무선 모바일 망에서 온라인 RPG의 설계 및 구현

오 선 진*

◈ 목 차 ◈

1. 서론

무선 인터넷의 급속한 발전과 더불어 모바일 컴퓨팅 환경에서 시간과 장소에 구애 없이 자유롭게 이동하면서 무선으로 인터넷에 접속하여 컴퓨팅 할 수 있는 다양한 응용들이 절실히 요구되고 있다. 최근 대표적인 모바일 장비로 휴대폰과 PDA가 널리 대중화를 이루고 있으며, 많은 마니아들을 중심으로 인터넷 정보 검색이나 동영상, MP3, 게임 등 여가활동에 중요한 자리를 차지하게 되었다. 그러나 이러한 대부분의 모바일 응용들을 살펴보면 간단한 오피스 관련 응용 프로그램으로부터 단순한 게임, 인터넷 접속과 이메일 검색과 관련된 응용들이 주류를 이루고 있으며, 대체로 너무 단순하고 일률적이어서 마니아들의 욕구를 충족시키기에 절대적으로 부족한 실정이다. 최근 무선 인터넷 사용자들 사이에서 가장 큰 관심의 대상은 단연 무선 온라인 게임이다. 하지만 이러한 온라인 게임들은 대부분 혼자서 즐기는 싱글 플레이가 주류를 이루고 있으며, 포켓 PC에서 온라인 상태로 여러 사람들이 함께 게임을 할 수 있는 본격 온라인 게임들의 개발이 매우 제한적이다. 이에 이러한 모바일 환경에 맞는 온라인 게임들이 본격 출현한다면 게임 시장이나 무선 컴퓨팅 시장의 새로운 도약을 기대할 수 있을 것이다.[1]

본 논문에서는 이러한 최근의 무선 인터넷을 중심으로 하는 정보 인프라와 더불어, 자유롭게 이동하면서 네트워크에 접속하여 컴퓨팅 할 수 있는 모바일 컴퓨팅 환경에서의 대표적인 무선 단말 장치인 WinCE 기반 PDA를 사용 하는 온라인 롤플레잉 게임(RPG) 컴포넌트를 설계하고 구현하였다. 일반적으로 롤플레잉 게임은 방대한 배경 이미지와 다양한 시나리오 전개 및 캐릭터들의 온라인 조작을 위한 컨트롤 등이 매우 복잡하므로 휴대용 단말 장치로써 제약이 많은 모바일 장치에서의 RPG의 구현은 그리 수월하지 않다. 특히 싱글플레이가 아닌 많은 사용자들이 동시에 온라인으로 접속해서 RPG 게임을 하는 멀티플레이 기능을 구현하는 것은 매우 어렵다. 본 논문에서 설계하고 구현한 온라인 RPG 컴포넌트는 이러한 문제들을 감안하여 모바일 단말기들 사이에 온라인 게임을 관리하고 주관하는 게임 서버를 두고, 온라인 상태에서 다수의 게임자가 무선 단말기인 PDA를 통해 실시간으로 게임을 할 수 있도록 설계하였으며, 이들 게임자들 사이에서 무선으로 게임 정보들이 실시간으로 송수신 되도록 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 PDA 기반 온라인 RPG 게임을 위한 시스템 모델을 살펴보고, 3장에서는 본 논문에서 설계하고 구현한 PDA 기반 온라인 롤플레잉 게임 컴포넌트의 구조와 알고리즘들을 자세히 서술하였으며, 4장에서는 구현된 모바일 환경에서의 온라인 롤플레잉 게임 컴포넌트들의 실행

* 세명대학교 정보통신학부 교수

결과 등을 소개하였다. 그리고 마지막으로 5장에서 향후 연구 과제와 함께 결론을 맺는다.

