한국항행학회 논문지 제 15 권 제 6 호 2011년 12월

분산형 서버 구조 기반 Map 밸런스 서버를 이용한 게임 서버 간 부하 관리 방법

A Management method of Load Balancing among Game Servers based on Distributed Server System Using Map Balance Server

김순곤*, 이남재**, 양승원***

Soon-Gohn Kim*, Nam-Jae Lee**, and Seung-Weon Yang***

요 약

일반적으로 분산 형 구조의 게임서버시스템 하에서 게임 배경처리는 일정크기로 나누어진 여러 개의 부분 배경들을 다수의 게임서버가 나누어서 처리한다. 그런데 분할된 게임 배경에 대한 게임 사용자들의 선호도가 사용자의 특성에 따라 다르게 나타나기 때문에 모든 게임 배경 내 사용자들의 분포를 일률적으로 만들기는 매우 어렵다. 이 때문에 캐릭터들이 한 장소에 급격히 집중되어 게임이 진행되는 경우, 서버가 처리할 수 있는 한계를 넘어 시스템이 일시적으로 다운되는 문제가 발생 할 수 있으며, 그 반대의 경우 수행할 캐릭터가 없는 상황에서도 배경처리를 계속 수행해야 하므로 게임서버의 효율이 상당히 떨어지게 된다. 이를 해결하기 위하여 본 논문에서는 Map 밸런스 서버를 이용하여 사용자 처리를 위한 부하를 비교적 균등화 시킬 수 있는 Map 관리 방법을 제안 하였다. 제안한 모델 하에서는 사용자가 활동하지 않는 게임 내 공간 처리를 일시 중지시키는 방법으로 게임 서버의 부하를 감소시킬 수 있으며, 서버 간 처리하는 배경을 새로 할당하여 부하를 재분배함으로써 서버들의 효율을 극대화할 수 있다.

Abstract

Generally, In distributed online game server system, game maps are processed separately by means of dividing into several unit blocks. But the keeping normal distribution of user in game map is very difficult because preferences of game users are not same according to individual user's private character. For this reason, if the huge number of users concentrate on particular region of same game map at once, the game server exceed their threshold so that the system can be getting down. Conversely, the efficiency of system goes down considerably because the game server must perform map processing continuously even under user_empty situation. To solve this problem, in this paper, we propose a Map management method to control relatively normal distribution of users in game maps using Map Balance Server. In suggested model, we can reduce the load of game servers by means of turn off the game map processing temporary when a server is under user_empty situation. we also can maximize server performance by means of redistribution of map processing load among servers.

Key words : Distributed System, Map Balancing, Online Game Server

* 중부대학교 컴퓨터학과(Dept. of computer, Jungbu Univ.)
** (주)엘맥스
*** 우석대학교 게임콘텐츠학과 (Dept. of Game and Contents, Woosuk Univ.)
· 제1저자 (First Author) : 김순곤, · 교신저자(Corresponding Author) : 양승원
· 투고일자 : 2011년 11월 11일
· 심사(수정)일자 : 2011년 11월 11일 (수정일자 : 2011년 12월 26일)
· 게재일자 : 2011년 12월 30일

Ⅰ. 서 론

최근에 PC의 비약적인 발전과 초고속 네트워크 기술의 지원으로 네트워크를 이용한 온라인게임들이 보다 다양한 형태로 발전되어가고 있다[1].

온라인 게임은 기존의 클래식 게임을 온라인 화한 테트리스 같은 고전 게임, PC 패키지게임을 기반으로 개발 한 후 여기에 네트워크 기능을 추가하여 게임의 흥미와 변화를 추구하는 스타크래프트 같은 리테일 하이브리드(Retail Hybrid)게임 및 리니지와 같은 MMORPG 계열의 PW(Persistent World)게임 등으로 크게 구분할 수 있다[1][2].

이 중 MMORPG들은 현재 제작비가 거의 100억에 가까운 대작들이 계속 제작되고 있으며 이에 따라 게임 그래픽 연출이 보다 화려해지고 다양한 게임 기능이 추가 되었으며 게임 캐릭터가 게임 스토리를 통하여 진행하는 게임 배경의 크기도 계속해서 확장되고 있는 추세이다[1][3][4].

