제2장 국내외 게임기술 동향

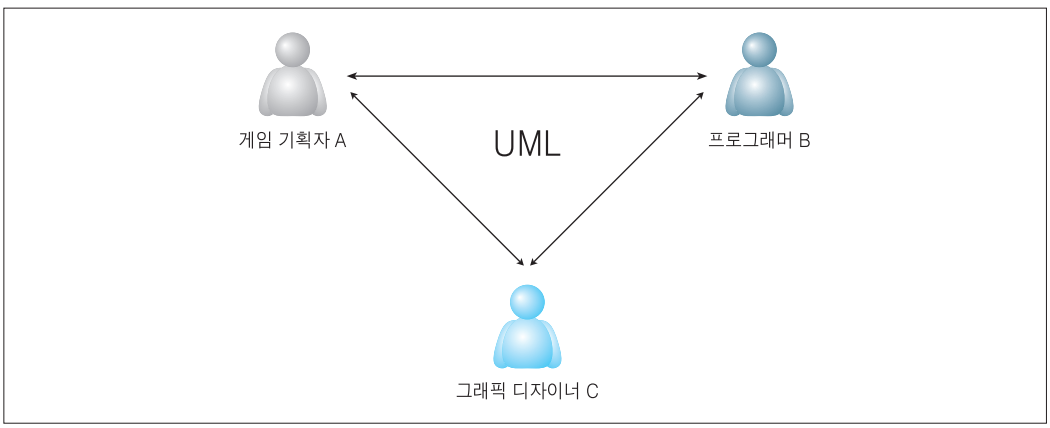
제1절 UML를 활용한 게임 디자인

1. 게임 기획하기

10여 년 동안 게임개발 기술은 급속도로 발전해 왔다. 외국과의 게임개발의 기술 차이도 많이 줄어들었고, 오히려 인터넷을 기반으로 한 온라인게임에서는 상당한 수준으로 발전하였다. 그러나, 게임기획 부분 즉, 게임을 디자인 하는 부분에서는 작업 방식이나 커뮤니케이션 방법이 다른 파트의 발전에 비해 느리게 진행되어 오고 있다. 또한, 게임 제작단계에서 기획의 역할이 갈수록 강조되고 있으나, 어떠한 방식으로 진행을 해야 하는지에 대한 연구는 부족한 실정이다.

UML(Unified Modeling Language)은 1990년대 후반에 표준 모델링 언어로 채택되어 현재까지 사용되고 있다. 일반적으로 UML은 소프트웨어 개발에서 객체지향 분석 설계 분야에 사용한다. 게임 또한, 소프트웨어이며,

〈그림 5-2-1-01〉 UML의 커뮤니케이션



게임개발에 UML의 사용은 개발자에게 편리함을 제공한다. 대부분의 경우, 프로그래머들이 많이 사용을 하게 되는데, 그렇다고 해서 프로그래머의 입장에서 UML을 사용해야만 하는 것은 아니다. UML은 각 작업자의 입장에서 개발해야 하는 소프트웨어의 비전을 제시하는 것이기 때문에 누구나 사용할 수 있으며, 배우기도 쉽다.

〈그림 5-2-1-01〉은 게임 기획자의 "A"라는 생각과 프로그래머의 "B"라는 생각, 그래픽 디자이너의 "C"라는 생각을 UML로 표현해서 의사소통을 한다라는 것을 표현하고 있다.

2. UML의 필요성

UML은 '생각의 정리', '논리적인 생각', '표준적인 표현방식'이라는 장점 때문에 게임 디자인 시 사용해 볼만한 가치가 있다. 인간은 글 보다는 그림 형태의 표현을 더 잘 이해한다. 디자이너는 자신이 생각하는 전투시스템을 UML 표준에 맞게 계속 그려봄으로써 생각을 정리할 수 있고, 잘못된 부분을 쉽게 찾을 수 있다. 그리고 UML은 표준화된 표현 방식이기 때문에 UML을 알고 있다면, 누구나 쉽게 이해할 수 있다.

프리 프로덕션 단계의 작업에서 무수히 많은 의견을 주고 받고, 정리했음에도 불구하고 얼마 뒤 작업을 진행하는 과정에서 "서로 바라보는 시각이 달랐구나!" 하고 느낄 때가 자주 있다. 이는 각자 맡은 업무가 다르기 때문에 당연한 일이다. 심지어는 같은 기획 파트 내에서도 다른 생각을 가지고 있었던 적이 많이 있을 것이다. 팀원이 3명 이상 넘어가면 서로의 생각을 조율한다는 것이 쉬운 작업이 아니라는 것을 우리는 알고 있다.

3. UML(Unified Modeling Language)

UML은 13개의 다이어그램으로 구성되어 있으며, 이 다이어그램들은 약속된 기호(Notation)

〈표 5-2-1-01〉 UML의 13개의 다이어그램 (UML 버전 2.0 기준)

구분	
구조 다이어그램	클래스 다이어그램 컴포넌트 다이어그램 복합 구조 다이어그램 배포 다이어그램 객체 다이어그램 패키지 다이어그램
행동 다이어그램	활동 다이어그램 상태 다이어그램 유스 케이스 다이어그램
교류 다이어그램	통신 다이어그램 교류 개요 다이어그램 시퀀스 다이어그램 타이밍 다이어그램

〈표 5-2-1-02〉 UML 도구

제품	웹사이트
Rational Rose	http://www-306.ibm.com/software/awdtools/developer/rose/index.html
Star UML	http://staruml.sourceforge.net/
Visual UML	http://www.visualobjectmodelers.com/
Poseidon for UML	http://www.gentleware.com/
Visio	http://office.microsoft.com/visio
Visual Paradigm for UML	http://www.visual-paradigm.com

로 표현하게 된다.

4. 게임 디자인에 UML 적용하기

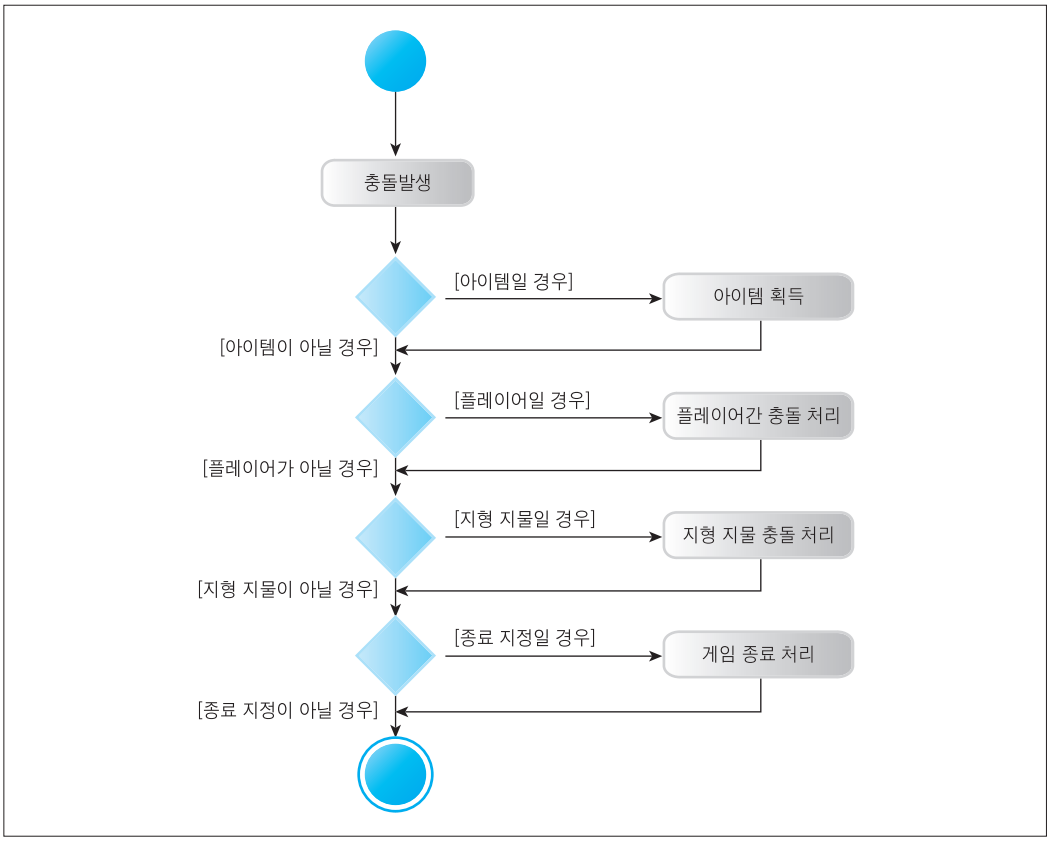
여기서 설명하는 내용은 UML 다이어그램의 전체 내용 중에서 게임 기획자가 쉽게 사용해 볼 수 있는 몇 가지를 적용해서 설명할 것이다. 표현하는 방법은 회사의 상황과 팀원 그리고 개발 방법에 따라서 차이가 있을 수 있다.

첫번째로 활동 다이어그램을 설명할 것이다. 활동 다이어그램은 일을 처리하는 과정에서 수행되는 동안 일어나는 일들을 간단하게 표현하기 위해 고안된 것이다.

검은 원의 원은 시작을 의미하고, 검은 원의 테두리가 있는 것은 종료를 의미한다. 타원의 둥근 형태는 활동을 의미하고 마름모 사각형은 조건에 따른 분기를 의미한다.

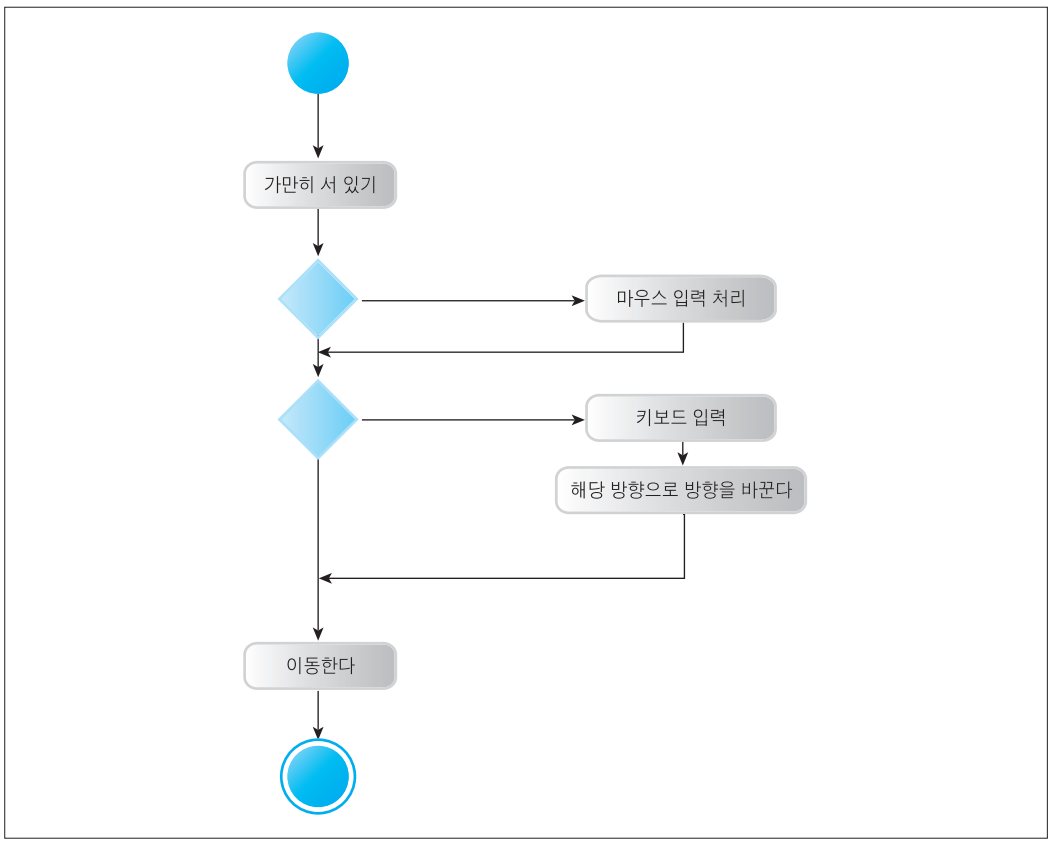
아래의 그림은 카트라이더에서 플레이어의 자동차가 충돌이 발생하였을 경우에 대한 상황을 활동 다이어그램으로 표현해 본 것이다. 자동차가 달리는 중 충돌이 발생하게 되면 무엇과 충돌이 발생하였는지 찾게 될 것이다. 충돌할 수 있는 경우의 수를 전부 검색하고 찾는 과정에서 위의 그림과 같이 표현될 수 있을 것이