2. 시스템 모델

오늘날 모바일 컴퓨팅 환경에서의 모바일 단말장치의 프로세스 처리 능력은 데스크 탑 컴퓨터에 비해 성능이 많이 떨어지고, 저장 장치 용량 역시도 상대적으로 매우 작고 제한적이다. 또한 모바일 사용자들의 이동성을 위해 주요 전원 공급원으로 배터리를 사용하기 때문에 이동 단말장치의 전원에 대한 많은 제약이 따르며, 무선의 낮은 대역폭을 이용하여 데이터를 송수신해야 하므로 전송지연과 데이터에 대한 신뢰도가 떨어진다. 특히, RPG와 같은 방대한 량의 멀티미디어 데이터 처리를 요구하는 온라인 게임의 경우 이동 단말장치에 대한 제약 사항들의 영향이 더욱 심각하다. 따라서 이러한 단말장치에 대한 비효율성을 고려하여 무선 모바일 컴퓨팅 환경에서는 모듈에 대한 효율성을 높이기 위해 클라이언트-서버 구조를 이용한다. 즉 전체 시스템을 중앙에서 관리하기 위해 중앙에 하나의 게임서버를 두고, 여러 클라이언트들이 여기에 접속하여 게임과 통신을 하는 형태로 구성되며, 클라이언트-서버 간 데이터 전달을 위해 서버인 데스크 탑 컴퓨터와 모바일 단말 간 윈속 API를 사용하여 무선망을 통해 실시간으로 통신한다.[2, 3] (표 1)은 본 논문에서 구현한 모바일 환경에서의 온라인 롤플레잉 게임을 위한 주요 시스템 모델을 보여준다.

(표 1) 시스템 모델

온라인 롤플레잉 게임 시스템 모델		
Server 사양	Ultra 10 Sun Workstation	
운영체제	Solaris 9	
프로세서	Blade 2000	
Database	My - SQL	
Client 사양	HP iPaq rx3715 PDA	
운영체제	Microsoft Windows Mobile 2003-Second Edition Pocket PC Professional한글판	
프로세서	400MHz Samsung S3C2440	
디스플레이	컬러 수	16-bit 65,536 컬러
	터치스크린	지원
	해상도	240 X 320
	크기	3.5 inch (96mm)
메모리	SDRAM	128MB 플래쉬 RAM (iPAQ File Store 83MB 사용가능)

솔라리스 기반 Sun 워크스테이션 게임서버의 기능은 APM (Aphache-PHP-MySQL)을 사용, 인터넷을 통한 안정성, 확장성, 보안 및 관리 효율성이 뛰어난 통합 웹 서버 기능을 제공한다. 또한, 데이터베이스로는 MySQL을 사용하여 높은 안정성과 완벽한 웹 연동을 지원한다. 게임서버에서는 PDA의 작은 액정 화면을 통해 인터넷에 접속할 수 있는 전용 홈페이지를 구축하기 위해 PDA의 환경에 맞는 240 × 320 크기의 화면으로 구성하였다. 게임 서버에 회원 가입을 통해 클라이언트는 게임 어플리케이션을 다운로드 받아서 게임 할 수 있으며, 게임과 관련 변경된 정보들은 게임 서버의 데이터베이스에서 저장 관리하게 된다.[4, 5]

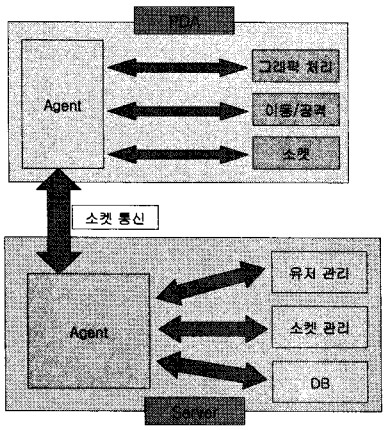
한편, 클라이언트 환경은 Windows 환경을 모바일 장치로 확장하여 시간 활용도를 극대화하는 PDA 포켓 PC를 사용하였고, Wi-Fi를 통해 무선 환경에 쉽게 접속할 수 있도록 설계하였으며, 클라이언트의 PDA에서는 게임 어플리케이션과 Pocket Internet Explorer를 통해 무선으로 인터넷 게임 홈페이지에 접속할 수 있도록 하였다.[3, 6]