이렇게 방대해진 게임의 세계를 구현하기 위해서는 초기 소규모의 PW 온라인게임처럼 그림 1과 같이 1대의 서버가 모든 일을 처리 하는 단일 서버구조에서는 현재와 같이 대규모의 사용자를 처리하기에는 불가능하다[2][5][6][7].

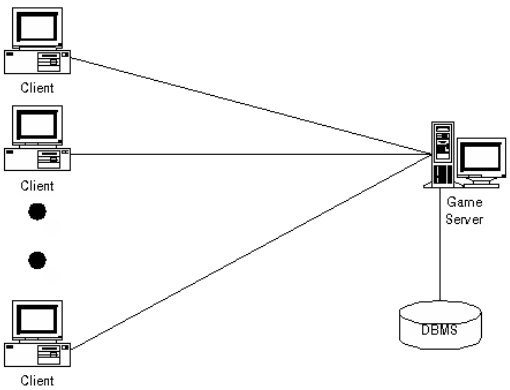

그림 1. 단일 서버구조
Fig. 1. Single Server Structure

이를 해결하기 위하여 그림 2와 같은 여러 개의 서버가 하나의 클러스터로 묶여서 하나의 게임배경을 처리하는 대칭형 클러스터링구조와 그림 3과 같이 게임서버가 수행하는 기능들을 독립적으로 분산 운용하여 각 기능을 독립적인 서버가 담당하도록 하는 분산형 다중서버구조의 채택이 대세를 이루고 있다[2].

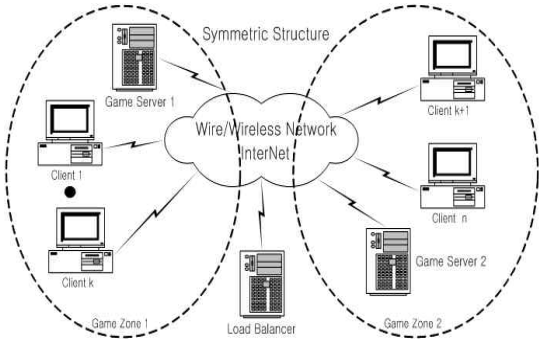

그림 2. 클러스터링 기법의 대칭형 다중 서버구조
Fig. 2. Symmetric Multi-Server Structure Using Clustering Method

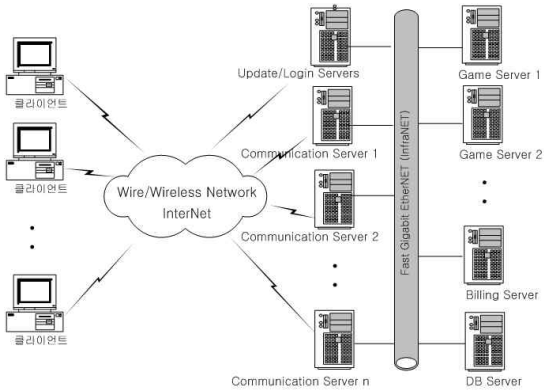

그림 3. 분산 기법의 비대칭형 다중 서버구조
Fig. 3. Asymmetric Multi-Server Structure Using Distributed Method

대칭형 서버는 그림 2에서 보는 바와 같이 게임클러스터에 속해있는 서버들이 하나의 집합개념을 가지고 동작하기 때문에 서비스 비용이 사용자 수 증가에 따라 선형적으로 증가한다.

그렇기 때문에 만약 서버클러스터 당 가능한 동시 접속자의 수가 2000명이라고 가정하면 2001번째 사용자가 생길 경우 비슷한 사용자에 대해 서비스비용은 2배를 넘게 된다. 따라서 게임 서버클러스터의 효율이 떨어지는 경향이 있다.

이를 해결하기 위해서는 클러스터 당 처리 가능한 동시 접속 사용자의 수를 조정하는 부가의 작업이 필요하다.

