는데 정보동기화를 위해 복잡한 과정이 필요하다.

본 논문에서는 가상 환경 구축을 위한 마이그레이션 에이전트를 제안한다. 본 논문에서는 가상현실 구축을 위해 Seamless[10] 서버 구조를 실험을 위해, 그리드[11] 기반 Seamless 가상 환경 서버를 구축하고, 구축된 환경에서 발생할 수 있는 많은 문제 중 이용자들이 서버 이동 시 발생할 수 있는 지연 최소화에 대한 방법을 제시한다.

균등 부하 분산을 위해 서버 간 정보 동기화를 위한 캐시 서버 역할을 하는 에이전트 서버를 제안한다. 본 논문에서는 이러한 서버를 마이그레이션 에이전트라 칭한다.

본 논문의 2장에서는 Seamless 가상 환경 구축을 위한 최근 연구에 대해 소개하고 3장에서는 본 논문에서 제안하는 마이그레이션 에이전트를 설명한다. 4장에서는 마이그레이션 에이전트 적용에 따른 차이점을 보여준다.

2. 관련 연구

2.1 그리드 기반 가상 환경

Fig. 1은 그리드 기반 가상 환경 시스템이다. 각 서버가 네트워크를 통해서 특정 지역을 관리하는 방식이다. Fig. 2는 실제 모니터 계층[11]에서 처리하게 되는 Player Character (PC)들을 그림으로 나타낸 것이다.

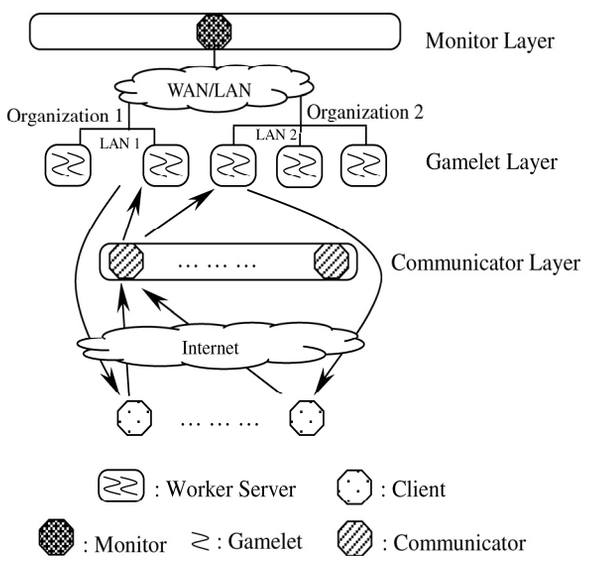

Fig. 1. Three-Layer Model

Fig. 2와 같이 분포된 PC들을 Fig. 3과 같이 각 지역을 담당하는 서버들이 처리하게 된다. 특정 지역에 많은 부하가 걸릴 경우에 맞게 미리 가상 세계를 설계하고 그것에 맞게 Field Server들을 할당하게 된다.

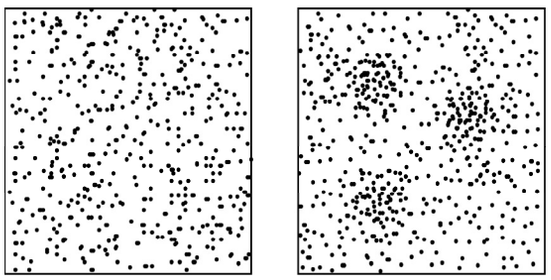

Fig. 2. PC Distribution Example [11]

이 방법은 시나리오에 맞게 Field Server를 배치하고 시나리오가 추가되어 가상세계를 확장하는 데 유용하나 한 지역에 너무 많은 PC가 접속할 경우 부하 분산에 어려움이 있다는 단점이 있다.