3. RPG 컴포넌트 설계

게임 서버의 주요 구성요소는 유저관리 컴포넌트, 소켓 컴포넌트, DataBase 컴포넌트, Agent 컴포넌트 등으로 구성된다. 유저관리 컴포넌트에서는 게임 사용자 계정에 대한 관리를 하며, 각 게임 사용자의 캐릭터 이름과 아이디 등은 실제 데이터베이스에 저장되지만 그것들의 관리는 이곳에서 한다. 또 게임 사용자가 동시에 접속 했을 경우에 대한 처리도 이곳에서 이루어진다. 소켓 컴포넌트는 클라이언트와 통신하는 곳으로 실제로 클라이언트와 패킷을 서로 주고받는 부분을 담당한다. DataBase 컴포넌트는 온라인 게임이 운영되는데 필요한 주요 정보들을 저장하고 관리하는 부분으로 여기에 모든 게임 데이터들이 담겨져 있으며, 이

들에 대한 정보는 Agent 컴포넌트에 의해 관리된다. Agent 컴포넌트는 다른 세 컴포넌트를 관리하는 부분으로 소켓에서 메시지를 분석하는 부분, 클라이언트와 통신이 원활하게 이루어지는지 모니터링하는 부분, 그리고 접속을 강제로 종료하는 등의 작업을 수행하게 된다. 또한, 해당 작업들에 의해 DataBase에 수정, 추가 및 삭제가 이루어질 경우 Agent에서 이 작업을 수행한다.

클라이언트의 주요 컴포넌트는 크게 그래픽 처리 컴포넌트, 소켓 처리 컴포넌트, 캐릭터들의 이동/공격 그리고 이들을 제어하고 관리하는 Agent 컴포넌트 등으로 구성된다. 그래픽 처리 컴포넌트는 캐릭터나 몬스터 그리고 공격 등의 그래픽 부분을 담당하는 곳으로 여기엔 스프라이트 및 각 그림을 프레임 별로 순차적으로 보여 주어서 연속된 동작으로 보이게 하는 부분 등이 들어있다. 소켓 처리 컴포넌트는 게임 서버로 패킷을 보내거나 또는 패킷을 받는 통신 작업을 수행하는 부분으로 이루어져 있으며, 수신 패킷에 대한 정보는 Agent 컴포넌트에서 해독하여 해당 작업에 넘겨주게 된다. 이동/공격 컴포넌트는 캐릭터나 몬스터의 이동 방향을 결정하고, 이동 위치의 변화와 공격 등의 컨트롤을 처리하는 부분이다.



(그림 1) 게임 전체 시스템 구성도

Agent 컴포넌트는 이들 컴포넌트들을 관리하여 상호 유기적으로 동작하게 조정하는 부분으로, 특히 게임 서버로 부터 받은 수신 메시지를 분석하고 해독하여 이에 맞는 부분에 해당 작업을 보내는 일을 수행하는 곳으로 클라이언트의 총체적인 관리를 담당하고 있다. (그림 1)은 본 논문에서 구현한 무선 모바일 환경에서의 온라인 롤플레잉 게임의 전체 시스템 구성도를 보여준다.

데이터베이스는 유저정보 테이블, 몬스터 테이블, 아이템 테이블, 맵 테이블, 길드 테이블, NPC 테이블로 구성되어 있다. 유저정보 테이블은 유저에 대한 정보와 캐릭터에 대한 정보를 저장 관리하며 캐릭터의 종류, 캐릭터가 갖고 있는 무기, 캐릭터가 소속된 길드, 친구, 파티 형성, 습득한 아이템(e-money 포함), 그리고 이 캐릭터의 능력 등이 저장한다. 몬스터 테이블은 몬스터의 현 위치, 생성 위치, 사망 시 드롭 할 아이템과 e-money, 몬스터의 능력 등을 저장한다. 아이템 테이블에는 아이템에 대한 능력과 가격 등이 저장되고, 맵 테이블은 마을 및 사냥터에 대한 정보를 갖게 된다. 여기에는 어느 위치에 몹이 있고 NPC들이 있는지에 대한 정보가 저장되어 있다. 길드 테이블은 길드장, 길드부원들 및 길드 레벨 등의 정보가 저장되고, NPC 테이블은 NPC가 판매하는 아이템 및 가격, NPC의 위치 정보가 담겨져 있다.[2]