이에 반하여 분산형 구조의 게임 서버 시스템은 그림 3과 같이 하나의 게임을 운영하는 데 필요한 기능요소 즉, 캐릭터생성/삭제, 게임 Map 관리, 채팅,

거래 등을 위한 인증, 게임 룰 처리, 통신접속 및 사용자 정보관리에 대한 기능을 분리한 후, 이들을 각각의 독립적으로 담당하는 로그인/업데이트서버, 통신서버, 게임서버, 데이터베이스서버 등으로 분산시킨 후 각 서버들을 초고속의 내부 망으로 연결하여 게임을 처리하는 구조이다.

이와 같은 분산형 게임서버구조는 사용자의 수가 증가하였을 때는 통신서버의 수를 증가시켜서 접속에 대한 부하를 분산 시키고, 게임배경이 커지면 이를 담당할 추가의 게임 서버를 이용해 사용자의 증가에 따른 시스템의 부하를 유연하게 조절할 수 있다.

이러한 장점 때문에 분산형 게임 서버구조는 게임의 배경을 비교적 저가의 서버를 사용하여 쉽게 확장시킬 수 있을 뿐만 아니라 클러스터링 기법에서 사용하는 수천만 원에서 수억 원을 호가하는 고성능의 대형 서버 대신 가격이 저렴한 소형 서버를 사용하는 방법으로 비용 절감이 가능하다[2][3][8].

하지만 분할된 게임 배경에 대한 게임 사용자들의 선호도가 사용자의 특성에 따라 다르게 나타나기 때문에 모든 게임 배경 내 사용자들의 분포를 일률적으로 만들기는 매우 어렵다.

이 때문에 캐릭터들이 한 장소에 급격히 집중되어 게임이 진행되는 경우, 서버가 처리할 수 있는 한계를 넘어 시스템이 일시적으로 다운되는 문제가 발생할 수 있으며, 그 반대의 경우 수행할 캐릭터가 없어도 배경처리를 계속 수행해야 되므로 게임서버의 효율성이 상당히 떨어지게 된다[9][10].

이를 해결하기 위하여 본 논문에서는 Map 밸런스 서버를 이용하여 사용자 처리를 위한 부하를 비교적 균등화 시킬 수 있는 Map 관리 방법을 제안하였다.

Ⅱ. 분산형 서버 시스템 구조 비교 분석

본 장에서는 참고문헌 [2]에서 제안한 기존 분산형 구조의 온라인 게임 시스템과 본 논문에서 제안한 Map 밸런스 서버의 기능과 동작을 비교하였다.

2-1 기존 분산형 게임서버 구조

기존의 분산 형 온라인 게임서버시스템의 물리적인 연결 구조는 그림 3에 나타나 있다. 그림 3에 나타난 구조는 그림 4와 같은 논리적인 구조로 다시 재구성할 수 있다.

기존의 분산시스템은 그림 4에서 보는 바와 같이 사용자측의 클라이언트 1-tier와 서비스 측의 업데이트/로그인, 통선서버, 게임 서버로 3-tier 구조로 되어 있다.

제안한 Map 밸런스 서버를 적용한 분산 형은 기존 분산 형 서버시스템을 기반으로 원안에 표현된 별도의 Map 밸런스 서버를 추가한 형태로 그림 5에 나타나 있다.

[2]에 기술된 분산 시스템의 동작은 다음 5단계로 정리할 수 있다.

1단계 : 게임을 시작하기 위하여 사용자 PC안에서 구동되는 클라이언트 프로그램을 통하여 사용자는 1차적으로 업데이트/로그인 서버에 접속한다.

2단계 : 사용자가 업데이트/로그인서버에 접속하면 서버는 데이터베이스 서버에 접속하여 사용자를 인증하여 새로운 계정을 만들어 주거나 기존계정이 있을 시에 해당 사용자가 등록한 게임 캐릭터의 정보를 클라이언트에 넘겨주고 부하가 적은 통신서버로 클라이언트를 연결시켜준 다음 클라이언트와의 통신을 단절한다.