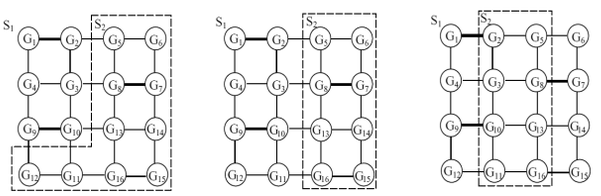

Fig. 3. Examples of Specific Area Processing [11]

2.2 Seamless 가상 환경 서버 아키텍처

가상 환경을 위한 서버 동기화 방식은 여러 가지가 있다. 그중 크게 채널방식과 지역별 서버 할당 방법이 있다[12-14]. 채널이란 하나의 게임 데이터 데이터베이스를 공유하는 여러 개의 Field Server가 평행하게 존재하여, 사용자는 그중 하나를 선택하여 접속하는 것이다. 각 채널은 거의 완벽하게 독립된 세계이고 채널 사이의 상호작용도 없으므로 이 방식을 구현하는 데 드는 추가적인 노력은 거의 없다고 할 수 있다. 그러나 단순히 사람이 모이는 것이 아니라 주변 사람과의 관계를 꾸준히 지속시켜 나가야 하는 것이 중요한 가상현실에서 이 방식을 사용하는 것은 그리 좋은 선택이 아니다.

이와 달리 지역별로 서버를 할당해서, 한 서버가 해당 지역에 있는 사용자들의 정보를 처리하는 지역별 서버 할당 방법이 있다. 존(zone)이 확실히 구분되는 경우에는 자연스럽게

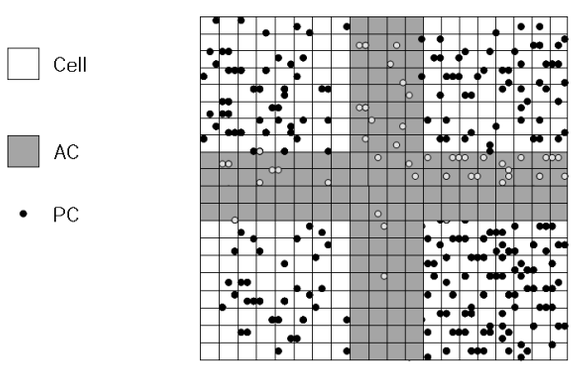

Fig. 4. Seamless Map

사용할 수 있는 방법이다. 사용자가 다른 지역으로 이동하는 경우 클라이언트에서 로딩이 일어나기 때문에 담당 서버의 전환도 이때 같이 처리해 주면 된다. 하지만 별도의 존(지역) 구별이 없는 Fig. 4와 같은 Seamless 게임일 경우에는 문제가 복잡해진다. AC는 Field Server와 Field Server간 인접지역을 의미하는데 이 위치에 있는 PC는 인접하고 있는 Field Server와 정보를 공유해야 한다.

2.3 Seamless migration

Seamless 환경에서 PC를 이전하는 일반적인 방식은 Fig. 5와 같다[15]. PC가 서버 A 지역에서 서버 B 지역으로 이동하는 경우, PC는 서버 A로부터 정보를 전달 받다가 PC의 AOI 범위가 A 서버의 경계를 벗어나는 지점부터 B 서버로부터 정보를 전달 받는다. PC는 A서버를 완전히 벗어나기 전까지 A,B 서버 모두로부터 정보를 전송 받는다.