캐릭터나 몬스터의 움직임은 연속된 그림을 바탕으로 이루어진다. 캐릭터는 한 방향 당 정지 화면을 포함해 최대 4 프레임까지 그림을 차례로 출력하였을 때 연속된 움직임으로 보일 수 있도록 구현하였으며, 캐릭터나 몬스터는 모두 각 8 개의 방향으로 이동할 수 있도록 하여 해당 방향마다 각각의 모양을 가진 프레임을 작성하였다. 캐릭터나 몬스터의 진행방향 결정 알고리즘은 게임 사용자가 PDA 화면을 터치 했을 때 터치한 좌표와 현재의 좌표를 비교하여 각각의 방향을 결정하게 된다. 이러한 캐릭터 이동 결정은 단순 if - else 문을 사용하여 구현하였다. (그림 2)는 캐릭터나 몬스터의 진행 방향을 결정하는 진행 방향 결정 알고리즘을 보여준다.

```
// Get Current and Touch Location
if(touch.y < cur_loc.y && touch.x == cur_loc.x)
      set direction = North;
else if(touch.y > cur_loc.y && touch.x == cur_loc.x)
      set direction = South;
else if(touch.y == cur_loc.y && touch.x < cur_loc.x)
      set direction = West;
else if(touch.y == cur_loc.y && touch.x > cur_loc.x)
```

```
        set direction = East;
else if(touch.y > cur_loc.y && touch.x > cur_loc.x)
        set direction = SouthEast;
else if(touch.y > cur_loc.y && touch.x < cur_loc.x)
        set direction = SouthWest;
else if(touch.y < cur_loc.y && touch.x > cur_loc.x)
        set direction = NorthEasr;
else if(touch.y < cur_loc.y && touch.x < cur_loc.x)
        set direction = NorthWest;
```

(그림 2) 캐릭터와 몬스터 진행방향 결정 알고리즘

```
// 몬스터 공격결정
RECT Monster; // 몬스터 사각형
struct CharPos; // 캐릭터 위치
struct ClickPos; // 클릭한 위치
StylusDown();   // 스크린 터치
SetAlive();     // 해당 몬스터가 살아 있는지 확인

if (ClickPos.x =< Monster.Left && ClickPos.x =>
   Monster.Right && ClickPos.y =< Monster.Top &&
   ClickPos.x => Monster.Bottom)
{
     if(SetAlive())
          monster_attack();
}
else
     move_character();
```

(그림 3) 몬스터 공격 결정 알고리즘

몬스터의 공격은 게임 사용자가 몬스터를 클릭했을 때 공격이 진행되며 공격을 위해 몬스터를 클릭하면, 이때 클릭한 위치에 몬스터가 존재하면 공격이 이루어지게 된다. 몬스터를 공격하기 위해 그 몬스터의 크기를 계산하는 방법은 몬스터를 사각형 모양의 상자에 포함시켜서 몬스터의 존재를 결정하며, 게임 사용자가 그 사각 영역을 클릭하면 공격을 시작하고 그렇지 않으면 공격을 하지 않는 방식을 사용하였다. 또한 몬스터가 존재하지 않는 장소에 대한 클릭은 게임 사용자가 캐릭터를 해당 위치로 이동시키기 위한 동작으로 보고 이 함수가 동작된다. 몬스터에 대한 공격은 그 몬스터가 사망 할 때까지 계속된다. (그림 3)은 몬스터 공격 결정 알고리즘을 보여준다.

```
DWORD WINAPI SocketReceive()
{
    FlagReceive = 1;
    ReceiveTime 시작;  // 0.1초 후에 1로 설정
    ReceiveMessage(Packet)

    if(exam_Packet)
    {
        monster[i].x;  // 몬스터들의 위치
        monster[i].y;
        Charinfo();   // 캐릭터의 상태 정보
    }
    else // 채팅 메시지
    {
        print (message);  // 채팅 메시지 출력
    }
}


DWORD WINAPI SocketSend()
{
    FlagSend = 0;
    SendTime 시작; // 0.1초 후에 1

    SendPacket(Packet); // 패킷 송신
}
```