3단계 : 업데이트/로그인서버에 의해서 클라이언트와 연결된 통신서버는 클라이언트를 통해 사용자가 가고 싶은 곳의 지형정보를 받아 분석한 후 해당 게임 서버로 자료를 전송하여 처리하게 하고, 계속 클라이언트와 게임 서버와의 송수신 채널을 확보하여 자료를 전달한다. 만약 채팅과 같이 게임 서버가 직접 간여할 필요가 없는 인스턴스 메시지는 게임 서버에 전송하지 않고 통신서버만을 통해 메시지를 전달한다.

4단계 : 게임서버는 해당 캐릭터가 자신이 처리하는 게임 배경 내에 머물러 게임을 수행할 경우는 계속해서 통신서버를 통해 클라이언트와 게임 진행을 수행한다. 하지만 만약 게임 캐릭터가 해당 서버가 처리하는 배경 밖으로 이동하고자 할 때에는 현재 클라이언트의 정보를 통신서버로 이송한다. 통신서버는 이 정보를 처리하여 사용자가 이동하고자 하는 해당 배경을 담당하는 서버로 자료를 넘기고 채널을 생

성하여 클라이언트가 재접속을 하지 않고도 배경을 이동할 수 있게 한다.

5단계 : 클라이언트가 네트워크상에서 연결이 단절되었거나 클라이언트의 종료 신호를 게임 서버가 받으면 현재 메모리 안의 자료를 데이터베이스 서버에 전송하여 사용자 정보를 갱신시키고 해당 아이디를 메모리에서 제거한다. 이러한 분산 구조에서는 사용자가 증가할 경우, 대부분의 부하가 접속처리에서 이루지는 만큼 통신서버 만을 증가시킴으로 문제를 쉽게 해결할 수 있다. 또한 게임 배경이 시나리오 추가나 이벤트에 의해서 확장될 경우에 그 배경을 담당하는 서버만 추가 시키면 되기 때문에 사용자의 증가 및 게임 배경의 처리 등에 유연하게 동작 할 수 있다.

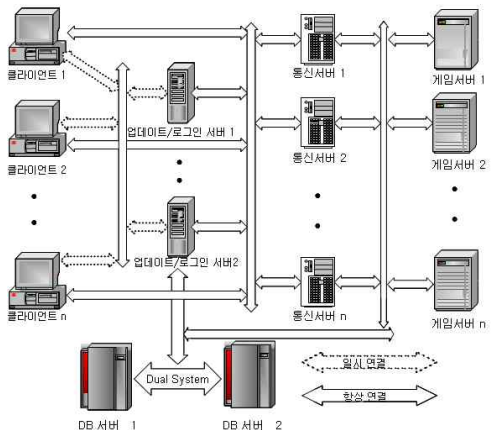

그림 4. 기존 분산형 서버의 논리적 구조
Fig. 4. Logical Structure of Existing Distributed Server

하지만 이 구조는 전체의 게임 배경을 분할하여 분할된 배경을 여러 서버가 함께 나누어 처리하는 방식으로 많은 사용자들이 한 배경에 갑자기 몰려들었을 경우 해당 게임 서버의 부하가 급격히 증가하여 최악의 경우 시스템 다운이 일어 날 수 있으며 반대로 어떤 배경 안에 캐릭터가 거의 존재하지 않을 경우 시스템은 거의 사용되지 않아 자원을 낭비하게 된다. 이를 해결하기 위하여 Map 밸런스 서버를 기존 시스템에 추가하여 기존 분산시스템이 가지는 단점을 해결하였다.

2-2 제안한 Map 밸런스 게임서버 구조

그림 5에 나타난 바와 같이 제안한 Map 밸런스서버구조는 그림 4와 비교하여 원형부분에 도시된 Map 밸런스 서버와 이를 게임서버들과 연결하는 내부 네트워크가 추가된 구조를 가진다[4].

즉, 그림 4에 도시된 분산구조의 5단계의 동작을 기본적으로 같은 방식으로 수행하면서 Map 밸런스 서버와 게임배경을 처리하는 게임서버와의 동작을 추가한 구조이다.