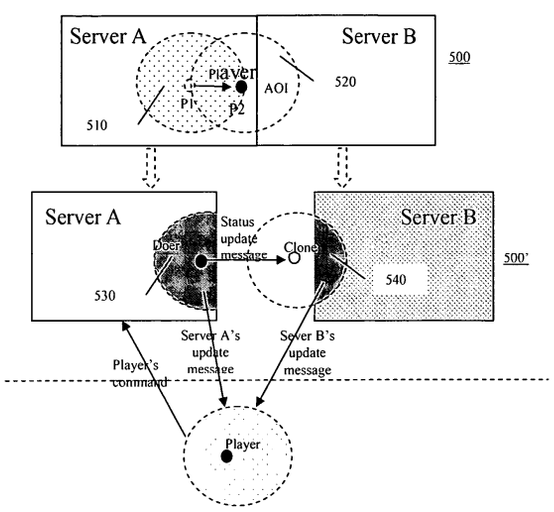

Fig. 5. Seamless Server Migration

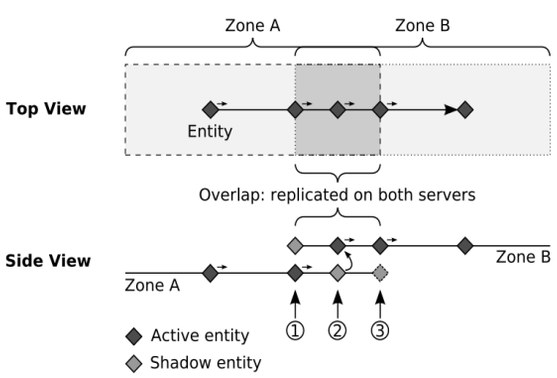

Fig. 6. Overlapping Zones for a Seamless Migration of an Entity Between Two Zones

[16]에서는 PC의 서버 이전을 위해, 서버 경계 인접 지역을 겹치는 방법을 제안한다. Fig. 6과 같이 PC가 서버의 인접 지역으로 이동하게[16] 되면 서버는 PC를 활성 개체와 비활성 개체로 생성한다. PC의 이동방향에 따라 실제 이동하고자 하는 방향의 서버의 PC는 활성개체가 되며 이전 서버의 PC 정보는 비활성된다.

3. 균등부하 분산을 고려한 마이그레이션 에이전트 제안 및 구현

3.1 마이그레이션 에이전트 기능

사용자 간 정보 동기화를 위해 인접하고 있는 모든 사용자의 동기화를 위해 Field Server 수 • $O(n^2)$의 연산이 필요하게 된다. 이러한 문제를 해결하기 위해 제안된 Migration 서버는 동기화를 위한 PC 정보를 따로 관리하고 있다. 사용자의 정보가 Field Server 별로 필요한 경우 Migration 서버가 각 정보를 제공하는 형태이다. 하지만 이러한 방식도 사용자 정보 동기화 요청이 늘어날수록 부하가 증가하게 되어 응답 지연시간이 늘어나게 된다.

이러한 문제를 해결하기 위해 본 논문에서는 Seamless 가상 환경에서 균등부하분산을 고려한 마이그레이션 에이전트를 제안한다. 제안하는 방식은 기존 사용자 정보를 따로 관리하는 마이그레이션 서버기능을 축소하여 단순한 캐시서버 임무를 수행한다. 실제 사용자 정보는 각 Field Server에서 관리하며 마이그레이션 에이전트는 단순히 위치 정보만을 제공함으로 사용자 간의 상호 작용이 없는 경우에는 각 Field Server 간의 데이터 동기화를 수행하지 않는다. Table 1은 각 방식의 차이점을 보여준다.

Table 1. Migration Server Comparison

Division	Migration Server	Migration Agent Server
PC synchronization	All AOI PCs	All AOI PCs
management system	Same as Field Server	Manage PC location information in memory

마이그레이션 서버와 마이그레이션 에이전트는 PC의 AOI(Area Of Interest) 범위 내 모든 PC의 정보를 동기화 하는 점은 동일하지만, 마이그레이션 서버의 경우 동일한 정보를 AOI범위 내 모든 Field Server들이 공유해야 한다. 하지만 마이그레이션 에이전트는 단순히 PC의 위치정보만 관리하고 있다가 실제 PC간 상호 이벤트가 발생하는 경우만 Field Server에 PC 정보를 동기화 한다.

3.2 마이그레이션 에이전트 구현

Fig. 7은 실험을 위해 설계된 서버 환경이다. 물리적 서버에 Ubuntu 16.04 LTS를 설치하고 KVM을 설치하였다. KVM으로 생성된 가상 서버에는 Field Server 및 마이그레이션 에이전트를 구현하기 위해 Ubuntu 16.04 LTS를 설치하였다.