〈그림 5-2-1-02〉 사례1 – 활동 다이어그램



다. 〈그림 5-2-1-03〉의 활동 다이어그램은 일반적인 MMORPG의 이동 방법에 대해 아주 간략하게 다이어그램으로 표현하였다. 리니지2나 월드오브워크래프트도 이와 같은 방식으로 캐릭터 이동이 구현되어 있다. 그렇지만 리니지2와 월드오브워크래프트의 캐릭터 이동 방법은 차이가 있다. 그래서 다이어그램을 일반화하여 그린 그림은 둘 다 수정해서 그려야 한다. 둘의 차이는 마우스 입력에 의한 이동방법에 관심을 두는 것이 아니라 키보드 입력에 따른 캐릭터 이동에 관심을 두고 있다. 리니지2 같은 경우, 대부분의 캐릭터 이동을 마우스로만 하도록 되어 있다. 그렇지만 키보드 입력으로 캐릭터 이동이 안 되는 것은 아니다. 자연스럽지는 못하지만, 키보드 입력이 들어오게 되면 약간의 시간이 지난 뒤 캐릭터는 이동을 하게 된다. 월드오브워크래프트의 캐릭터 이동 방법은 FPS 게임과 유사한 방식인 키보드와 마우스를 사용하여 캐릭터가 이동하도록 구현되어 있다. 키보드 입력이 들어오면 리니지2와 같은 방식이 아닌 즉시 반응을 한다. 만약, 앞으로 가는 키 버튼을 누르게 되면, 즉시 앞으로 이동하게 된다. 만약, 왼쪽으로 방향을 바꾸는 입력 버튼을 입력하게 되면 우선 캐릭터는 방

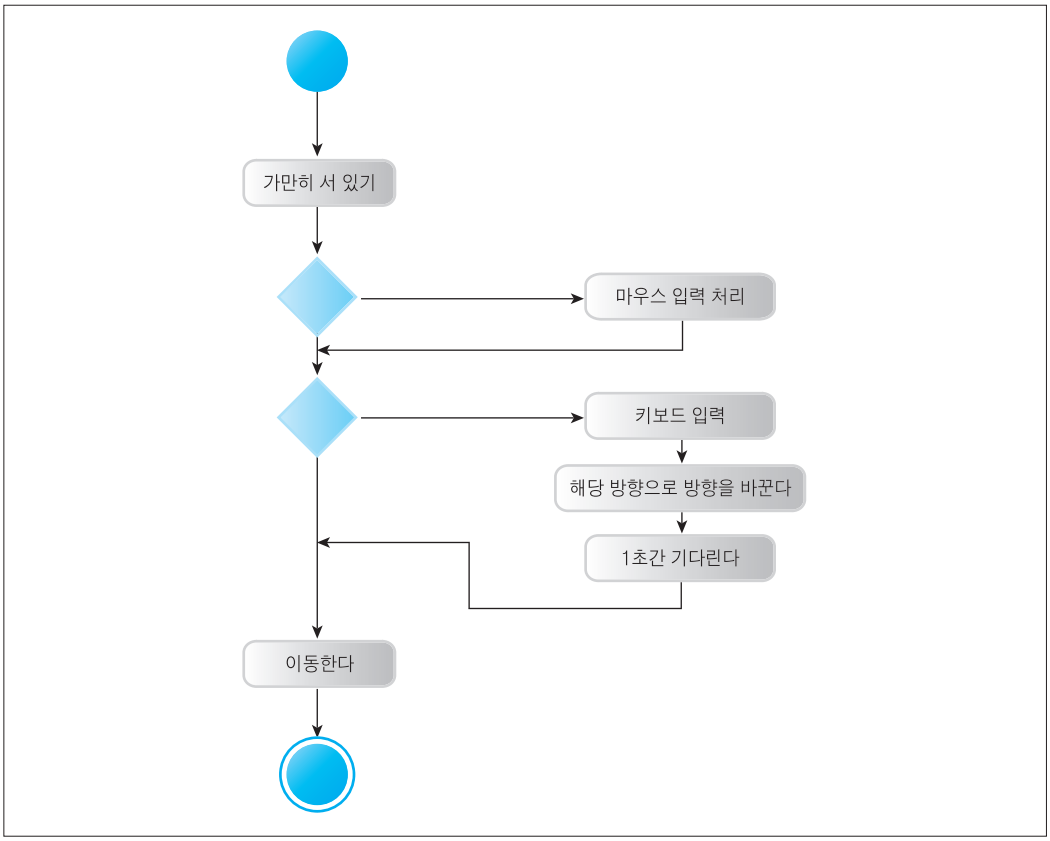
〈그림 5-2-1-03〉 사례2 – 활동 다이어그램 : MMORPG에서 일반적인 캐릭터 이동



향을 바꾸게 되는 몸의 상체 부분이 방향을 바꾸고 난 다음에 캐릭터의 하체 부분이 연이어서 방향을 바꾸게 되는 것이다. 물론, 게임을 기획하는 기획자가 이러한 차이를 알고 있으면 자신의 게임에 어떤 방식이 좋을지 벤치마킹을 하게 되고, 자신의 게임 시스템에 맡는 방법을 고안해낼 것이다. 그리고 기획서 작성 시에도 이러한 방법에 대해서 자세히 적어 놓을 것이다. 그렇지만, 이러한 방법을 표현하는데 있어서, 글이든 그림이든 자신만의 방식으로 작성하게 될 것인데, 이러한 부분에서 팀원들과 오해가 생길 수 있다. 그리고 실제로 데모 게임을 만들고 난 다음에 서로가 다른 생각을 하고 있었다는 것을 생각할 수 있다. UML은 이러한 부분에서 표준적인 방식으로 표현을 하기 때문에 UML로 작성을 하게 되면 서로가 같은 생각을 하게 된다.

또한, 다른 게임을 분석할 경우, UML로 그려보게 되면 차이점을 쉽게 알 수 있다. 그리고 간단하게 생각하고 넘어간 부분을 세부적으로 생각해 볼 수도 있다. 월드오브워크래프트처럼 캐릭터가 방향을 바꾸는 경우에 상체의 방향을 먼저 바꾼 뒤 하체의 방향을 바꾼다는 것을 처음에는 생각하고 있지 않다가 데모를 보고 잘

〈그림 5-2-1-04〉 사례2 – 활동 다이어그램 : 리니지2 캐릭터 이동

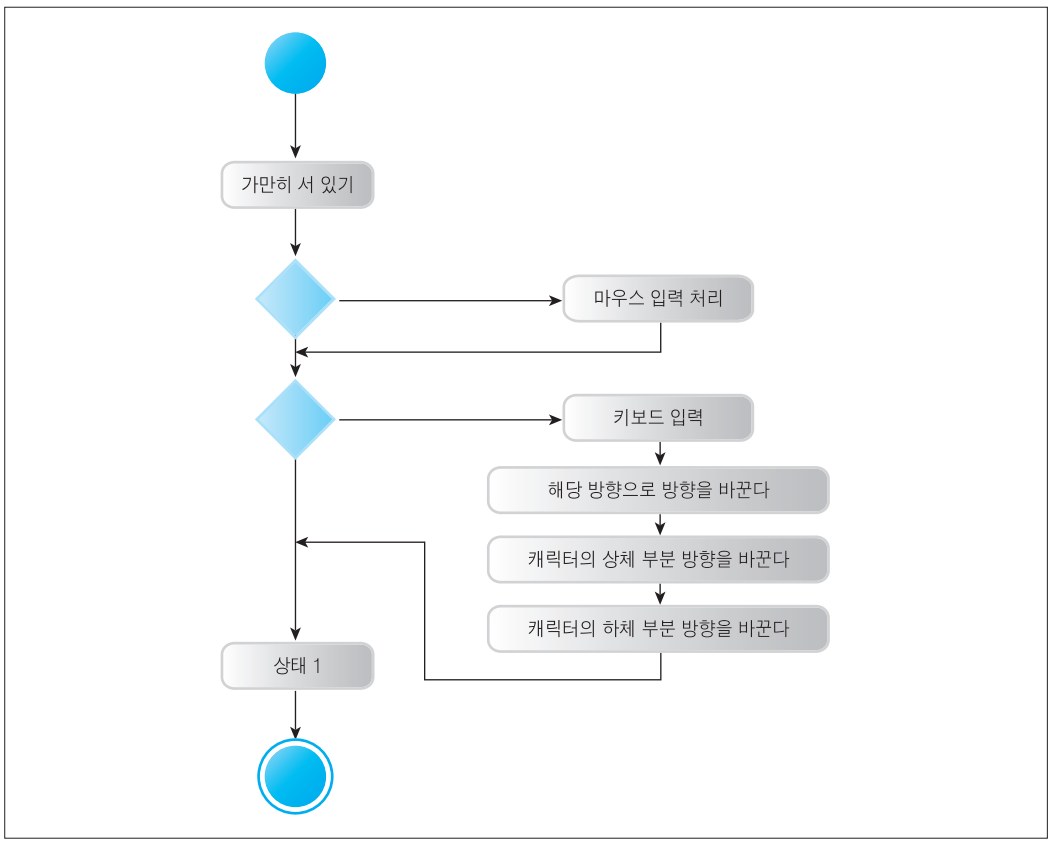


못된 것을 알고 다시 구현을 하게 되는 경우가 발생한다면 직접 이 작업을 해야 하는 프로그래머는 당황할 것이다. 물론, 기술적으로는 가능할 것이다. 그러나, 현재의 객체지향 프로그래밍 방법에서는 처음에 설계에 중점을 많이 두고 작업을 하게 되고 설계에 시간 투자를 많이 하려고 한다. 그러나, 프로그래머 입장에서 처음에 얘기가 없었던 부분에서 수정을 하게 되면 설계를 수정해야하는 부담이 생길 수 있다. 특히, 다른 분야와 다르게 게임개발은 개발 도중에 기획이 바뀌는 경우가 자주 있기 때문에 다른 분야보다는 프로그래머가 미리 이러한 부분을 염두에 두고 작업하는 경우가 많은 것 또한, 사실이다. 그렇지만, 이러한 작업 변경이 당연하다고 생각하면 안되며, 기획단계에서 완벽을 기하는 것이 현명한 방법이라 하겠다.

두 번째로 살펴볼 것은 상태 다이어그램이다. 상태 다이어그램은 사건이나 시간에 따라 시스템 내의 객체들의 상태 변화를 표현한 것이다.

게임 내에서 상태 변화의 예는 많이 있다. MMORPG의 경우, 일반적으로 전투 시스템이 존재하게 되는데, 간단한 예로는 캐릭터가 평화모드에 있다가 몬스터나 다른 캐릭터를 공격

〈그림 5-2-1-05〉 사례2 – 활동 다이어그램 : 월드오브워크래프트 캐릭터 이동

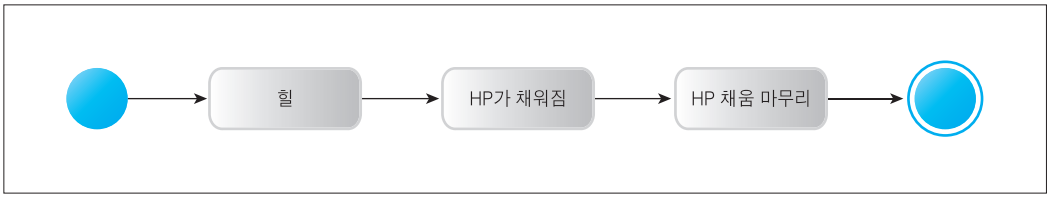


하게 되면 전투 모드로 바뀌게 되는 경우, 또는 전투 중에도 버프나 디버프에 의한 캐릭터의 상태가 변화되는 경우, 다른 예로는 시간에 따라 특정한 종족의 캐릭터 상태가 변화되는 경우도 있을 수 있다. 밤에는 특정한 캐릭터의 공격력이 높아지는 경우가 이러한 예일 것이다.

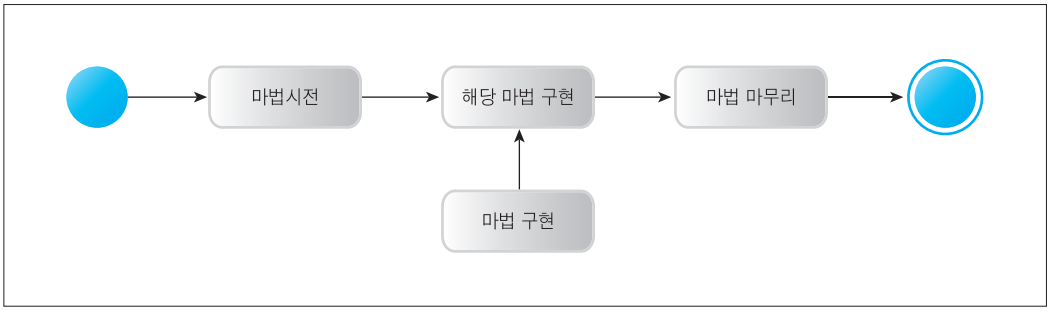
자신의 캐릭터가 스스로 힐 마법을 시전하는 경우, 옆에서 다른 캐릭터가 다른 마법을 나에게 사용할 수도 있다. 그것이 버프 마법이든, 아니면 디버프 마법이든 이 경우에는 사례3(상태 다이어그램 : 복합적인 상태 변화) 같이 표현할 수도 있을 것이다.