그림 5에 나타나 있듯이 Map 밸런스서버는 다른 서버들과는 다르게 오직 게임서버에만 연결되어 동작을 수행한다. 물론 Map 밸런스서버가 게임서버와 연결되어 일련의 게임서버에서 수행되는 Map 상태를 변화시키면 게임 서버는 그 내용을 통신 서버에 전달하여 클라이언트가 이를 인지할 수 있게 해야 한다.

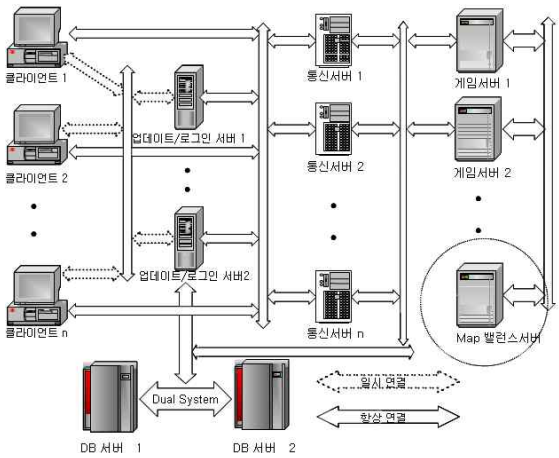

그림 5. Map 밸런스 서버가 추가된 분산형 서버의 구조
Fig. 5. Distributed Server Structure with Map balance Server

Ⅲ. Map 밸런스서버의 운영

Map 밸런스서버를 운용하기 위해서는 기존 분산 시스템과 마찬가지로 여러 서버가 배경 하나를 일정 크기로 나누어 서버의 성능에 따라 1개 혹은 그 이상의 조각 배경을 처리할 수 있게 해야 한다.

즉, 특정 게임 서버에 게임 배경을 전용으로 할당하는 것이 아니라 이미 할당된 배경을 게임 서버의 성능에 따라 동적으로 할당하는 역할이 필요하다

Map 밸런스서버는 기본적으로 Map을 처리하기 위한 연산을 어떻게 할 것인가가 운영의 목표가 된다. 즉, 하나의 서버가 관리하는 복수의 Map에서

Map 안의 사용자의 숫자에 따라 사용자가 존재하지 않아 유지시킬 필요가 없는 Map은 off 시키고 필요한 Map만 on 시켜서 불필요한 연산을 줄임으로써 서버의 효율을 높인다.

이를 위해 Map 밸런스서버는 유저가 없는 Map을 처리 중인 게임 서버에게 Map_off() 명령을 보내고 유저가 현재 off 중인 Map으로 입장하려고 하면 Map_on() 명령을 내려서 Map을 켠다.

Map_off 상태란 연산 동결 상태로 게임서버가 Map 유지를 위한 기본적인 연산은 하지 않지만 메모리에 상주하며 언제든 Map을 가동 시킬 준비가 된 상태를 말한다.

3-1 Map 밸런스 서버 역할

3-1-1 Map과 서버 간의 사용자 및 연산 대상 수치 비교

Map 밸런스서버는 실시간으로 각 Map들에 있는 사용자의 숫자를 지속적으로 검사한다. 이것을 통하여 유지 시켜야 할 Map과 off 시켜야 할 Map을 찾아낸다. 그리고 Map 밸런스서버의 총 인원에 따른 연산 량을 감시하고 그 데이터를 통하여 일정 시간마다 서버 간 부하를 비교하여 과부하 때문에 연산 장애가 나타날 수 있는 가능성 여부를 검사하고 부하가 가장 적게 걸려있는 이동시킬 서버를 찾아낸다.

3-1-2 Map의 on-off 명령

사용자가 없는 Map들을 off 상태로 만든다. 그리고 Off 상태인 Map에 사용자가 들어가면 해당 Map을 On 시킨다. 이것을 통하여 서버에서 불필요한 Map 처리로 인해 발생하는 부하를 줄일 수 있다.

3-1-3 Map의 서버 이동 명령

과부하가 걸릴 위험이 있는 서버에서 처리하고 있는 비교적 부하가 덜한 Map을 게임 서버들 중 비교적 부하가 덜 걸려있는 서버로 이동 시킨다. 이것을 통해 과부하가 걸린 서버는 부하가 큰 Map처리를 위해 조금이라도 많은 연산 량을 할당할 수 있고 모든 Map 서버군이 비교적 고른 연산 량을 보일 수 있게 된다.