Fig. 8은 마이그레이션 에이전트가 PC의 정보를 Field Server간 및 사용자에게 동기화하는 과정을 보여준다. Field

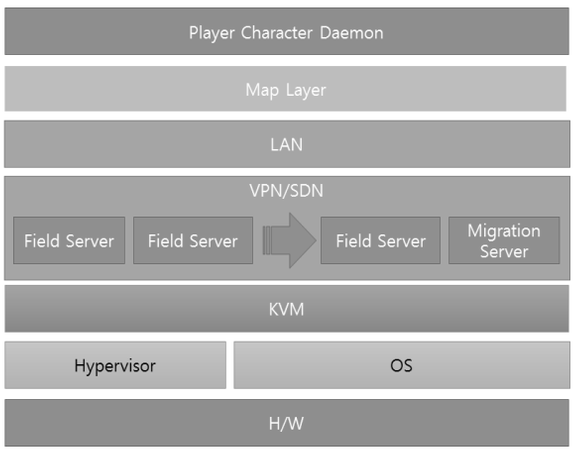

Fig. 7. Overall System Configuration Diagram

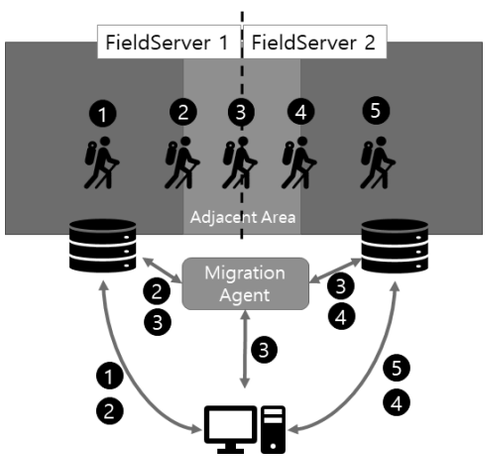

Fig. 8. Migration Agent's Field Server PC Information Synchronization

Server 1에서 Field Server 2로 이동하는 과정은 다음과 같다. ①②PC가 인접지역에 진입하기 전까지 해당 Field Server 1이 관련 정보를 사용자에게 전달한다. 이때 Field Server1과 PC간의 정보가 동기화 된다. ②③④ PC가 인접지역에 진입하면 마이그레이션 에이전트는 해당 정보를 인접 Field Server와 사용자에게 전달한다. 이때 마이그레이션 에이전트는 Field Server간에 PC의 위치 정보를 제공한다. ④⑤ PC가 인접지역을 지나 Field Server 2로 이동하면 Field Server 2가 PC 정보를 사용자에게 전달한다. PC가 완전히 Field Server간 인접지역을 벗어나게 되면 마이그레이션 에이전트로부터 PC의 정보를 제공받지 않는다. 사용자가 PC의 Field Server간 이동을 인지하지 못하도록 PC가 ②④와 같이 인접지역 경계 부분에 존재하는 경우 PC 정보는 Field Server와 마이그레이션 에이전트가 공동 관리한다.

Fig. 9는 마이그레이션 에이전트 Pseudo code이다. 마이그레이션 에이전트는 PC의 위치정보를 캐시하고 해당 PC의 정보를 요청하는 경우 그 위치를 돌려보내기 위해 Send_User_Data_to_Field Server 이벤트와 Send_User_Data_to_Client 이벤트를 수행한다.

```
Start MigrationAgent

MigrationServer MS = MigrationServerThreadPool();

if(MS.isFull == false )
  MS.Run();

End MigrationAgent

MigrationAgentr :
 while(true)
   Client = open.Connection()
   ResultInfo = MigrationServer.GetUserInfo(Client)
   if(!ResultInfo)continue
   if( Client.Action == Send_User_Data_to_FieldServer )
     response.to.server = Client.FieldServerLocation
     response.to.client = GetInfoFromMemory(Client)
     ResultInfo = response.result
   if( Action == Send_User_Data_to_Client )
     response.to.client = GetInfoFromMemory(Client.Location)
     ResultInfo = response.result

   RESPONSE(ResultInfo)
```

Fig. 9. Migration Agent Server Pseudo Code

Send_User_Data_to_Field Server 이벤트는 PC 정보를 요청하는 Field Server에게 전달하는데 데이터베이스로부터 정보를 읽어오는 대신 메모리에 저장된 값을 불러온다. PC 정보를 요청한 Field Server는 동기화와 같이 PC의 최신 정보가 필요한 경우 해당 Field Server에게로 요청하며 동기화가 필요하지 않은 경우에는 PC에게 단순 위치 정보만을 제공한다. Send_User_Data_to_Client 이벤트는 PC 간 정보 요청 시 PC의 위치 정보를 제공한다. 만약 PC와의 동기화가 필요한 경우 PC의 정보를 요청한 PC는 PC 정보를 저장하고 있는 Field Server로부터 정보를 요청한다.