이러한 상태 다이어그램의 경우도 중요한 부분이라고 할 수 있는데, 예전의 디아블로를 예로 들면, 캐릭터가 몬스터를 공격하는 상황에서 캐릭터가 공격을 받으면 타격받은 데미지 애니메이션을 취하게 된다. 이런 상황에서 캐릭터는 아무것도 할 수가 없다. 1:1로 몬스터와 전투를 하는 상황에서는 큰 문제가 아닐 수도 있지만, 주위 8방향에서 모든 몬스터에게 공격을 받는 경우에 자신의 캐릭터는 타격 데미지 애니메이션을 표현하느라 아무것도 하지 못하고, HP가 0이 되어 죽게 되는 경우가 발생하였던 적이 자주 있었다. 물론, 이러한 부분에 있어서 수치적으로 조정을 할 수도 있을 것이고, 게임 기획자의 의도였다면, 플레이어는 이러한 상황이 발생하지 않도록 안전하게 사냥을 하는 방법을 찾아야 할 것이다. 그러나, 여기서 이야기하고자 하는 것은 그러한 상황이 발생하게 되는 것에 대해 미리 생각하고 게임을 기획해야 한다는 것이다. UML 다이어그램을 이용하게 되면 이러한 부분에 대해서도 자연스럽게 생각을 하게 되기 때문에 게임을 기획한 기획자가 의도하지 않은 문제가 발생하는 것을 방

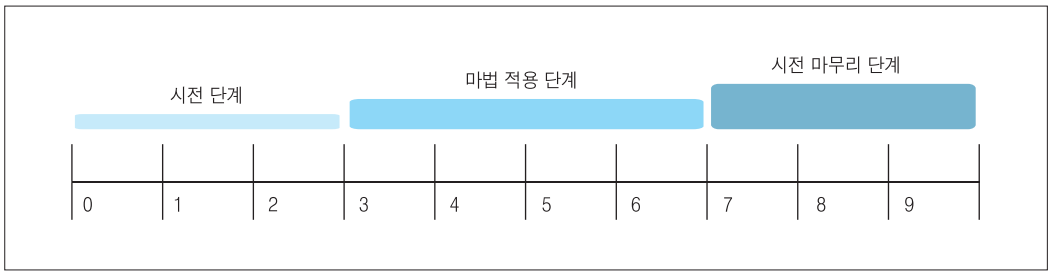
〈그림 5-2-1-06〉 사례3 – 상태 다이어그램



〈그림 5-2-1-07〉 사례3 – 상태 다이어그램 : 복합적인 상태 변화



〈그림 5-2-1-08〉 사례4 – 타이밍 다이어그램



지할 수 있을 것이다.

마지막으로 살펴볼 것은 타이밍 다이어그램이다. 타이밍 다이어그램은 시간에 따른 변화를 표현할 수 있다.

〈그림 5-2-1-08〉은 마법의 시간에 따른 흐름을 보여주고 있다. 마법의 시전은 3초 동안 이루어지고, 마법 효과를 적용받는 시간은 4초, 마법의 마무리가 3초간 이루어진다.

타이밍 다이어그램에서는 시스템적인 관점 외에도, 캐릭터의 레벨에 따른 밸런싱 단계에서도 유저가 느껴야 하는 재미요소를 타이밍 다이어그램으로 표현함으로써 전체적인 큰 그림을 그려볼 수 있을 것이다.

지금까지 크게 3가지 다이어그램에 대해서 살펴보았다. 활동 다이어그램, 상태 다이어그램, 타이밍 다이어그램, 이 3가지 다이어그램 외에도 UML에는 게임 기획에 활용할 만한 다이어그램들이 있다. 단지 어떻게 활용하는가가 문제이다. 클래스 다이어그램의 경우, 프로그래밍 기반이 없는 기획자가 작성하기에는 무리가 있을지 모르지만, 기획자가 UML 다이어그램을 활용하여 게임을 디자인하기 위해서는 자신만의 추상적인 개념으로 클래스 다이어그램을 작성하고, 그것을 바탕으로 다른 다이어그램을 그려나간다면, 전체적인 설계를 하는데 있어 많은 도움이 될 것이다. 또한, 여기서 설명하지 않은 유스 케이스 다이어그램, 시퀀스 다이어그램도 게임을 기획하는 기획자의 의도에 따라 활용한다면 얼마든지 도움이 되는 다이어그램들이다.

5. 결론

앞에서 살펴본 다이어그램들은 7개의 다이어그램으로 구성되어 있다. 게임 제작 시 전부 사용할 수 있는지 없는지는 아직 더 많은 연구가 필요하다. 물론, 프로그래머와 프로그램 설계를 담당하는 개발자는 모든 다이어그램을 사용해서 제작이 가능하겠지만, 많은 경우 프로그래밍을 잘 알지 못하는 게임 기획자의 경우, 모든 다이어그램을 사용해서 표현하는 것이 불가능할 수도 있다. 또한, 게임 기획자가 7개의 다이어그램을 모두 사용해서 게임을 디자인했다 하더라도, 클래스 다이어그램과 같은 경우는 실제로 제작에 사용하기에는 무리가 따르는 경우가 발생해서 프로그래머가 다시 그려야하는 경우도 있을 것이다. 그러나, 다이어그램을 활용함에 있어서 활동 다이어그램, 타이밍 다

이어그램 등과 같은 다이어그램들은 누가 그리든지 거의 유사하게 표현되어져야 할 것이다. 이렇게 그려진 다이어그램들은 실제 프로그래밍 설계 시에도 참고가 가능하다.

지금까지의 게임 기획은 경험에 의존하는 경우가 많이 있었지만, 앞으로는 체계적인 방법으로 기획을 해야 할 것이다. 현재에도 많은 회사에서는 나름대로 체계를 갖추고 있지만, UML를 사용한 표준적인 방법을 도입한다면 보다 유용하게 사용할 수 있을 것이다.

● UML 사용의 장점

- 배우기 쉽다.
- 보다 이해하기 쉽도록 비주얼하게 제시할 수 있다.
- 하나의 상황을 여러 시각으로 바라볼 수 있다.
- 표준화된 커뮤니케이션이 가능하다
- 프로그래머와 기획자 간의 커뮤니케이션이 명확해진다.
- 게임 시스템 분석에 적용 가능하다.

UML를 사용하더라도 표현된 모든 것이 논리적이고 완벽하게 표현되었다고 결론내릴 수는 없다. 단지 팀원 간의 커뮤니케이션에 있어서 표준화된 방법을 사용함으로써 잘못된 표현으로 게임 시스템을 오해하는 일을 방지할 수 있다. 또한, 자신이 생각한 시스템을 UML 다이어그램으로 표현해 봄으로써 논리적인 생각을 추구할 수 있다. 게임 디자인 작업의 모든 곳에 UML을 사용하여 표현할 수는 없겠지만, 거의 대부분의 게임 시스템은 어떤 상태의 흐름도를 그릴 수 있다. UML은 우리에게 막강한 툴임에 틀림없다. UML 도구를 사용하면 개발팀 내 표현방식이 표준화되고, 비주얼을 제시함으로써 보다 동적인 커뮤니케이션이 가능할 것이다.

제 2 절 게임과 가상세계

1. 가상세계 제작 동향

가상세계(Virtual World)는 다수의 플레이어들이 자신을 나타내는 가상의 아바타(Avatar)를 통해서 상호작용할 수 있는 컴퓨터를 기반으로 구현된 가상공간을 의미한다. 가상세계는 컴퓨터를 매개로 플레이어 간의 커뮤니케이션이 진행되는 인터렉티브 환경으로서, 플레이어 개개인이 서로 만나고, 정보를 교환하고, 특정 목표를 위해 협동하거나 경쟁하는 등의 사회적 활동이 가능한 공간이다. 이인화 교수는 가상세계를 몰입 및 상호작용의 정도에 따라 게임형 가상세계와 생활형 가상세계로 구분하였다.

게임형 가상세계는 플레이어들이 상호작용하게 되는 물리적인 공간뿐만 아니라, 상호작용의 방법과 목표까지 제공하는 가상세계다. 이 공간에서는 가상세계가 플레이어에게 할 일을 제시하는 구조이고, 플레이어는 특정 캐릭터(역할)를 선택하여 제공되는 콘텐츠를 즐긴다. 현재 게임형 가상세계는 생활형 가상세계에 비해 시장 규모가 매우 크며 다양한 제품이 서비스 중이다. 게임형 가상세계는 전문적인 개발사가 제작한 콘텐츠를 플레이어가 소비하는 구조이므로 상대적으로 생활형 가상세계에서 플레이어들이 만들어내는 콘텐츠에 비해 품질이 높다. MMOG(Massively Multiplayer Online Game)를 포함한 대부분의 온라인게임이 여기에 포함된다.

생활형 가상세계는 플레이어 사이의 상호작용을 위한 환경을 제공하는 것에 초점을 맞춘 가상세계 형태이다. 생활형 가상세계는 플레이어의 아바타나 개인적인 공간을 마음대로 편집하거나 바꿀 수 있으며 상황에 따라 다양한 역할을 수행하고 콘텐츠를 제작 또는 공유할 수 있는 환경을 제공한다. 이 범주의 가상세계에서는 플레이어 사이에서 발생하는 상호작용 그 자체 혹은 제공되는 환경을 이용하여 플레이어가 제작한 2차적 콘텐츠가 중심이 된다. 플레이어가 직접 콘텐츠를 제작하므로 가상세계가 시시각각 변하고 항상 새로운 콘텐츠를 제공할 수 있다. 싸이월드나 Facebook과 같은 SNS(Social Networking Service)가 여기에 해당된다.

분류의 관점에서 가상세계를 바라보는 시각은 여러가지가 존재한다. 위에 설명한 시각도 있지만 생활형 가상세계를 게임형 가상세계에 속한 하나의 장르로 보기도 하고, 생활형 가상세계와 SNS서비스를 합하여 가상세계로 보고 게임은 분리하여 다루는 시각도 있다. 이 글에서는 생활형 가상세계를 중심으로 현재 상황을 살펴보고 발전 가능성을 예측해 보고자 한다.

1) 가상세계 시장 규모

2008년 현재 게임형 가상세계를 제외한 전 세계의 가상세계 가입자는 약 2억 3,000만명이다. 이 중 10대를 위한 가상세계 서비스의 플레이어 수가 전체의 67%로 가장 높은 비율을 차지하며, 서비스 중이거나 개발 중인 가상세계 서비스의 수도 가장 많다. 가상세계 리서치

〈표 5-2-2-01〉 주요 생활형 가상세계의 가입자 수 (단위 : 백만명)

사용자 평균연령	가상세계 이름	가입자 수
10세 이하	Chapatiz	0.7
	Webkinz	1
	Neopets	45
	Barbie Grils	11
	Club Benguin	15
10대	Whyville	3
	Habbo Hote	90
	Gaia Online	12
	Stardoll	15
	IMVU	20
	vSide	0.3
20대	vMTV	0.8
	There	1.5
	Kaneva	1
	Activeworlds	1
30대	Second Life	13
계		230.3

※ 자료 : KZERO, Virtual Worlds By The Numbers: Today and The Future, Virtual World Expo 2008.

그룹인 KZERO의 분석에 따르면 10대 플레이어의 폭발적인 증가 요인은 아바타 꾸미기가 핵심이다. 10세 이하 어린이를 위한 가상세계 서비스는 전체의 25% 정도를 차지한다. 장난감 회사가 자사 브랜드 장난감을 가지고 놀 수 있는 가상세계 환경을 만들어 제공하고 가상세계에서 장난감을 구매하면 실제 장난감도 주는 사업 모델을 사용한 것이 최근 플레이어 수 증가의 동력원이라 할 수 있다. 20대 이상을 대상으로 하는 가상세계 시장의 경우, 상대적으로 플레이어 수가 적으나 직접적인 구매력은 높은 층이기 때문에 플레이어 수에 비해 수익 규모는 높다. 플레이어 수가 9,000만명인 10대 대상의 가상세계 하보 호텔의 연 수익이 7,700만달러(2006년)인 것에 비해, 플레이어 수 1,350만명 수준인 Second Life의 연 수익은 5,000만달러 규모로 상당히 크다. 가상재화 판매가 주 수입원이 되는 생활형 가상세계 시장은, 월정액 모델을 사용하는 가상세계에 비해 플레이어 수 대비 수익은 낮으나, 현재 가상재화 판매 시장이 매우 빠르게 커지는 추세이기에 미래의 기대수익은 매우 높다.