(그림 4) 클라이언트 소켓 송수신 알고리즘

(그림 4)는 클라이언트에서 패킷을 송신 또는 수신하고 분석하는 클라이언트의 소켓 송수신 스레드 알고리즘을 보여준다. 소켓 수신 스레드가 실행을 시작하면 우선 수신 플래그가 설정되고, 시간을 측정하여 0.1초가 지나면 ReceiveTime이 1로 설정된다. 클라이언트 소켓 송수신 알고리즘을 이와 같이 구성한 이유는 Windows CE 기반인 PDA를 사용하는 클라이언트에서 사용하는 소켓이 WinCE 환경에서의 API 특성 상 논 블로킹 모드 소켓을 구성할 수 없기 때문이다. 따라서 논 블로킹 모드를 클라이언트에서 인위적으로 구현하기 위해 스레드를 이용하여 강제로 소켓을 닫고 다시 수신을 개시하는 방법을 사용하여 논 블로킹 모드를 구현한다.[7, 8] 이를 ClientAgent()에서 제어하며 수신이 제대로 이루어 지지 않으면 ClientAgent()에서 재전송을 요청한다. 클라이언트의 소켓 송신부는 위의 소켓 수신부 보다는 비교적 간단한 구조를 갖고 있지만

기본 골격은 거의 동일하다. 이 클라이언트 송신 부분은 단지 메시지를 송신하는 것이지만 논 블로킹 모드의 소켓 구성이 역시 되지 않기 때문에 논 블로킹 모드와 같은 동작을 하게 구현하기 위해서 시간과 플래그를 두었으며, 이를 이용해서 강제로 종료할 수 있도록 설계 하였다. 이 동작 역시 ClientAgent()에서 관리를 한다.

```
int FlagReceive = 0;
int FlagSend = 0;
int SendTime = 0;
int ReceiveTime = 0;

DWORD WINAPI ClientAgent()
{
  if(FlagReceive && ReceiveTime) {
    SocketReceive() 종료;
    SocketSend(message);  // 수신 실패 메시지
  }
  if(FlagSend && SendTime) {
    SocketSend() 종료;
    SocketSend(message);  // 송신 실패 메시지
    SocketSend 스레드 다시 시작;
  }
}
```

(그림 5) 클라이언트 소켓 스레드 알고리즘

(그림 5)는 본 논문에서 구현한 ClientAgent() 클라이언트 소켓 스레드 알고리즘을 보여준다. 위에서 설명한 대로 통신을 위한 소켓이 블로킹 모드이기 때문에 발생하는 문제를 해결하기 위해 송·수신부를 스레드로 나누고, 스레드가 실행되고 일정 시간이 경과해도 다음 동작으로 진행되지 않으면 강제 종료하도록 구현하였다.[8]

```
void *SendSocket()  // 소켓송신 스레드
{
      Sendto(Message);
}

void *ReceiveSocket()  // 소켓 수신 스레드
{
     recvfrom(Message);
}
```

(그림 6) 서버 소켓 송수신 알고리즘

게임 서버 측에서의 소켓은 PDA에서와 같은 제약이 없으므로 논 블로킹 모드로 구성되어 있으며 각각을 스레드로 나누어서 구현하여 놓았다. (그림 6)의 상단의 소스는 게임 서버에서 클라이언트로 소켓을 보내는 스레드이다. (그림 6)의 하단의 ReceiveSocket() 함수는 서버에서 클라이언트로부터 소켓 메시지를 받는 스레드 부분이다.

```
iint Analysis(Message);  // 패킷 분석 함수

void *ServerAgent()
{
  switch(Analysis(Message)){
  case DB 관련:
      ProcessingDB(); // DB 관련 작업
   break;
  case 채팅 메시지:
     Sendto(Messge); // 해당 목적지로 송신
  break;
  case 클라이언트 수신 실패:
     // 수신 실패한 클라이언트로 다시 송신
     Sendto(Message);
  break;
  case 클라이언트 수신 실패:
     // 송신 실패한 클라이언트로부터 다시수신
     recvfrom(Message);
  break;
  }
}
```