4. 실험 및 평가

4.1 실험환경

- 접속시간 : 일반적인 가상 환경인 게임의 경우 평균 하루 2시간의 이용 시간을 가진다[17]. 본 논문에서도 평균 2시간을 기준으로 접속시간을 적용하였다.
- Map 크기 : 가로세로 10Km의 중소형 도시 크기를 기준으로 Cell 당 50cm를 할당하였다
- AOI : 사람의 음성이 0.1초 이내 도달 가능한 거리인 34.4m를 기준으로 PC를 중심으로 상하좌우 70 Cell을 할당하였다.
- PC의 수: 10,000 30,000 50,000 70,000 90,000을 할당 하였다.
- 서버환경 : 실험을 위해 총 4대의 가상 서버를 생성하였다. 테스트를 위한 가상 서버는 2대의 인텔 제온 E3-1280V (8M캐시,3.7GHz) 32GB RAM으로 구성된 워크스테이션에 Ubuntu 16.04 LTS 버전을 설치하고 가상 서버로 KVM을 적용하였다. 각 서버에는 3대의 Field Server를 설치하고 core2개 RAM 4G를 할당하였다. 총 6대의 서버 중 4대는 Field Server, 1대는 마이그레이션 에이전트 Server, 1대는 PC 접속을 위한 트래픽 발생 서버로 사용하였다.

Field Server와 PC간 교환 정보는 Table 2와 같다.

Table 2. Messages between Server and Client

Message	Description
S_MOVE	PC move request message
S_VELOCITY_CHANGE	PC movement speed processing message
S_STOP	PC stop request message
S_MESSAGE	Request a message
S_REQUEST_MIGRATION	Request for PC movement between Field Servers
S_RESPONSE_MIGRATION	Request response for PC movement between Field Servers
S_REQUEST_FIELD	Adjacent Field Server Request
S_RESPONSE_FIELD	Response of an adjacent Field Server
S_REGIST_MIGRATION	Request to register with Migration Server
S_REPLY_REGIST_MIGRATION	Responding to Migration Server Registration Requests
S_UPDATE_STATUS	Tells the Field Server when its state changes.

4.2 마이그레이션 에이전트 성능평가

Table 3은 마이그레이션 에이전트 적용에 따른 Field Server의 응답시간 차이를 보여준다. PC 응답시간은 마이그레이션 에이전트를 적용할 경우 그렇지 않은 경우보다 더 좋은 응답시간을 보이고 있으며 특히 PC의 수가 많아질수록 더 좋은 성능을 제공하고 있음을 알 수 있다. 하지만 PC 30,000과 50,000에서 약 9%성능 감소를 보이고 있는데 Field Server에 적정 수준의 PC가 존재하는 경우 마이그레이션 에이전트는 오히려 0.03초 응답시간을 증가시켰다. Table 4를 보면 Field Server의 CPU 이용률 60%~70%인 경우 마이그레이션 에이전트 없이도 좋은 성능을 보이고 있지만 CPU 이용률이 떨어지기 시작하면서부터 마이그레이션 에이전트로 인한 성능 개선 효과를 보였다.

CPU 이용률은 10,000~50,000까지 증가하다가 70,000~90,000은 감소하고 있다. 반면 마이그레이션 에이전트는 지속해서 이용률이 증가하고 있었다. 분석결과 데이터베이스에 접근하여 데이터를 가져오는 동안에는 CPU 대기 시간이 증가하여 실제 CPU 이용률이 내려간 것으로 확인되었다. 실제 데이터베이스 접속이 늘어나는 70,000부터 CPU 이용률이 감소하고, Field Server에 PC 정보를 전달하는 Migration Server의 CPU 이용률이 올라가고 있음을 볼 수 있다.