2) 가상세계 연구 및 제작 동향

현재 서비스 중이거나 곧 서비스할 예정인 가상세계는 전 세계에 약 100여 개 정도로 파악된다. 2년 내에 서비스가 시작될 것으로 예상되는 개발 사실이 발표된 가상세계의 수도 100여개 이상이다. 가상세계 서비스가 주목을 받음에 따라 투자도 크게 늘어나고 있다. 2008년 1분기동안 23개 가상세계 관련 회사가 1억 8,000만달러의 투자를 받았고, 2007년 한 해 동안 35개 가상세계 회사에 10억달러가 투자되었다. 이 중 가장 큰 규모의 단일 투자 대상은 7억달러에 디즈니가 사들인 Club Penguin이고, 1억달러의 투자를 받은 중국의 9You가 그 뒤를 이었다. 한국의 누리엔 소프트웨어도 1,500만달러의 투자를 유치했다.

가상세계 서비스는 특징에 따라 크게 두 가지로 분류할 수 있다. 아바타 채팅과 미니 게임 등의 요소가 중심이 되는 10대와 그 이하 연령층을 대상으로 한 가상세계와 현실세계와 비슷한 모습으로 다양한 콘텐츠와 높은 자유도를 제공하는 미러월드(Mirror World) 형태의 가상세계가 그것이다. 하보 호텔이 전자를, Second Life가 후자를 대표하는 가상세계로 볼 수 있다.

현재 개발 중인 가상세계들은 Second Life와 같이 모든 것을 다 할 수 있는 가상세계가 아니라, 특정 대상과 주제를 가진 모습을 띠는 경우가 많다. Proton Media에서 개발 중인 Virtual Congress는 고등학생들을 대상으로 한 교육용 가상의회 서비스이고 SCE의 PlayStation Home은 콘솔 기반 가상세계로 플레이어들이 모여서 게임을 하거나 콘텐츠 공유를 할 수 있는 환경을 제공한다. 한국에서는 펀투펀, 퍼니또 등 교육적 기능을 접목한 가상세계가 다수 서비스 중이다.

가상세계 관련 연구 역시 활발하다. 가장 큰 규모의 게임관련 컨퍼런스인 GDC(Game Developer Conference)에서는 2008년, 40여 개의 세션이 가상세계를 주제로 다루었다. 가상세계 내에서의 비즈니스에 중심을 둔 학회인 Virtual World Conference에서는 가상세계 내의 미디어, 엔터테인먼트, 마케팅 등 기업활동과 가상세계 관련 법률에 관한 연구를 다룬다. 또한, Metaverse roadmap을 통해 전 세계적인 가상세계 관련 연구 및 개발 상황과 정보를 공유할 수 있다. 기타 여러 IT리서치 기업 및 연구소에서도 가상세계 관련 연구를 진행하고 있다.

2. 가상세계 사례 연구

1) 세컨드 라이프(Second Life)

세컨드 라이프는 린든 랩(Linden Lab, Linden Research Inc)에서 2003년 개발한 가상세계이다. 플레이어는 세컨드 라이프 내에서 '거주자(Resident)' 라는 아바타를 통해 다른 플레이어와 만나고 그룹 활동에 참가하며, 가상 자산과 서비스를 창조하고 다른 이와 거래하는 등 보편적인 메타버스(Metaverse)의 모습과 결합한 소셜 네트워크 서비스(Social Networking Service, SNS)를 제공받는다.

세컨드 라이프는 이름 그대로 플레이어에게 현실의 삶과 다른 제 2의 인생을 살아갈 수 있는 기회를 제공하는 가상세계이다. 플레이어는 세컨드 라이프가 제공하는 가상공간 속에서 다양한 방법으로 자신을 표현하고 다른 플레이어와 교류할 수 있다. 가상의 아바타를 꾸미고, 가상의 도시와 건축물을 지어간다. 가상세계 내에서의 노동을 통해 가상재화를 얻으며 세컨드 라이프의 세계를 살아갈 수 있다. 세컨드 라이프 공식 가이드북에서는 세컨드 라이프를 이렇게 표현한다.

"세컨드 라이프는 기본적으로 플레이어가 원하는 그 무엇이든 될 수 있다. 그것은 결국 플레이어의 가상 인생이며, 이 인생으로 무엇을 할 것인지는 전적으로 플레이어에게 달려 있는 것이다. 세컨드 라이프는 가상의 환경이다. 여기서는 그 안의 콘텐츠가 거의 대부분 플레이어에 의해 창조된다. 그렇다면, 세컨드 라이프가 당신에게 어떤 의미를 가질까? 그 대답은 바로 당신이 거기에서 무엇을 하고 싶은지에 달려있다. 온라인 상에서 사람들을 만나 이야기를 나누고, 실시간으로 어떤 일을 함께 하기를 즐기는가? 무엇인가를 창조하고 실현하는 것을 좋아하는가? 사업 경영을 통해 이윤을 창출하고 싶은가? 그렇다면 세컨드 라이프에 온 것을 환영한다. 과연 세컨드 라이프에서 당신은 어떤 일을 할 수 있을까? 그 대답은 당신의 상상력의 범위가 어디까지 미치는지에 달려있다."

가. 플레이어 규모

세컨드 라이프의 총 가입자 수는 2008년 2월 현재 1,350만명으로 최근 6개월 동안 평균 매월 50만명의 플레이어가 새로 유입되었다. 액티브 유저(Active user : 60일 이내에 접속한 플레이어의 수)의 수는 110만명이며, 세컨드 라이프 내의 토지를 구입한 유료 플레이어(Premium account)는 9만명, 평균 동시 접속자 수 45,000명을 유지하고 있다.

세컨드 라이프 플레이어의 평균 연령은 약 30세이며 30~40대 이상의 중장년층이 전체 유저의 60% 비중을 차지하고 있는 것이 특징이다. 아시아 온라인게임 유저의 연령대가 주로 10~20대인 것과 비교하면 세컨드 라이프의 플레이어 연령이 현저히 높다는 것과 알 수 있다.

나. 수익 모델

세컨드 라이프는 현재 세 가지 주요 수익 모델이 있다.

(1) 프리미엄 멤버십

세컨드 라이프의 기본 회원으로 가입하면 토지 소유 이외의 모든 활동은 무료로 즐길 수 있다. 그러나, 토지를 소유하기 위해서는 프리미엄 회원으로 가입해야 한다. 토지를 소유하면 그 공간은 플레이어가 원하는 용도로 사용하거나 꾸밀 수 있다. 프리미엄 회원으로 등록하면 512㎡의 토지를 소유하게 된다. 그러나, 꼭 프리미엄 멤버십을 통하지 않고도 플레이어가 토지를 사용할 수 있기 때문에 실제 프리미엄 멤버십을 이용하는 플레이어의 수는 많지 않다. 대체로 다른 플레이어가 소유한 토지를 대여하는 형태로 토지를 사용하는 플레이어가 많다. 현재 9만명 정도의 유료 플레이어가 존재하며, 이로 발생하는 수익은 대략 월 90만달러 정도로 추산된다.

(2) 추가 토지 사용료

프리미엄 멤버십에 기본으로 제공되는 512㎡ 이상의 토지를 사용하려면 땅 넓이에 따라 사용료를 지불해야 한다. 2008년 4월 현재 14,278개의 지역(Island)이 플레이어 소유로 등록되어 있으며, 한 지역당 최소 195달러(작은 토지의 경우, 면적당 사용료가 더 높다)의 토지 사용료가 부과되므로 매월 280만달러 이상의 수익을 예상할 수 있다.

(3) 환전 수수료

세컨드 라이프에서는 자체 외환 시장인 LindeX (Linden Dollar exchange)에서 실제 미국달러와 세컨드 라이프 내의 통화인 린든달러를 환전할 수 있다. 이 때 환전 수수료로 3.5%를 받는다. 미국달러 대 린든달러의 환율은 현실 통화와 마찬가지의 요인에 의해 결정된다. 평균적으로 1 미국달러는 270 린든달러로 환전된다. 2008년 4월 현재, 환전 규모는 하루 8,000만 린든달러(미국달러로 약 30만달러, 월간 900만달러) 수준으로, 환전 수수료 수입은 월 36만달러에 이른다.

다. 특징

세컨드 라이프의 세계에서는 이전의 다른 가상세계에서는 볼 수 없었던 다양한 종류의 상호작용이 가능하고, 플레이어의 요구에 따라

얼마든지 그 범위를 확장시켜 나갈 수 있다. 이러한 열린 구조의 세계는 현실에서 가능한 모든 것이 이루어질 수 있는 또 다른 현실로서의 가상세계를 구축함과 동시에, 현실 세계와 연결시킴으로써 가상세계 내의 활동이 현실 세계의 활동에 영향을 주는 사회적, 경제적 실체를 가진 가상세계의 가능성을 보여주고 있다.

(1) 플레이어의 참여에 의해 구성되는 세계

세컨드 라이프는 플레이어의 직접적인 참여에 의해 구성되는 세계이다. 게임을 포함한 기존의 다른 가상세계는 간접적 또는 부분적인 이용자 참여만을 인정하는 모델이었다. 즉, 플레이어는 요구를 전달하기만 하고 기업이 이를 판단하여 개발에 반영할 것인지 여부를 결정한다. 직접적인 플레이어 콘텐츠 제작을 지원하는 가상세계도 있었지만, 플레이어가 영향을 줄 수 있는 것은 가상세계 전체에 비하면 매우 적은 부분에 불과했다. 반면, 세컨드 라이프는 플레이어가 직접적으로 콘텐츠를 창작한다. 제작사인 린든 랩은 플레이어가 콘텐츠를 제작할 수 있는 기술적 기반만을 제공하며 세컨드 라이프 내의 거의 모든 콘텐츠는 플레이어가 제작한다. 샌드박스(Sand Box)라는 일종의 3D 모델링 툴(3D Modeling Tool)을 통해 사물의 외형을 제작하고 LSL(Linden Script Language)을 통해 기능을 부여함으로써 이론적으로는 플레이어가 원하는 어떠한 콘텐츠도 제작할 수 있다. 세컨드 라이프에서는 원하는 콘텐츠를 얼마든지 생산할 수 있는 환경 덕분에 커뮤니티로서의 기능을 비롯한 몇 가지 정해진 활동으로 제한되어 있는 다른 가상세계와는 달리 교육과 쇼핑, 방송, 광고, 게임, 기업 및 정치활동에 이르기까지 온라인 네트워크 상에서 가치를 가지는 모든 활동의 총체적 포털로서 기능할 수 있게 되었다.

(2) 실제 세계와 연결되는 가상경제 시스템

세컨드 라이프에서는 현실 세계와 거의 유사한 경제활동이 가능하다. 세컨드 라이프에서는 창작물에 대해 플레이어의 소유권 및 저작권을 인정받을 수 있다. 플레이어는 자신의 창작물을 마음대로 복사하여 판매할 수 있으며, 다른 플레이어가 변형하거나 도용하지 못하도록 막을 수 있다. 따라서, 세컨드 라이프 내의 창작물들은 확실한 가치를 지니게 되고, 이를 소유하기 위한 경제활동이 활발히 일어난다. 세컨드 라이프는 자체 외환 시장인 LindeX를 통해 린든달러와 미국달러를 환전할 수 있는 시스템을 제공하기 때문에 가상세계 내의 경제 활동을 더욱 촉진시키며, 현실 경제와 연결시킨다.