3-2 게임 Map 서버들의 구조

하나의 Map 서버는 1개 이상의 분할된 복수 Map을 처리하도록 하며 이때 복수의 Map들은 각기 별도의 프로그램처럼 처리된다. 각 서버는 하드디스크 안에 모든 Map의 데이터를 가지고 있기 때문에 서버 간 Map 이동 명령이 Map 밸런스 서버를 통해서 내려지면 서버는 해당 Map을 하드디스크에서 로딩하여 연산을 시작한다.

3-3 Map 구조 및 관리

제안된 Map의 구조는 그림 6과 같다. 각 Map은 한 서버 안에 있더라도 각기 별도의 프로그램처럼 독립적으로 유지 연산된다. 그래서 각 Map들을 정지 또는 작동 하는 것을 선택적으로 하는 Map의 On-off 가 가능하다.

Map 안에 사용자가 1명도 없는 Map이 발생 할 경우 해당 Map은 Off 상태가 되며 1명의 사용자라도 Map 내부에 있으면 Map은 On 상태가 된다.

Map off 상태의 Map 데이터는 서버의 메모리에 올라가 있지만 몬스터의 이동과 같은 실질적인 Map의 기능을 하는 연산은 하지 않는다.

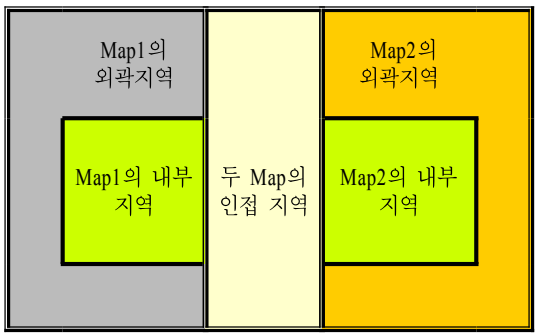

그림 6. 제안된 Map의 구조
Fig. 6. Proposed Map Structure

Map off 상태인 Map의 외곽지역에 사용자가 들어왔을 때에 해당 게임 서버는 Map 밸런스 서버로 부터 해당 Map에 사용자가 들어 왔다는 신호를 받게

되고 이 때 off 되어 ready 상태를 유지하고 있던 Map은 연산에 필요한 대상들을 연산하기 시작한다. 이때 연관된 해당 Map들은 내부와 외부로 나누어 관리를 한다. Map 내부는 해당 Map만 가지고 있는 고유영역이고 Map의 외곽은 인접 Map들과 공유를 한다.

즉, Map의 외곽지역이 On 상태로 켜져 있는 Map에서 Off 상태의 꺼져 있는 Map으로 사용자가 이동할 때, 2개의 Map이 서로 공유 하고 있는 외곽지역을 지나는 시간 동안 현재 On 상태의 맵에서 관리되는 사용자가 Off 상태인 Map의 내부지역으로 완전히 진입하기 이전에 Map 서버는 Off 상태인 Map을 On 상태로 변경 할 시간을 얻게 된다.

3-4 Map의 실시간 서버 간 이동방식

Map 밸런스 서버에서 처리하는 Map의 실시간 서버 이동방식은 아래와 같이 총 5단계로 이루어져 있다. 이를 그림 7에 순서도로 나타내었다.