Table 3. Average Response Time Comparison(sec)

PC	Unapplied	Apply	Performance improvements
10,000	0.246	0.243	1.21%
30,000	0.274	0.301	-9.85%
50,000	0.336	0.369	-9.82%
70,000	0.904	0.450	50.22%
90,000	1.588	1.166	26.57%

Table 4. Average Server CPU Utilization by Migration Agent

PC	Unapplied	Apply	Migration Agent
10,000	53%	56%	48%
30,000	66%	64%	52%
50,000	71%	66%	68%
70,000	63%	70%	87%
90,000	56%	68%	91%

마이그레이션 에이전트를 적용한 경우 PC 증가에 비례하여 마이그레이션 에이전트와 Field Server의 CPU 이용률이 올라가고 있음을 볼 수 있고 반면 적용하지 않은 경우 CPU 이용률이 낮아지고 있음을 확인 가능한데 이를 통해 마이그레이션 에이전트가 Field Server의 이용률을 높여 주고 있다고 할 수 있다.

전체 성능 비교를 위해 시스템별 CPU 이용률과 마이그레이션 에이전트 적용에 따른 성능 차이를 하나의 그림으로 나타내었다. 그림에서 Agent라는 문구가 추가된 그래프는 마이그레이션 에이전트를 적용하였다는 의미이며 아래 x축의 1~5는 10,000~90,000까지의 PC 수이다. 막대그래프는 응답시간을 나타내며 꺾은선은 CPU 이용률을 나타낸다.

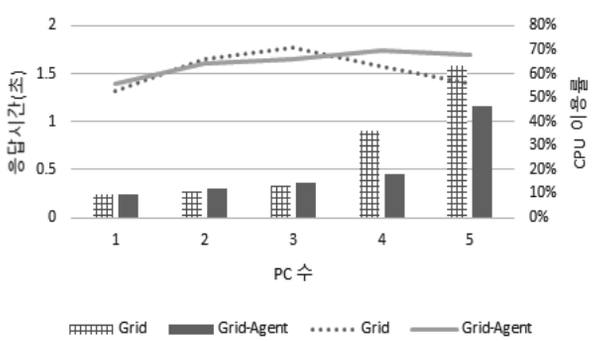

Fig. 10. Performance Comparison According to Migration Agent Application

Fig. 10을 살펴보면 마이그레이션 에이전트를 적용했을 때 4지점 즉 PC 70,000인 지점에서의 Field Server CPU 이용률 저하를 마이그레이션 에이전트가 보완하여 Field Server의 CPU 이용률을 높여 주었다. 그로 인해 PC에 대한 응답 시간을 감소시켜주는 효과가 있었다. 하지만 PC 수가 90,000인 경우 전반적인 성능 개선에 대한 효과를 기대하기 어려울 것으로 예상된다. Gird 방식의 Seamless 가상 환경 시스템의 경우 하나의 Field Server로 이용자가 몰리는 경우 인접한 Field Server가 부하를 담당해주지 못하기 때문에 응답시간이 증가하게 된다.

5. 결 론

본 논문에서는 그리드 방식 Seamless 가상 환경에서 PC에 대한 응답시간을 단축하게 하고 Field Server 간 효율적인 운

용을 위해 마이그레이션 에이전트를 적용하고 테스트하였다.

마이그레이션 에이전트를 사용함으로 응답 지연을 줄일 수 있었다. 하지만 특정 지역을 담당하는 Field Server를 따로 두는 그리드 방식 Seamless 가상 환경의 경우 PC가 증가하는 경우 마이그레이션 에이전트는 제한적인 성능 개선효과만을 보였다.

추후연구에서는 더 많은 PC를 지원하면서도 지연 발생 비율을 현재보다 더 개선한 마이그레이션 에이전트 시스템을 개발하고자 한다.

References

[1] Wikipedia [Internet], https://en.wikipedia.org/wiki/Virtual_reality.