세컨드 라이프에서는 아바타 의상 및 텍스처 디자이너, 건축가, 애니메이터, 댄서, 금융 투자자 등 다양한 직업이 존재한다. 세컨드 라이

〈표 5-2-2-02〉 세컨드 라이프 내에서 수익을 올리는 플레이어 수

수익(USD)	2008. 3.	2008. 4.
$ 10 USD	31,082	29,598
$ 10 to $ 50 USD	16,566	16,999
$ 50 to $ 100 USD	3,754	3,642
$ 100 to $ 200 USD	2,389	2,493
$ 120 to $ 500 USD	2,093	2,191
$ 500 to $ 1,000 USD	935	961
$ 1,000 to $ 2,000 USD	537	515
$ 2,000 to $ 5,000 USD	637	358
$ 5,000 USD	165	173
계	58,158	56,930

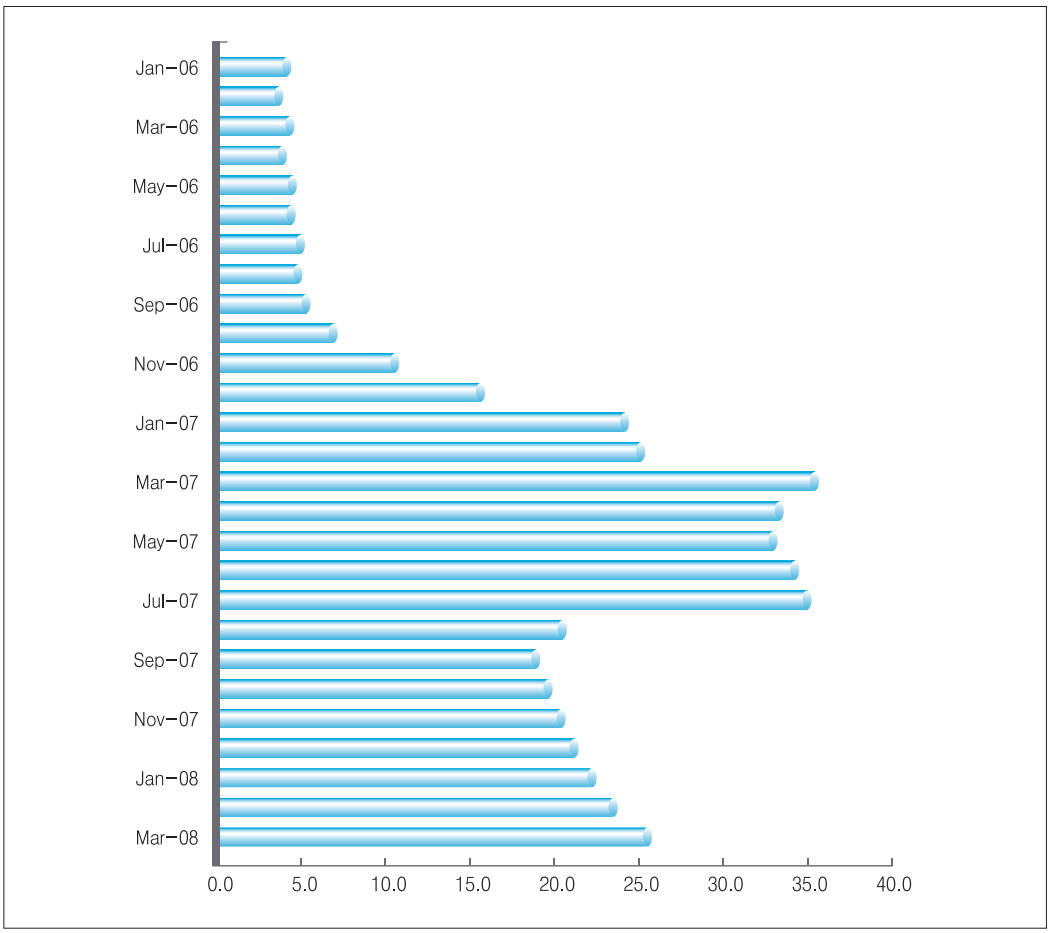
프 내의 사이버 경제 활동을 통해 매월 5,000 달러(USD) 이상의 수익을 올리는 유저가 150 명 이상이며, 부동산 사업을 통해 2년만에 100 만달러(USD) 이상의 수익을 거둔 플레이어도 있다. 기업의 진출도 활발하여 다양한 기업들이 세컨드 라이프에서 활동하고 있다. 신제품이나 서비스의 테스트, 기업과 제품 홍보, 실제 제품의 판매와 고객 지원 등의 활동을 하고 있다.

라. 가능성과 한계

(1) 가상세계 비즈니스

세컨드 라이프 내의 경제활동은 매우 다양하고 그 규모가 상상 이상으로 크다. 누구나 사업 아이디어만 있다면 뛰어들 수 있다. 현재, 세컨드 라이프에서는 6만개 이상의 사업이 흑자를 내고 있다. 2008년 3월 기준으로 세컨드 라이프의 전체 거래량은 연간 3억달러 규모로 추산되며, 반(反)도박 정책 발효 이전에는 연간 4억 달러 규모였다. 이는 거의 전적으로 가상세계

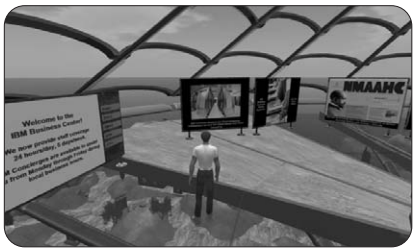
〈그림 5-2-2-01〉 세컨드 라이프의 경제 규모 (단위 : 백만달러)



내에서의 사업에 의한 거래량이다. 현재 많은 기업들이 세컨드 라이프 내에 진출하여 비즈니스 활동을 벌이고 있으며, 그 영향은 현실경제에서 나타나게 된다. 때문에 세컨드 라이프의 경제적 가치를 단순히 가상세계 내의 거래량으로 측정할 수 없으며, 현실세계와 관련된 기업 활동을 고려하면 훨씬 커지게 된다.

세컨드 라이프 내의 기업 활동은 주로 마케팅과 제품 판매에 집중되어 있다.

제품이나 로고를 자주 노출시켜 인지도를 높이는 기본적인 광고 방법 외에 세컨드 라이프 내에 3D로 구현된 제품을 통해 가상 체험을 할 수 있도록 제공하고, 웹으로 연결하여 플레이어에게 자세한 제품 설명을 제공하거나 구매를 유도하는 방법을 주로 사용한다. 도요타와 메르세데스 벤츠는 홍보관을 세우고 새 모델의 디자인을 홍보하거나 세컨드 라이프 내에서 사용할 수 있는 자동차를 판매하기도 한다. IBM의 경우, 마케팅 이외에 기업 커뮤니케이션의 수단으로도 활용하고 있다. 개발자의 온라인 공동 작업을 위한 커뮤니티 공간을 구축하고, 취업 및 고객 상담소를 운영하며 사내 프레젠테이션을 위한 공간도 마련하여 사용하고 있다. 하버드 대학의 경우, 2006년 '법과 여론 재판'이라는 법학 강좌를 세컨드 라이프 내에 개설하기도 했다. 이 강의는 수강생 이외에도 청강이 허용되었으며, 이후 정규 강의 외에 비디오 자료, 토론, 강의로 구성된 확장 콘텐츠가 일반에 제공되었다. 또한, 세컨드 라이프 내에 선거 캠프를 만들고 선거 활동을 펴기도 했다. 미국의 힐러리 클린턴을 비롯하여 프랑스 대선 후보 4명이 세컨드 라이프 내에서 선거 활동을 했다.



세컨드 라이프의 IBM 비즈니스 센터

〈표 5-2-2-03〉 세컨드 라이프 기업들의 활동 영역

업체명	마케팅	제품판매	고객지원	커뮤니케이션	시제품테스트	교육
삼성전자	○					
소프트뱅크	○					
도요타	○	○				
메르세데스 벤츠	○	○				
IBM	○		○	○		
썬마이크로시스템	○	○	○			
아메리칸 어패럴	○	○			○	
아마존닷컴	○	○				
서킷시티	○	○	○			
소닉BGM	○	○				
로이터	○	○				
하버드 대학교	○					○
ABC	○			○		

세컨드 라이프 내의 비즈니스는 주로 마케팅과 제품 판매에 집중되어 있으면서도, 하보 호텔 등의 다른 가상세계나 Facebook 등의 SNS 서비스에 비해 플레이어 수가 적기 때문에 그 효과를 크게 기대하기 어렵다는 문제를 가지고 있다. 그러나, 자유로운 콘텐츠 창작과 큰 경제 규모라는 요소는 다양한 종류의 비즈니스를 발생시킬 가능성이 충분히 있다.

(2) 플레이어에 의해 제작되는 세계

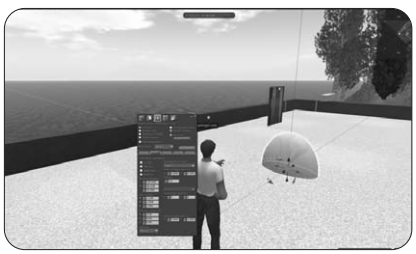
세컨드 라이프는 대부분의 콘텐츠를 플레이어가 제작하게 되면서 개발비용이 대폭 감소하였다. 콘텐츠를 추가하기 위해서는 많은 개발비용을 들여 콘텐츠를 제작해야 하는 기존 가상세계와는 달리 세컨드 라이프는 콘텐츠 제작을 플레이어에게 맡기고 가상세계 관리만을 하면서도 더 다양하고 많은 콘텐츠를 플레이어에게 제공할 수 있다. 그렇지만 플레이어가 제작하는 콘텐츠의 품질은 그 편차가 매우 심하여 수준 미달의 콘텐츠도 대량으로 생산되는 문제점도 지적되고 있다. 또한, 플레이어가 원하는 내용 또는 높은 품질의 콘텐츠를 즐기기 위해서는 경험을 통해서 직접 찾아야 하는 문제가 있다.

(3) 기술적 한계

세컨드 라이프에서 플레이어 콘텐츠 제작 인터페이스는 복잡성으로 인해 일반인이 사용하기에는 한계가 있다. 플레이어가 콘텐츠를 제작하기 위해서는 필요한 모델을 만들고 텍스처와 애니메이션을 제작하고 기능을 부여하는 단계를 거쳐야 하는데 세컨드 라이프에서 제공하는 방법으로 콘텐츠를 제작하는 것은 일반적인 플레이어에게 비효율적이다. 모델링의 경우, 세컨드 라이프에서 제공하는 프림(Prim ; primitive ; 모형을 제작하기 위한 기본 모델. 육면체, 구, 원 뿔 등)을 변형하고 조합하여 제작해야 하는데 일반적인 3D 모델링 프로그램과 크게 다를 바가 없다. 텍스처, 애니메이션의 제작도 모델링 프로그램의 난이도와 비슷한 수준으로 제공된다.

기능 부여의 경우에도 어려움이 따른다. 기능을 부여하기 위해서는 세컨드 라이프에서 제공하는 LSL(Linden Script Language)을 사용해야 하는데 이는 일반적인 스크립트 언어와 비슷한 구조로 되어 있기 때문에 프로그래밍 언어를 전혀 모르는 일반인이 자유자재로 사용하기에는 한계가 있다. 일반적인 유저들은 직접 콘텐츠를 제작하기 보다는 좀 더 전문적으로 제작할 수 있는 다른 플레이어가 제작한 창작물을 구입하여 사용하는 경우가 대부분이다. 또한, 가상세계 자체가 플레이어에게 제시하는 목표가 전혀 없기 때문에 쉽게 흥미를 잃어버릴 위험성이 있는데도 불구하고, 신규 플레이어가 세컨드 라이프의 세계에 익숙해질 수 있도록 도와줄 수단이 부족하다.

세컨드 라이프는 시각적인 면에서 일반적인



세컨드 라이프에서 Prim을 이용하여 모델 제작하기

게임형 가상세계에 비해 비주얼의 품질이 상대적으로 낮다. 정지 화면에서의 랜더링 품질의 차이뿐 아니라 동작 속도의 문제도 포함한다. 또한, 세컨드 라이프의 모든 콘텐츠는 플레이어에 의해 제작되기 때문에 어떤 장소에 가든 새로운 데이터를 계속 받아와야 한다. 특히, 텍스처 데이터의 크기 때문에 지역을 바꿀 때마다 수 십 메가바이트의 데이터를 받아야 한다. 이런 플레이어 제작 콘텐츠 데이터의 양 때문에 서버가 처리할 수 있는 플레이어의 수가 매우 적은 것으로 알려져 있다.

2) 하보 호텔(Habbo Hotel)

하보 호텔은 핀란드의 Sulake Labs가 2000년 제작하여 서비스하고 있는 가상세계다. 플레이어는 Habbo라고 불리는 아바타 캐릭터를 생성하여 친구와 대화하거나 게임에 참가하기도 하고, 아바타와 자신의 방을 다양한 아이템으로 꾸밀 수 있다. 하보 호텔은 현재 전 세계에서 월드 오브 워크래프트(World of Warcraft) 다음으로 가장 많은 플레이어가 이용하는 가상세계다. 생활형 가상세계만을 기준으로 하면 가장 큰 규모의 가상세계로 알려져 있다.

하보 호텔은 가상세계의 형태를 띤 SNS이다. 하보 호텔은 아이소메트릭 시점(Isometric view)의 호텔방의 집합으로 구성되어 있다. 이 호텔방들은 각각 독립된 공간으로 웹사이트를 통하여 호텔방 사이를 이동할 수 있다. 호텔방 하나 하나는 각 플레이어의 개인 공간이자 사교적인 공간으로서 주인이 원하는대로 꾸민 호텔방에 많은 다른 플레이어가 모여 대화를 하게 된다.

가. 플레이어 규모 및 현황

하보 호텔은 현재 32개국에서 서비스되고 있다. 2008년 4월 현재, 총 가입자 수는 9,400만명이며 한달 내에 접속한 플레이어 수는 950만명으로 집계 되었다. 이 중 63%의 플레이어가 매일 하보 호텔에 접속한다. 평균 동시 접속자 수는 10만명이다. Sulake사의 보도자료에 따르면 2005년 7월, 3,000만명이었던 가입 자 수는 2007년 10월, 8,000만명으로 증가했고 2008년 4월 현재, 9,400만명이다. 2년간 평균 185만명 수준이었던 월 평균 신규 가입자 수가 최근 6개월 동안 233만명으로 증가하였다. 하보 호텔은 철저히 10대들을 위한 서비스로 플레이어의 90%가 13~18세 사이다.