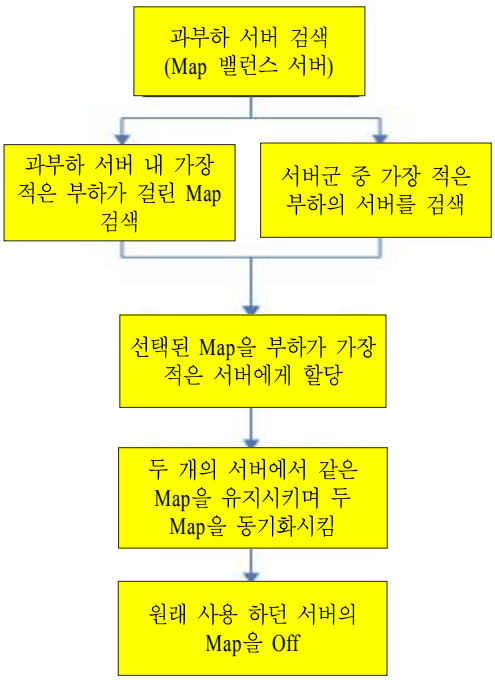

그림 7. Map의 서버 이동 방법에 대한 순서도
Fig. 7. Work Flow of Map Moving Method among Servers

1단계 Map 밸런스 서버는 주기적인 체크루틴을 통해 과부하가 걸릴 위험이 있는 서버를 찾아낸다. 동시에 과부하 서버에서 Map을 옮길 현재 가장 부하가 적게 걸린 서버를 찾는다.

2단계 과부하가 걸릴 위험이 있는 서버에서 가장 연산 량이 많은 Map과 가장 연산 량이 적은 Map을 찾는다.

3단계 가장 부하가 적은 Map을 선정하여 현재 가장 부하가 적게 걸린 이동 할 서버에 통보한다.

4단계 이동 할 서버에 Map 데이터 로딩부터 시작하여 일부 유저의 데이터를 동시 연산하는 것으로 시작하여 그 비율을 점차 높임과 동시에 은행에서 금융정보를 보관하기 위해 2개의 서버를 동시에 돌리는 것과 같이 2개의 같은 Map을 서로 다른 서버에서 돌리다 두 개의 서버 상에 있는 Map들의 연산을 완전히 동기화한다. 이때 모든 Map을 메모리에 실시간으로 올려놓는 부담을 줄이기 위해 Map이 이동할 서버는 해당서버의 하드디스크에 있는 Map 데이터를 로딩하는 방법을 사용할 수 있다.

5단계 이전에 운영하던 서버의 Map 관리 처리를 종료하고 Map을 이동한 새로운 서버에서에서 Map 관리를 수행한다. 이 때 Map 관리의 일관성을 유지하기 위하여 Map 관리를 이동한 상호 두 서버들은 완전한 동기화가 이루어지기 전에는 원래 Map을 관리한 서버에서 처리한 게임 데이터를 Map 밸런스 서버를 통하여 이동할 예정인 서버로 보내주어 게임 데이터를 동일하게 유지시켜야한다.

Ⅳ. 결 론

본 논문에서는 대규모 사용자를 수용하는 MMORPG에서 사용자의 선호도로 인한 불균등한 게임 배경 점유에 따른 각 서버 부하의 불균형을 Map 밸런스를 제어하는 방법을 통하여 해결하여 게임 사용자 처리를 위한 각 서버들의 효율을 높이고 지연없는 게임이 진행될 수 있도록 하는 분산 처리 모델을 제안하였다.

제안한 Map 밸런스서버 모델하에서는 개개의 게임 서버에서의 불필요한 연산을 사전에 제거하여 서버의 효율을 향상시키고, 이에 따라 생성되는 여분의 성능을 다른 서버들의 부하를 나누어 가질 수 있게 하여 전체 시스템의 효율을 높일 수 있다.

이는 게임 개발사가 서비스를 진행할 경우, 일부 서버에만 부하가 걸렸을 때 무작정 서버의 숫자를 늘리지 않고 적절한 부하 분산으로 모든 서버들이 고른

연산량을 수행할 수 있게 하여 전체 서버들의 효율을 최고로 끌어 올릴 수 있다.

물론, 대부분의 서버들이 고르게 과부하가 걸린 경우 불필요한 Map들의 서버 간 이동이 발생 할 수 있다. 이 때문에 최소 부하가 걸린 서버의 여유 부하량이 최고 부하량을 보인 Map의 최저 부하량을 넘어서는 것과 같은 전체적인 과부하 상황에서는 도리어 서버 과부하를 유발시킬 수 있다. 따라서 전체 서버 내의 부하량이 일정 범위 안에 있을 경우에는 Map 밸런스 서버는 Map On-Off 기능을 수행하지 않게 해당 범위를 테스트를 통하여 결정해야 한다.