(그림 7) 서버 소켓 스레드 알고리즘

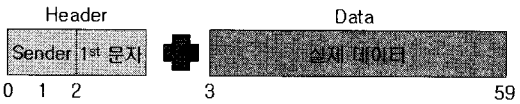
클라이언트에서의 소켓 알고리즘 구현에서 가장 어려운 문제는 WM_PAINT 메시지가 수신 될 때마다 클라이언트의 액정 화면에서 화면 깜빡임을 일으키는 윈도우즈 개발 환경에서의 repaint 문제를 해결하는 것이다. 이 문제를 해결하는 일반적인 방법으로는 가상 윈도우를 사용하여 repaint 문제를 해결한다. 즉, 응용 프로그램에서 사용한 DC와 호환이 가능한 메모리 DC를 만든 후 클라이언트 영역의 모든 출력 내용을 메모리 DC에 동일하게 유지하였다가 WM_PAINT 메시지가 수신될 때마다 이 메모리 DC로 복구하는 방법이다. 그러나 무선 모바일 온라인 컴퓨팅 환경에서는 모바일 단말 장치의 저장장치 제약이나 처리 속도의 제한 등으로 인해 충분히 만족할 만한 결과를 얻기

어렵고 절대적으로 메모리 제약을 받는다. 따라서 본 논문에서는 이 문제를 해결하기 위해 RPG의 배경이 되는 대용량을 차지하는 배경 이미지 데이터와 캐릭터를 분리시킬 수 있는 더블 버퍼링 방법을 사용하여 배경 이미지를 그대로 유지한 채 캐릭터의 움직임을 제어함으로써 repaint 문제를 해결한다.

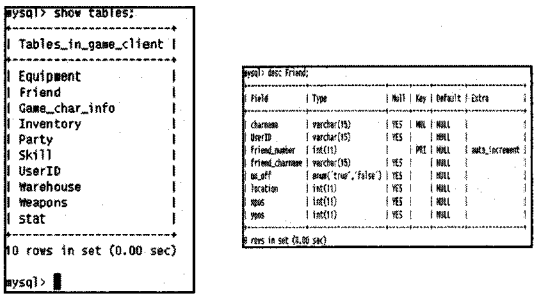
(그림 7)은 서버에서의 소켓 스레드 알고리즘을 보여 준다. 여기서 ServerAgent()는 클라이언트로부터 받은 메시지를 분석하고 그에 따른 작업을 지시하는 역할을 하는 부분이다. 클라이언트로부터 수신한 패킷을 분석하는 Analysis() 함수의 결과에 따라 Database와 관련된 작업을 할 것인지, 아니면 클라이언트로부터의 송·수신이 실패했을 경우 이를 다시 주고받기 위해 다시 한 번 소켓을 송·수신 할 것인지를 결정한다. 게임 서버는 하나의 클라이언트만 관리하는 것이 아니기 때문에 클라이언트 쪽에서 실패 메시지가 온다고 하더라도 소켓 스레드를 종료해서는 안된다. 따라서 다시 한 번 메시지를 보내거나 받는 임시 함수를 따로 만들어 이를 실행해야 한다.[8, 9]

4. 구현 결과

이 장에서는 본 논문에서 제안한 무선 모바일 컴퓨팅 환경에서의 온라인 롤플레잉 게임을 설계하고 구현한 결과를 보여 준다. (그림 8)은 본 논문에서 데이터 전송을 위해 사용한 패킷의 구조를 보여준다. 패킷의 총 길이는 전송 속도 및 지연을 효율적으로 관리하기 위해서 60 바이트로 제한하였다. 패킷은 처리를 위해 검사를 받게 되는데, 이 패킷이 일반 작업 처리를 위한 데이터인지 아니면 채팅을 위한 데이터인지를 판별 한 후 해당 작업을 하게 된다. 이를 판단하는 방법은 패킷의 Sender 다음 1 바이트를 두어 이를 판별하게 구현하였다. 처음 문자가 만약 '@'나 '*' 등의 특수 문자로 시작되면 채팅을 위한 데이터로 간주하여 처리하게 된다.