나. 수익 모델

(1) 부분 유료화 모델

하보 호텔의 기본적인 서비스는 무료로 제공된다. 그러나, 아바타나 호텔방을 꾸미기 위한 액세서리와 가구의 구입은 유료 서비스이다. 하보 호텔 내의 아이템 구입은 사이버 머니인 Credit을 통해서 이루어진다. 1 Credit은 20센트(USD)이다. 플레이어는 구매한 Credit,또는 가구, 액세서리를 다른 플레이어와 교환할 수

〈표 5-2-2-04〉 하보 호텔의 연간 매출 규모

년도	매출액(유로)	매출액(USD)
2000	170,000	220,000
2001	4,000,000	519,000
2002	2,300,000	3,000,000
2003	4,900,000	6,350,000
2004	13,700,000	17,750,000
2005	29,800,000	38,625,000
2006	59,200,000	77,000,000

있다. 하보 호텔에는 Habbo Club이라는 프리미엄 서비스가 존재한다. Habbo Club에 가입하면 일반 플레이어는 구입할 수 없는 아이템을 구입할 수 있고, VIP 전용 채팅방에 입장할 수 있다. Habbo Club에 가입하기 위해서는 한 달에 30Credit(6 USD)를 지불해야 한다. Credit 판매로 하보 호텔이 벌어들이는 돈은 2006년 기준 7,700만달러이다. 아이템 판매가 꾸준히 증가하고 있어 향후, 매출과 수익은 더 늘어날 것으로 예상된다.

(2) 광고 수익

하보 호텔은 기업 광고로도 수익을 올리고 있다. 호텔방 사이를 이동할 때 로딩 화면에 광고를 보여주며, 선키스트 라운지 등 스폰서의 광고가 포함된 호텔방이 존재한다. 또한, 광고용으로 제작된 아바타 아이템이나 가구들도 존재한다.

다. 특징

하보 호텔은 철저하게 특정 플레이어층인 10대를 겨냥하여 제작된 가상세계다. 플레이어의 90%가 13~18세라는 점에서 그 전략이 확실히 성공했다고 볼 수 있다. 특정 대상을 목표로 한 가상세계 서비스는 현재 주로 10대 이하를 목표로 한다. 그 중 하보 호텔보다 낮은 연령대(Pre-teenager)를 대상으로 하는 서비스가 상당수를 차지하고 있으며, 앞서 지적했듯이 18세 이하를 대상으로 서비스를 하고 있거나 개발 중인 가상세계는 100여개에 달한다. 하보 호텔과 이들의 공통적인 특징은 콘텐츠가 매우 단순하고 접근성이 높고 사용하기 쉬운 구조를 가지고 있다는 점이다. 하보 호텔의 콘텐츠를 한마디로 요약하면 시각적인 채팅 도구(Visual chat utility)이다. 단순히 자신 소유의 호텔방을 꾸미고 플레이어끼리 모여서 대화하는 것이 주된 콘텐츠이다.

하보 호텔은 설치도 필요 없고 설정도 필요 없다. Adobe Shockwave로 제작된 웹기반 프로그램으로 웹 브라우저만 있으면 어디서든 바로 실행할 수 있다. 콘텐츠가 많으면 필연적으로 발생할 수 밖에 없는 복잡성이 하보 호텔에는 없다. 옷과 성별만 선택해서 아바타를 만들고 아무 호텔방에 들어가서 채팅을 하면 된다. 그것이 즐거운 플레이어는 남아서 자신의 방도 꾸며 보고 친구들을 초대해서 자랑도 한다.

하보 호텔의 리드 디자이너인 Sulka Haro는 하보 호텔을 'Gameless game'이라고 표현한다. 하보 호텔의 플레이어들은 항상 'Play'라고 부를 수 있는 무언가를 한다. 단지 뛰어내리는 중에 움직일 수 있다는 것 빼고는 아무것도 없는 다이빙대에서 수백 번씩 뛰어내린다든가, 통나무 위에 사람(아바타)이 서 있을 때 밀어서 떨어뜨린다든가 그러나, 목적도 보상도 선택도 없는 이 '놀이'는 게임이라고 부를 만한 것이 아니다. 하보 호텔에서 일어나는 역할놀이(Role play) 역시 마찬가지다. 자신의 방을 마굿간으로 꾸며 놓고 말 행세를 하고 다른 플레이어는 조련사 흉내를 낸다든가, 음식점으로 만들어 놓고 들어오는 플레이어들에게 인사하고 서빙하는 척 한다든가 하는 등 실제로는 플레이어끼리의 대화 이외에 아무런 인터렉션도 일어나지 않기 때문에 게임이라고 부를 수는 없다. 그러나, 플레이어들은 이런 행동을

좋아하고 즐긴다.

라. 가능성과 한계

하보 호텔은 가상세계 형태의 소셜 네트워크 서비스(SNS)이다. 소셜은 가상세계 성공의 핵심 요소이며, 현실을 적극적으로 네트워크 세계에 적용하려는 메타버스(Metaverse)의 핵심이자, 네트워크 기반의 모든 콘텐츠 및 서비스의 핵심이기도 하다. 하보 호텔은 이용자가 적극적인 상호작용에 따른 소셜 네트워크를 요구하고 있음을 보여주는 사례일 것이다. 뛰어난 기술력으로 무장된 서비스도 아니며 초기 온라인게임 정도의 가볍고 심플한 그래픽만으로 이용자의 적극적인 참여를 유도하고 있다. 이는 현재 필요한 것은 현실과 흡사한 그래픽이나 복잡한 시스템보다 소셜이라는 관계가 핵심임을 보여주고 있다. 그러나, 한국에서는 팝플(www.popple.co.kr), 펀투펀(www.fun2fun.co.kr) 등 이와 유사한 가상세계 서비스가 이루어지고 있지만 아직까지는 크게 성공을 거두지 못했다.

3) 플레이스테이션 홈

플레이스테이션 홈은 Sony Computer Entertainment(이하 SCE)의 가정용 콘솔 게임기인 Playstation3(PS3) 플레이어들이 만나서 대화하고 같이 게임을 하거나 자신의 공간을 꾸미고 콘텐츠를 공유할 수 있는 환경을 제공하는 가상세계이다.

플레이스테이션 홈 서비스는 PS3의 세컨드 라이프라 불릴 정도로 세컨드 라이프와 유사한 점이 많다. 가상공간에서 플레이어의 아바타와 플레이어의 개인 공간인 아파트를 꾸미고 새로운 가구나 의상, 애완동물, 액세서리를 구입하고 다른 플레이어와 교환할 수 있다. 플레이어는 자신이 가진 영화나 음악, 사진 등의 콘텐츠를 자신의 아파트에서 다른 플레이어와 공유하고 다양한 게임을 함께 즐길 수 있다. 또한, 플레이스테이션 홈 안에는 실제 상품을 구입할 수 있는 온라인 3D 쇼핑몰이 존재한다.

플레이스테이션 홈은 2008년 5월 현재 클로즈 베타테스트 중이며, 2008년 가을 이후 일반 플레이어를 대상으로 테스트를 진행할 예정이다. 플레이스테이션 홈은 PC가 아닌 비디오게임 콘솔 환경에서 구현된 첫번째 가상세계이다. 따라서, 플레이어층은 콘솔 게임을 즐기는 플레이어로 제한되며, 플레이스테이션 홈의 기능도 다른 플레이어와 모여서 게임을 하기 위한 중간자로서의 역할에 초점이 맞추어져 있다. 플레이스테이션 홈은 게임형 가상세계를 포함하여 현재까지 서비스한 가상세계와 비교하였을 때 시각적으로 가장 뛰어난 품질의 3D 가상공간으로 인정받고 있다. 또한, 서드 파티와 최종 플레이어가 각종 미디어, 게임, 아이템 등의 콘텐츠를 제작하여 플레이스테이션 홈에서 공유 또는 판매할 수 있다.

가. 수익 모델

(1) 직접적인 상품 판매

SCE에서 직접 제공하는 상품의 판매로 수익을 올린다. 주로 옷이나 액세서리 등의 아바타 상품과 아파트를 꾸밀 가구 등의 아이템을 판매할 것으로 예상된다. 잘 꾸며진 아파트를 통째로 판매하는 상품도 존재한다.

(2) 서드 파티 콘텐츠 판매 수익 분배

서드 파티에서 개발한 콘텐츠를 플레이스테이션 홈을 통해 판매하고, 수익의 일부를 가져가는 형태의 수익 모델이다. 서드 파티 콘텐츠는 매우 다양할 것으로 예상된다. 그러나, 기본적으로 비디오게임 콘솔 환경이기 때문에 게임과 관련된 상품이 대부분을 차지할 것으로 예상된다. 아바타 아이템 및 가구, 기타 비디오와 사운드 상품, 그리고 플레이스테이션 홈에서 직접 즐길 수 있는 게임은 물론, 현재 PSN (PlayStation Network) 스토어에서 판매되는 상품도 플레이스테이션 홈에서 판매될 것이다. 또한, 가상세계 내에서만 사용되는 가상재화뿐 아니라, 실제 제품의 판매와 광고도 이루어져 3D 쇼핑몰로서의 기능도 겸할 것으로 예상된다.

(3) 옥션 수수료

플레이어 간 아이템 판매 및 교환을 위한 중개자로서 경매 시스템을 도입할 예정이다. 이때 경매장을 매개로 발생한 거래에 대해 수수료를 받는다.

(4) 광고

플레이스테이션 홈 환경의 옥외 광고를 기본으로 각종 프로모션 아이템과 가구, 영화 및 게임 타이틀의 예고편 동영상 등의 광고를 게시하여 수익을 올릴 것으로 예상된다.

나. 특징

(1) 높은 비주얼 퀄리티

플레이스테이션 홈의 외관상 가장 큰 특징은 매우 높은 수준의 그래픽이다. 현재 서비스 중인 게임형 및 생활형 가상세계 서비스와 비교하여 최고 수준의 비주얼 퀄리티를 보여준다. 시각적으로 좀 더 멋진 가상세계는 플레이어의 몰입도를 상당히 높여주는 요인이 된다. 섬세한 아바타 편집 기능과 다양한 옷, 액세서리로 플레이어의 취향대로 아바타를 만들 수 있다. 여러가지 아바타 패턴을 등록해 놓고 대화 상대에 따라, 기분에 따라 다양한 아바타를 연출할 수 있다. 고급스런 가구와 아파트는 소유욕을 충분히 자극할 수 있으며, 실제 상품과 똑같은 모델의 전시를 통해 3D 쇼핑몰의 기능을 수행할 수 있다.

(2) 게임과 인간을 연결하는 게이트웨이

플레이스테이션 홈의 특성을 한마디로 표현하면 '게임기반 가상세계' 이다. 플레이스테이션 홈과 다른 가상세계의 가장 큰 차이점은 '게임과 사람을 연결하는 게이트웨이' 라 할 수 있다. 지금까지의 게임 플레이어 간의 커뮤니케이션은 소프트웨어 중심으로 분할되어 있었다. 즉, 각 게임 소프트웨어가 창구 역할을 하고 동일한 게임을 즐기는 유저 간에만 커뮤니케이션을 할 수 있었다. 이에 반해, 플레이스테이션 홈은 모든 네트워크 게임에 연결되는 커뮤니티이다. 예를 들어, FPS 등 특정 장르의 게임을 즐기는 플레이어들이 모여서 여러가지 네트워크 게임을 즐기고 커뮤니케이션을 할 수 있다. 또는, 즉석 커뮤니케이션을 통해 다른 장르 게임으로 함께 이동할 수도 있다. 현실의 친구가 게임을 같이 할 때 하는 행동 즉, 사람이 모여서 어떤 게임을 할지 커뮤니케이션을 통해 결

정하고 그 게임을 즐기는 형태의 과정이 플레이스테이션 홈의 환경을 통해 언제든지, 누구와도 이루어지게 된다.

플레이스테이션 홈에는 오프라인에서 타이틀을 구매해야 하는 게임 이외에도 PSN 스토어를 통해 다운로드하여 구매 가능한 게임들과 플레이스테이션 홈에서 플레이 가능한 아케이드게임, 네트워크 플레이 데모 게임 등과 같이 즉시 플레이 가능한 게임이 서비스된다. 플레이어는 현재 속한 플레이어 그룹에서 하고자 하는 게임을 소유하고 있지 않더라도 즉시 구매하여 다른 플레이어들과 같이 게임을 할 수 있다.