이 문제를 해결하기 위하여 향후 서버 간 Map의 이동이 이루어지게 되는 기준이 되는 서버 간 부하 차이의 적정한 레벨의 선정에 관한 연구가 필요하다. 또한 2개의 서버에서 같은 Map을 동시에 수행할 때 야기되는 동기화 시의 오차 발생을 막기 위한 상호 데이터 확인 작업에 대한 통계적인 실증적 연구도 추가로 필요하다.

감사의 글

본 논문은 2011년 우석대학교 연구비와 지식경제부 지원 우석대학교 RIC의 지원에 의하여 이루어짐.

참 고 문 헌

[1] 제시카 멀리건, 브라짓 패트로브스키, 온라인게임기획, 이렇게 한다, *제우미디어*, 2006.

[2] 이남재, 곽훈성, 진화하는 온라인 롤플레잉 게임을 위한 분산형 게임 서버 모델, *한국게임학회 논문지 제 2 권 1 호*, pp 36~41, 2002.

[3]이남재 외 2, “온라인 게임에서의 효율적인 클라이언트 접속처리를 위한 비대칭 분산형 다중 서버 구조”, *정보처리학회 논문지 제12-B 권 제4 호*(통권 제100호), pp.27-38, 2005.

[4] 김법균, 안동언, 정성종, MMORPG에서의 부하 분산을 위한 가상 영역 정보 기반 동적 지역 분할, *한국정보처리학회 논문지 A,* VOL. 13 NO. 03, pp. 223 ~ 230, 2006.

[5] 하수철, 성해경, 게임 공간의 분류와 시나리오의 기간 및 공간 동기화 표현법, *한국정보처리학회 논문지 A,* VOL. 06 NO. 10, pp. 2630 ~ 2641, 1999

[6] 김태준, 서봉수, 종합 서비스 네트워크를 위한 예방적 경로 부하 밸런싱, *한국정보기술학회 논문지* VOL.9 NO. 2, pp.161, 2011

[7] T. Henderson: "Latency and Behaviour on a Multiplayer Game Server", *Proc. of 3rd Int'l. Workshop on Networked Group Communication (NGC2001),* LNCS2233, pp.1-13, 2001.

[8] 장봉석, 양기철, 배상현 광대역 이동통신망의 데이터 트래픽 공평서비스를 위한 상향링크 스케줄링 알고리즘, *한국정보기술학회 논문지 VOL.9 NO. 7,* pp.85, 2011

[9] G. Huang, M. Ye, and L. Cheng, "Modeling system performance in MMOG," *IEEE Communications Society Globecom Workshops 29,* pp.512-518, 2004.

[10] A. Abdelkhalek, A. Bilas, and A. Moshovos, "Behavior and performance of interactive multi-player game servers," *ISPASS01(2001), Vol.4, No.6*, pp.137-146, 2001.

김 순 곤(金舜坤)

1987년 2월 : 동국대학교 전산교육학과 (교육학석사)
1999년 8월 : 전북대학교 전자계산기공학과 (공학박사)
1987년 5월 ~ 1995년 3월 : 한국원자력연구소 선임연구원
1995년 3월 ~ 현재 : 중부대학교 컴퓨터학과 교수
관심분야 : 온라인게임, 게임데이터베이스, 게임보안, 분산형 온라인 게임서버구조

이 남 재(李南宰)

1988년 2월 : 전북대학교 컴퓨터공학과 (공학사)
1991년 2월 : 전북대학교 전산통계과 (이학석사)
2003년 8월 : 전북대학교 컴퓨터공학과 (공학박사)
1999년 9월~ 현재 : (주)엘맥스 대표이사
관심분야 : 분산형 온라인 게임서버구조

양 승 원(梁昇垣)

1995년 2월 : 전북대학교 전산통계학과(박사)
1998년 7월~1999년 12월 : ETRI 초빙연구원
2003년 9월~2004. 8월 : Univ. of Guelph in CANADA 객원교수
1994년 3월~ 현재 : 우석대학교 컴퓨터공학과 교수
관심분야 : 분산형 서버구조, 자연언어처리