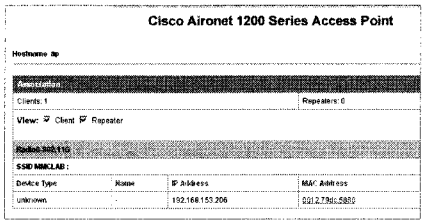


(그림 8) 패킷 구조



(그림 9) 서버 데이터베이스의 구조

(그림 9)는 게임 서버의 데이터베이스에 저장된 캐릭터에 대한 정보 테이블을 보여준다. 이 테이블에는 장비, 친구, 캐릭터 정보, 인벤토리 등 캐릭터에 필요한 정보들이 저장 관리하게 된다. 친구 테이블은 캐릭터의 친구와 연결되며, 친구의 캐릭터 이름, ID, 접속 유무, 위치 등의 정보가 저장되어 있어서 이를 이용하여 친구의 상태를 알 수 있다.



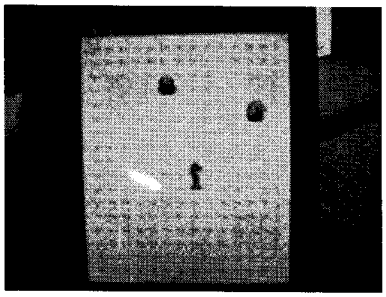
(그림 10) 서버에 클라이언트의 접속결과

(그림 11) 서버홈페이지 접속화면

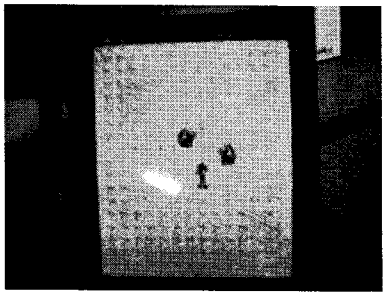
(그림 10)은 게임 서버에 장착된 시스코의 AP 장비를 이용한 무선 네트워크를 구축하여 클라이언트들이 게임 서버에 무선으로 접속한 결과를 보여준다. 테스트 상에서는 한 개의 PDA를 접속하여 접속 유무를 확인하였고, 시스코 AP는 Cisco Aironet 1200 Series를 사용하였다.

(그림 11)은 게임 서버의 홈페이지에 클라이언트가 PDA를 통해 접속한 화면을 보여준다. 이렇게 접속하여 클라이언트들이 회원가입을 할 수 있고 게임의 응용 프로그램을 다운로드 받아서 클라이언트의 무선 모바일 단말 장치인 PDA에 설치 할 수도 있으며, 다른 로그인 사용자들과 온라인 상태로 멀티플레이 게임을 할 수도 있다. 또한 접속한 다른 사용자들과 실시간으로 채팅을 할 수 있도록 기능을 제공한다.

(그림 12)와 (그림 13)은 본 논문에서 구현한 온라인 롤플레잉 게임을 실행한 결과 화면이다. (그림 12)는 몬스터와 캐릭터가 맵 상에 출현한 화면 그림으로 이때 맵은 Jonathan S. Harbour의 Pocket PC Game Programming의 맵 타일을 사용하였다.[4]



(그림 12) 롤플레잉 게임 실행화면 1



(그림 13) 롤플레잉 게임 실행화면 2

(그림 13)은 몬스터가 캐릭터를 향해 공격을 하는 실행 화면 그림이다. 화면에서 빨간 점으로 표시된 것이 몬스터가 캐릭터를 향해 공격을 시도하는 것이며, 이것에 대한 방향은 랜덤하게 움직이고, 시야에 캐릭터가 들어오면 바로 공격을 하게 된다. 이때 시도된 공격의 탄환은 한 번 진행되면 방향을 바꾸지 않고 계속 진행되도록 구현하였다.

5. 결론

최근 모바일 컴퓨팅 기술과 무선 통신 기술의 급속한 발전과 더불어 모바일 장비 기술과 응용도 급속히 발전 보급되고 있다. 그러나 무선 모바일 장비를 위한 응용이나 게임들은 장비나 환경의 제약으로 인해 매우 제한적이다. 특히 RPG와 같이 방대한 양의 이미지 데이터와 다양한 시나리오 그리고 복잡한 컨트롤을 가진 온라인 게임의 경우 무선 모바일 단말이 갖는 저장장치 크기의 제약이나 처리기의 성능 한계, 배터리 기반의 전원 공급의 제약과 무선을 통한 통신으로 인한 대역과 전송 속도 제약 등이 따르는 모바일 장치에서의 구현이 매우 난해하다.