다. 가능성과 한계

플레이스테이션 홈은 게임 기반 가상세계로서 '게이머' 라는 특정 대상 집단을 겨냥한 가상세계라는 점에 경쟁력과 차별성이 있다. 그러나, 가정용 콘솔 기반이라는 점은 플레이어 풀에 있어서 매우 큰 약점을 가진다. PC환경인 다른 가상세계에 비해 접근 가능한 총 플레이어수가 너무 적다. 플레이스테이션 홈의 서비스가 시작되리라 예상되는 2008년말 또는 2009년초까지 PS3의 전세계 판매량이 2천만대를 넘기 힘들 것으로 보이며, PSN 플레이어 비율(2008년 2월 현재, 북미 PS3 판매량 400만대, PSN계정 수 330만)을 고려할 때 플레이어 풀은 최대 1,600만명 정도로 예상된다. 실제 주기적으로 네트워크 게임을 즐기는 플레이어 수는 훨씬 적다는 것을 감안하면 PC기반 가상세계와 차이가 상당하다. 때문에 같은 조건이라면 PC기반 가상세계에 비해 상대적으로 매출 규모가 작을 것으로 예상할 수 있다. 그러나, '사람이 중심인 게임 커뮤니티' 로서 플레이스테이션 홈이 제대로 기능할 수 있다면 충분히 그 차이를 극복할 수 있을 것으로 보인다.

앞서 이야기 한 것과 같이, 기존의 게임 플레이어 커뮤니티는 '특정 게임' 에 기반하여 발생하기 때문에 커뮤니티의 존재는 플레이어가 다른 게임으로 이동하는 것을 막는 역할을 한다. 플레이어가 다른 게임으로 이동하는 것은 곧 커뮤니티의 해체를 의미한다. 지속적으로 매출이 발생하는 온라인게임의 경우, 커뮤니티의 존재는 긍정적인 요소로 작용하지만, 타이틀의 일회성 판매가 매출의 중심이 되는 콘솔게임의 경우, 게임의 라이프 타임이 늘어나 전체 콘솔 타이틀 판매량 측면에서 보면 부정적 요소로 작용한다. 플레이스테이션 홈은 이러한 상황을 개선시킬 가능성이 있다. '사람이 중심인 게임 커뮤니티' 가 제대로 형성된다면 플레이어 그룹의 일부가 특정 게임을 구매할 경우, 그룹 전체 혹은 일부가 같은 게임을 구매하여 같이 즐길 가능성이 높다. 이로 인해, 발생하는 게임 판매량의 증가 효과가 얼마나 될 것인지는 아직 알 수 없으나 충분히 기대할 만하다.

Mircosoft는 자사의 콘솔 게임기인 Xbox 360의 네트워크 서비스인 XBLA(Xbox Live Arcade)에서 철저하게 관리하여 고급 콘텐츠만 서비스하는 정책을 취했기에 품질 관리는 제대로 이루어졌으나, 최근 2년 동안 100여 개의 게임 밖에 나오지 않았다. XBLA가 성공하기 위해서는 플레이어 제작 콘텐츠의 관리 수준을 적절히 유지하여 다양하고 창조적이면서도 구매할 가치가 있는 콘텐츠가 지속적으로 서비스되도록 하는 것이 중요하다.

제 3 절 크로스 플랫폼

1. 크로스 플랫폼 게임개발 환경

1) 크로스 플랫폼의 정의

크로스 플랫폼이란 하나의 게임콘텐츠를 다양한 플랫폼(PC, 비디오, 모바일, 아케이드 등)의 제품으로 제품을 내놓는 것을 말한다. 특정 플랫폼 전용 게임을 만드는 것보다 크로스 플랫폼으로 개발하는 것이 더 이익이 된다. 이전에 캡콤(CAPCOM)이 One Source, Multi Use를 슬로건으로 내세웠다면, 크로스 플랫폼은 One Construct, Multi Platform을 지향한다.

위키피디아에 따르면, 크로스 플랫폼 또는 멀티 플랫폼이란 컴퓨터 프로그램, 운영체제, 컴퓨터 언어 및 프로그램 언어나 다른 소프트웨어들을 다양한 플랫폼에서 작동하도록 만드는 것이다. 예를 들어, 크로스 플랫폼 어플리케이션은 MS Windows나 Linux, Mac의 아키텍처에서 동시에 운영되는 것이다. 다시 말해서 모든 플랫폼에서 작동하거나 적어도 2개 이상의 플랫폼에서 작동하는 것을 말한다.

이러한 관점에서 하나의 어플리케이션이 다양한 플랫폼에서 작동하는 것은 개발사 입장에서나 사용자 입장에서 모두 이득이 되는 것이다. 개발사는 더 많은 판매 기회를 얻게 되고, 사용자 입장에서는 자신의 플랫폼을 지원하는 다양한 어플리케이션을 사용할 수 있기 때문이다.

2) 크로스 플랫폼의 프로그래밍 툴킷

사실 프로그래밍은 어디서나 해도 된다. 텍스트 편집기라면 어디나 관계가 없는 것이다. 이렇게 만들어진 소스를 컴파일러나 인터프리터에서 작동시키면 작동하는 것이다. 하지만, 프로그래밍 툴킷이 필요한 이유는 프로그래밍의 생산성 증가와 품질 향상에 기여하기 때문이다.

이러한 크로스 플랫폼 프로그램밍을 할 수 있는 툴은 다음과 같은 것이 있다.

- Simple DirectMedia Layer
- Cairo
- ParaGUI
- wxWidgets
- Qt
- GTK+
- FLTK
- Mozilla
- molib
- fpGUI
- Tcl/Tk
- XVT

3) 크로스 플랫폼의 환경들

크로스 플랫폼 어플리케이션은 물론, 상용 IDEs를 이용하여 개발할 수도 있다. 또는, 소위 RAD 툴을 이용할 수도 있다. 수많은 개발환경에서 크로스 플랫폼을 지원하는 어플리케이션을 제작하고 배포하는 것이 가능하다.

- Eclipse
- IntelliJ IDEA
- NetBeans
- Omnis Studio
- Runtime Revolution
- Code::Blocks
- Lazarus
- REALbasic

하지만, 실질적으로 위의 경우는 단지 크로스 플랫폼을 지원하는 '개발 환경'이다. 그러나, 실질적으로 우리가 관심을 가지는 것은 크로스 플랫폼을 지원하는 '게임개발 환경'이다. 이러한 게임개발 환경에는 다음과 같은 것이다.

가. XNA

MS에서 개발하였다. 개발 환경은 Visual Studio 2005 C# Express Add-on 방식이다. 전체적으로 프레임워크 형태이며 PC와 Xbox 계열의 게임을 동시에 개발할 수 있다. DirectX를 기반으로 하여 DirectX 개발자라면 쉽게 접근할 수 있는 장점이 있다.

나. PhyreEngine

소니에서 발표한 크로스 플랫폼 엔진으로 PC와 PlayStation 계열의 게임을 동시에 개발할 수 있다고 한다. 아직 발표 초기이며 XNA에 비해 불편한 감이 많고, 국내 개발사들의 반 SONY 영향으로 이 엔진이 성공하기 위해서는 편의 사항이 보완되고 한국 개발사에 대한 지원이 뒤따라야 할 것으로 보인다.

크로스 플랫폼을 위한 운영환경에서는 스크립트 언어나 버추얼 머신을 통한 운영 환경이어야 한다. 각 머신 코드는 플랫폼마다 다르기 때문에 이를 통합할 수 있는 버추얼머신 환경이 필요한 것이다. 하지만, 이것은 컴파일에 의해 나타난 네이티브 실행 코드에 비해 수행 속도에 큰 페널티를 가진다. 수행 성능은 게임에서 큰 이슈다. 현재의 대부분의 게임들은 고퀄리티의 그래픽과 방대한 양의 데이터를 다룬다. 그러나, 현실적으로 성공한 게임(스타크래프트, 카트라이더, 카운터스트라이크 등)들이 그런 게임들이었는지는 다시 한번 생각해봐야 할 문제이다.

XNA나 PhyreEngine 모두 소형 개발사나 아마추어 개발자를 타켓으로 하고 있다. 아직은 개발 중인 버전들이라 성능이 다소 떨어지며, 소형 게임 및 캐주얼 게임에 적합하지만, 보다 성능이 낮은 모바일게임이나 휴대용 게임 분야에서는 향후, 하드웨어의 성능 개선과 크로스 플랫폼 환경의 개선을 통해 차후 대형 게임의 제작도 가능할 것이라 예상된다.

이미 국내 거대 개발사들(엔씨소프트, 넥슨, 웹젠 등)은 비디오(콘솔)게임 개발을 진행 중이며, 크로스 플랫폼이라는 시대적 흐름에 빠르게 대응하고 있다. 하지만, 지금은 단지 하나의 게임을 다양한 환경에서 돌아갈 수 있도록 만드는 단계에 지나지 않는다.

2. 크로스 플랫폼 게임개발 사례

1) Console 환경에서 PC 환경으로

가. Gears of War

에픽사(언리얼 엔진 제작사)가 만든 TPS

(FPS + Strategy) 게임으로 Xbox360을 지금의 위치에 올려놓은 Xbox360의 킬러타이틀 중 킬러타이틀이다. 많은 매니아가 있을 정도로 초대형 게임이며, 에픽의 기술력을 보여준 게임이다. 시나리오는 가까운 미래의 암울한 시대에 로커스라는 종족이 Sera라는 행성에서 인류를 멸망시키려 하나 주인공이 막는 내용이다. 뛰어난 그래픽과 단순한 인터페이스, 뛰어난 액션성이 특징인 게임이다.

많은 매니아들이 게임 포털에 글을 올렸고, 확장팩이 나올 때마다 게임계를 흔들고 있다. 현재 2008년 후반에 Gears of War 2가 발매될 예정이다. 에픽사는 이 게임을 PC 환경에서도 작동할 수 있게 다시 제작하였다. 물론 PC 버전은 Xbox 만큼 큰 흥행을 기록하지는 못하였으나 이를 통해 Xbox360 플레이어와 PC 플레이어가 함께 게임을 하는 재미있는 환경을 연출하였다.

Gears of War

나. 헤일로

액션 슈팅게임으로, 역시 Xbox360 이전에 Xbox를 빛낸 게임 타이틀이다. 벌써 3번째 후속작이 나왔으며, 이 후속작이 PC로 구현되어 게이머를 즐겁게 하였다. 코버넌트라는 외계종족이 인류를 이교도로 단정하고 인류를 말살하려 한다. 이에 지구 어딘가 있는 '헤일로'라는 전설의 무기를 둘러싼 전쟁을 표현한 게임이다. MS GameStudio 게임이다. Xbox의 성능을 최대한 활용하여 만든 Xbox 킬러타이틀 중 킬러타이틀이다.

헤일로 3

2) PC 환경에서 Console 환경으로

가. 커맨드앤컨커3

Westwood는 XT 시절의 Dune부터 지금의 Command and Conquer 3까지 전략시뮬레이션 제품을 만든 회사이다. 커맨드앤컨커3는 GDI와 NOD의 타이베리움이란 광물을 둘러싼 전쟁을 그린 게임이다. Command and Conquer는 PlayStation으로도 크로스 컨버팅을 한 작품이다. 콘솔의 주요 타켓이였던 FPS와 RPG를 제외한 다른 장르 게임을 컨버팅하여 주목을 받았으나 흥행에는 성공하지 못하였다.

Command and Conquer 3

나. Call of Duty 4

Activision사에서 만든 FPS 게임이다. 처음에는 펜티엄 3~4 시절 PC게임으로 출시하였으나 Call of Duty 4는 Xbox360과 PC로 함께 출시한 게임이다. 2차 대전을 배경으로 하고 있으며, Band of Brothers 라는 소설과 영화(TV 시리즈)를 기반으로 하고 있다. 뛰어난 시나리오와 연출력이 돋보이는 게임이다.

Call of Duty 4

3) PC 환경에서 Handheld 환경으로

가. 어스토니시아 스토리

어스토니시아 스토리는 AT 컴퓨터 시절에 손노리에서 만든 국산 RPG 게임이다. 창세기전과 함께 PC용 국산 RPG 게임의 양대 축을 이루는 게임으로 모바일게임과 Handheld 게임으로 컨버팅이 되었다.

어스토니시아 스토리

주인공 로이드가 지팡이를 찾아가면서, 동료들을 만나고 함께 떠나는 이야기이며, P-Man이라는 패스워드 질문과 수많은 숨겨진 이야기를 즐길 수 있는 게임이었다. 플랫폼에 따라 게임의 외형은 변하였으나, 기본적인 틀은 비슷한 게임이다.

3. XNA 소개

1) 프레임워크

가. 프레임워크의 필요성

프레임워크는 다른 소프트웨어를 개발할 수 있는 기본 구조를 말한다. 지원 프로그램, 라이브러리, 언어, 다른 소프트웨어 구성 요소들을 엮어주는 소프트웨어 등을 포함한다. 쉽게 말해서 프레임워크는 소프트웨어의 구조이다. 프레임워크를 사용하면 다음과 같은 장점이 있다.

첫째, 초기 개발 시 반복되는 작업을 줄일 수 있다.