[2] R.A. Earnshaw, “Virtual Reality Systems,” RA Earnshaw - 2014.

[3] K.-B. Park, and J. Y. Lee, “Comparative Study on the Interface and Interaction for Manipulating 3D Virtual Objects in a Virtual Reality Environment,” *Transactions of the Society of CAD/CAM Engineers*, Vol.21, No.1, pp.20-30, Mar. 2016.

[4] World of Warcraft [Internet], https://worldofwarcraft.com

[5] Lineage [Internet], https://lineage.plaync.com/

[6] Helena Cole and Mark D. Griffiths, “Social Interactions in Massively Multiplayer Online Role-Playing Gamers”, *CyberPsychology & Behavior*, Vol.10, No.4, pp.575-583, Aug. 2007,

[7] Bo-Ri Choi, Nam-Choon Park, “Definition of virtual reality service design prototyping system based on Head Mounted Display,” *International Design Conference of KSDS and ADADA with Cumulus*, 2015.

[8] Quake [Internet], https://quake.bethesda.net

[9] Tao Ni, Hongyan Zhang, Changzhi Yu, Dingxuan Zhao, and Songyue Liu, “An International Journal for All Aspects of Design,” *Computers & Electrical Engineering Volume*, Vol.39, Issue 7, pp.2112-2123, Oct. 2013.

[10] Nicole Yankelovich, Bernard J. Haan, Norman K. Meyrowitz, and Steven M. Drucker, “Intermedia: The Concept and the Construction of a Seamless Information Environment,” http://research.microsoft.com/en-us/um/people/sdrucker/papers/intermedia1.pdf

[11] T. Wang, C. L. WangFrancis, and C. M. Lau, “An architecture to support scalable distributed virtual environment systems on grid,” *The Journal of Supercomputing*, Vol.36, No.3, pp.249, Jun. 2006.

[12] Chul-Min Lee, Hong-Seong Park, “Virtual Data Grouping for Performance Enhancement of Multi-User Games,” The Jouranl of Korean Institute of Information Scientists and Engineers, Vol.30, , No.3・4, pp.231-238, 2003.

[13] Jeongjin Lee, Gilsoo Doo, Dongun Ann, Seungjong Chung, “Design of Dynamic Map - Divide System for Load Distribution of MMORPG (Massively Multi - player Online Role Playing Game)” Korea Computer Congress 2005.

[14] Dae Un Lee, “Cloud Computing,” Auto Journal, Jouranl of Korea Society of Automotive Engineers, Vol.33, No.7, pp.64-65, 2011.

[15] US20060258462 A1, US 11/403,024.

[16] Frank Glinka, Alexander Ploß, Jens Müller-lden, and Sergei Gorlatch, “RTF: a real-time framework for developing scalable multiplayer online games,” *Proceeding NetGames '07 Proceedings of the 6th ACM SIGCOMM Workshop on Network and System Support for Games* (pp.81-86), ACM, 2007.

[17] Wen-Chi Kuo, Shih-Ting Wang, and Jie-Chi Yang, “An Empirical Analysis of the Playing Time by Different Genders and Ages in an MMORPG,” Digital Game and Intelligent Toy Enhanced Learning (DIGITEL), *2012 IEEE Fourth International Conference on Date of Conference*: 27-30 Mar. 2012.

원 동 현

https://orcid.org/0000-0002-2952-9405

e-mail : dkwon@jbnu.ac.kr

2003년 전북대학교 컴퓨터과학과(공학사)

2006년 전북대학교 컴퓨터공학과(공학석사)

2017년 전북대학교 컴퓨터공학과(공학박사)

관심분야 : DVE, 분산시스템, 그리드 컴퓨팅

안 동 언

https://orcid.org/0000-0002-4145-0732

e-mail : duan@jbnu.ac.kr

1981년 한양대학교 전자공학과(공학사)

1987년 한국과학기술원 전산학과(공학석사)

1995년 한국과학기술원 전산학과(공학박사)

1995년～현 재 전북대학교 컴퓨터공학과 교수

관심분야 : 정보검색, 한국어 정보처리