본 논문에서는 이러한 문제들을 해결할 수 있는 모바일 환경에서의 온라인 롤플레잉 게임 컴포넌트를 설계하고 구현하였다. 효율적인 온라인 서비스를 위해 클라이언트-서버 구조 기반의 게임 서버를 두고, 회원관리나 게임 관리를 담당하게 하였고, 클라이언트들은 모바일 단말인 PDA를 이용하여 무선으로 서버의 AP를 통해 접속하여 게임이나 기타 서비스를 받을 수 있도록 컴포넌트 기반으로 설계하였다. 통신은 무선 링크를 통해 TCP 프로토콜 기반 1 대 1 통신에서부터 1 대 다 통신까지 되도록 설계하여 실시간으로 동시에 여러 클라이언트들이 동시에 게임을 할 수 있도록 구현하였다. 무선 통신에서 클라이언트 부분은 소켓의 제한으로 인해 블로킹 모드로 구현하였으며, 이에 따른 문제점을 극복하기 위하여 송·수신 소켓 프로세스를 스레드로 나누고 시간에 따라 강제로 종료하고 재시작하게 하는 방법으로 이 문제를 해결하였다. 또한 윈도우 개발 환경에서의 repaint 문제를 해결하기 위해 배경 이미지와 캐릭터를 분리한 더블 버퍼

링 기법을 통해 화면 깜빡임 등 repaint 문제를 해결하였다. 그리고 PDA에 비해 제약이 없는 게임 서버는 논 블로킹 모드로 구현하였다. 구현한 온라인 롤플레잉 게임을 실행한 결과 게임 서버를 중심으로 원활한 온라인 기능이 수행됨을 확인할 수 있었고, 싱글플레이 뿐만 아니라 멀티플레이도 무난히 수행할 수 있음을 확인할 수 있었다. 향후 연구과제로는 고정된 인프라 구조를 갖지 않는 모바일 애드 혹 네트워크 환경에서 서버를 두지 않고 수행할 수 있는 무선 온라인 게임의 설계에 관한 것이다.

참고문헌

[1] John W. Fisher II, "Methods and Considerations in Online Game Design," Master's Thesis at Michigan State University, East Lansing, MI USA, 2003.

[2] Jim Adams, Programming Role Playing Games, Course Technology PTR, pp. 974, 2002.

[3] Anthony Jones, Jim Ohlund : Network Programming For Microsoft Windows, Microsoft Press, 2003.

[4] Jonathan S. Harbour, Pocket PC Game Programming: Using The Windows CE Game API, Microsoft Press, 2003.

[5] Charles Petzold, Programming Windows, 5th ed. Microsoft Press, pp. 1536, 2005.

[6] 오선진, 임베디드시스템 소프트웨어 개발방법론, 한올출판사, pp. 461, 2007.

[7] 백창우, TCP/IP 소켓 프로그래밍, 한빛미디어, pp. 744, 2005.

[8] E. L. Lamie, 실시간 임베디드 멀티스레딩, 에이콘 출판, 2005.

[9] Alex J. Champandard, 인공지능 게임 프로그래밍 실전 가이드, 에이콘 출판, 2004.

◐ 저 자 소 개 ◑

오 선 진

1981년 한양대학교 공과대학(공학사)
1987년 미국 Wayne State University 컴퓨터과학과(이학사)
1989년 미국 University of Detroit 컴퓨터과학과(이학석사)
1993년 미국 Oklahoma State University 컴퓨터과학과(이학박사 수료)
1999년 대구 가톨릭대학교 전자계산학과(이학박사)
1994년~2000년 선린대학교 컴퓨터정보학과 교수
2000년~현재 세명대학교 정보통신학부 교수
관심분야 : 모바일 멀티미디어, 모바일 컴퓨팅, 무선 인터넷 등
E-mail : sjoh@semyung.ac.kr