〈표 5-2-3-01〉 프레임워크의 종류

프레임워크 환경	종류
Web (MVC)	- Ruby on Rails (php)
	- Velocity
	- FreeMaker
	- Turbine
Persistent/ORM	- iBATIS
	- Hibernate
Configuration	- Obix
	- Kilim
Testing	- JUnit
	- CUnit
Logging	- Log4J
Security	- Acegi Security
종합	- JBOSS
지원	- CVS
	- SVN
	- ANT

둘째, 지원 라이브러리나 구성요소들을 제공함으로써, 플랫폼 변환을 용이하게 한다.

나. 프레임워크의 종류

잘 알려진 프레임워크로는 다음과 같은 프레임워크가 있다.

다. 좋은 프레임워크의 조건

1 지속적인 지원 : 소프트웨어의 변화에 따라 지속적인 지원이 가능해야 한다.
2 확장성이 좋을 것 : 여러가지 모듈/소켓 등을 연결하여 확장할 수 있어야 한다.
3 사용성이 좋을 것 : 프레임워크는 '기본' 틀로서 소프트웨어 개발에 큰 영향을 미치기 때문에 사용이 쉬워야 한다.
4 플랫폼으로부터 독립성 지원 : 프레임워크를 사용하는 이유는 여러가지 환경으로부터 개발을 분리하기 위함이기도 하다. 개념적인 프레임워크의 조건에 따라 실질적인 구현은 다르더라도, 같은 형태를 이루도록 플랫폼으로부터 독립성을 지원하여야 한다.
5 많은 자료 : 지속적인 지원, 사용성 등 위의 조건을 만족하기 위해서는 많은 자료가 지원되어야 한다.

라. XNA 프레임워크

XNA의 프레임워크는 〈그림 5-2-3-01〉과 같다.

MS에서 공개한 XNA의 프레임워크는 DirectX 게임에서 포함하는 많은 부분을 모듈화하여 프레임워크별로 분류하고 있다.

1 작성 코드 : 실제 개발자가 만들어야 하는 코드 부분이다.
2 작성 콘텐츠 : 실제 개발자가 만들어야 하는 콘텐츠(그림, 음악 등) 부분이다.
3 컴포넌트 : XNA 및 XNA Game Creator 에서 지원하는 컴포넌트들이다.
4 시작 킷 : MS에서 XNA 시작 킷으로 지원하는 툴, 예제, 리소스 들의 집합이다.
5 어플리케이션 모델 : XNA 프레임워크의 Game 클래스가 제공하는 큰 틀이다. 크게 Update, Draw, Initialize가 있다. 각각의 게임에 맞게 오버라이드하여 사용하면 된다.
6 콘텐츠 파이프라인 : 게임에서 사용하는 자원들을 각 파일 포맷에 따라 관리하고 유지하는데 사용되는 모듈이다.
7 Graphics 라이브러리 : XNA 프레임워크 Graphics 라이브러리는 Low-Level 자원을 로딩하고 렌더링하는데 사용된다.
8 Audio 네임스페이스 : Audio 네임스페이스는 오디오 파일들을 재생하는 함수와 클래스를 제공한다.
9 Input 네임스페이스 : Input 네임스페이스는 키보드, 마우스, Xbox360 컨트롤러 장치를 사용하는 함수와 클래스를 제공한다.
10 Math 라이브러리 : 벡터나 메트릭스를 다루는 함수와 클래스이다.
11 Storage 네임스페이스 : Storage 네임스페이스는 파일을 읽고 쓰는 것을 담당하는 클래스를 제공한다.
12 Networking 모듈 : 게임에서 네트워크와 관련된 부분을 담당하는 모듈이다. 이전에는 하위에 Gamer Service가 포함되었으나, 2.0 부터는 Gamer Service가 독립되었다.
13 Gamer Service 네임스페이스 : Xbox LIVE와 관련된 기능을 담당하는 모듈이다. 이곳에는 LIVE 관련 GUI 등이 포함되어 있다.
14 DirectX SDK : XNA는 DirectX를 기반으로 한다.
15 XDK : XNA SDK

〈그림 5-2-3-01〉 XNA 프레임워크

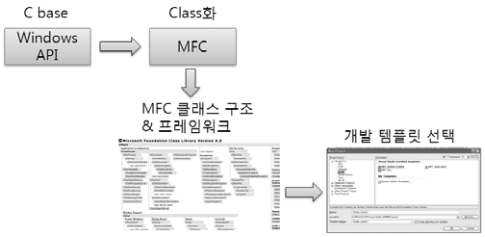
16 .NET Framework : XNA 프레임워크는 .NET 프레임워크를 기반으로 한다.

17 .NET CF(Compact Framework) : 실제 배포 버전에는 .NET CF를 기반으로 한다.

마. 프레임워크 환경과 개발 템플릿

프레임워크가 초기 반복되는 작업을 줄인다면, 어디까지 프레임워크가 지원하고 우리는 어떻게 이를 활용해야 하는가? MS의 경우, Windows API를 Class Library화하여 프레임워크화한 MFC가 있으며, Visual Studio에서는 MFC를 사용하기 위한 환경을 제공하였다. 이것이 바로 개발 템플릿이다.

Windows 어플리케이션을 개발할 때에 사용했던 API를 클래스화하고, 이를 프레임워크화하여 개발 템플릿을 선택함으로써 개발이 가능하다.



Windows API와 MFC 그리고 개발 템플릿

2) XNA를 통한 게임개발의 적용 가능 범위

가. XNA 개발 환경

XNA의 개발환경은 마이크로소프트의 XNA Game Studio라는 통합 환경에서 개발하게 된다. 이곳에는 게임개발을 위한 기본 템플릿과 많은 라이브러리, 도움말 등을 지원하고 있다. 개발 환경에 대한 그림은 아래와 같다.

XNA Studio의 개발 환경

XNA Studio의 개발 환경은 Visual Studio 2005 C# Express를 기반으로 하고 있다. 그 아래 XNA 프레임워크가 동작하며, 이 XNA 프레임워크는 .Net 프레임워크와 .Net Compact 프레임워크를 기반으로 동작하고, 이 두 프레임워크는 윈도우와 Xbox360에서 작동하도록 만들어진 프레임워크이다.

나. XNA와 개발 템플릿

Visual Studio의 개발 템플릿은 XNA에도 마찬가지로 정의되어 있다. 프레임워크화하여 직접 작성해도 되나, 기본적으로 마우스 선택 몇 번으로 게임의 뼈대를 만들어준다.

XNA의 개발 템플릿

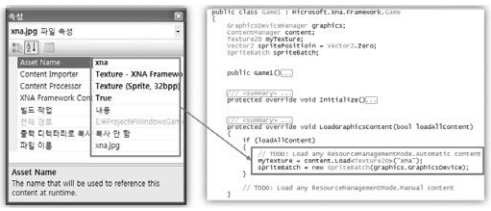
개발 템플릿에서 Windows Game, Window Game Library, Xbox360 Game, Xbox360 Game Library 중 선택하면 된다. Windows Game의 경우, 윈도우즈 게임을 만들 때 사용한다. Xbox360 Game은 Xbox360의 게임을 개발할 수 있게 한다. Library의 경우, 게임에서 사용할 라이브러리를 만드는 개발 템플릿이다. C/C++에서의 LIB, DLL과 같은 것이라 생각하면 된다. 추가적으로 2개의 Starter Kit이 있다. 이것은 예시 코드까지 나오는 환경이다. 이렇게 개발 템플릿을 선택하면 다음과 같이 코드가 자동으로 만들어진다.

개발 템플릿을 선택하고 나온 자동 코드

만들어진 소스는 그리 복잡하지 않다. 하나는 Program.cs 파일이고 나머지 하나는 Game1.cs 파일이다. Program.cs 파일은 단지 Game1.cs 파일을 호출하는데 사용한다. 언어의 엔트리 포인트가 정의된 곳이다. Game1.cs 파일 내부가 프레임워크에서 사용자가 직접 코드를 작성해야 할 부분이다. 코드를 읽어보면 XNA 프레임워크 클래스를 상속

받음을 알 수 있으며 이곳에서 몇몇 필수 함수를 오버라이드(변경)하여 게임을 만들 수 있다.

이렇게 개발 템플릿으로 불린 게임소스를 변경하고 싶을 때에도 쉽게 변경할 수 있다. 그리고 VS의 속성 탭을 이용하여 쉽게 게임을 개발할 수 있다. 예를 들어 'xna.jpg'를 부르고 싶다면 다음과 같이 하면 된다. 이렇게 하면 아래와 같이 코드가 생성된다. 물론 필요에 따라, 직접 작성하는 것도 가능하다.



리소스 로드의 예시

다. XNA의 단점 및 보안

기본적으로 XNA는 DirectX의 대부분의 기능을 구현할 수 있다. 하지만, 기반이 C#이며, JAVA와 같은 인터프리터 방식을 기본으로 하기 때문에 C/C++에 비해 느린 것이 사실이다. MS의 공식적인 입장으로는 2.65배 정도 느리나, 이것은 차후 XNA의 개선을 통해서 점차 빨라질 것으로 예상된다.

C#의 형태이기 때문에, 인터프리터의 모든 소스가 그러하듯, 운용을 위한 VM이 필요하다. 윈도우의 경우, .NET 프레임워크가 반드시 설치되어야 하며 이진 코드로 만들기 위해선 반드시 컴파일을 하여야 한다. 하지만, Xbox360의 경우, 다운받으면 그 즉시 컴파일을 하기 때문에 특별히 문제가 될 것은 없다.

또한, 초기 실행시 VM을 구동해야 하기 때문에 지연이 발생한다. 그리고, 아직은 개발 버전이기 때문에 재배포가 힘들고, 개발 언어도 C#이라는 것이 단점이다(기존 게임개발자는 보통 C/C++ 사용자이다).

라. XNA의 주요 타켓과 개발 범위

현재 XNA의 주요 타켓은 아마추어 게임개발자 또는 소형 개발사이다. XNA를 통해 이들이 쉽게 다양한 게임을 개발할 수 있도록 지원

5

〈표 5-2-3-02〉 게임 템플릿으로 만들어진 Game1.cs 의 필수 함수

오버라이드 함수	설명
Initialize	게임이 실행되면 단 한번 수행되는 함수. 초기화 함수로 생각하면 된다.
LoadGraphicsContent(1.1) LoadContent(2.0)	Initialize 함수가 불리고 난 후, Initialize 함수가 종료하기 직전에 불리는 함수. 게임에 사용될 리소스(그림, 음악 등)를 부르는 곳이다. 2.0에서는 LoadContent로 함수명이 변경되었다.
UnloadGraphicsContent(1.1) UnloadContent(2.0)	게임을 종료하면 불리는 함수. 게임에 사용된 리소스(그림, 음악 등)를 메모리에서 해제시키거나 게임의 종료시 사용할 수 있다. 2.0에서는 UnloadContent로 함수명이 변경되었다.
Update	게임 타이머(정기적인 이벤트)에 의해 호출되거나 사용자의 이벤트에 의해 불리는 함수. 정확하게 WinProc와는 다르지만, 유사하다고 보면 될 것이다. 입력과 게임의 내부 상태를 갱신할 때 사용하는 함수이다.
Draw	그리는 함수이다.

함으로써 MS의 Windows 또는 Xbox360의 콘텐츠를 넓히려는 전략이 숨어있는 것으로 보인다. 또한, XNA Creator Club에 가입하고 참여함으로써 LIVE Arcade라고 불리는 Xbox360의 다운로드 게임 환경을 통해 게임을 제공할 수도 있다.

다시 말해서 지금은 단순한 아케이드게임 또는 가벼운 게임 정도가 개발 가능하며, 레이싱 게임, 우주 비행 시뮬레이션, 아케이드, 퍼즐 게임 등에 적용할 수 있다.

XNA로 개발된 게임들

4. 향후 전망 및 결론

한편에서는 XNA도 DirectX처럼 5.0부터 실제 게임개발에 적용이 가능할 것이라는 시각도 있다. 또한, 너무 느리기 때문에 현재 고퀄리티의 게임을 XNA로 개발하는 것은 당분간 힘들 것이라는 의견도 있다. 프레임워크로 기반 환경을 다져놓았다고 하지만, 실제로 과거 DirectX도 COM 기술을 이용하여 고정된 환경을 제공하려 했으나 실상은 그러지 못하였다. 시간이 지남에 따라 너무 많은 변화가 일어났고, 과거의 코드를 현재에 그대로 사용하긴 힘들 것이다. 그러나, MS의 추진력을 생각하면 이번 XNA는 DirectX와는 다를 것으로 예상된다. XNA 3.0에서는 휴대용과 모바일게임까지 영역을 넓힐 예정이다. 그 이후에 점차 속도와 메모리 사용에 최적화한다는 로드맵을 가지고 있는 듯하다.

한편, 비디오 콘솔 시장에서 슈퍼페미콤, 메가드라이브 등이 초기 시장을 이끌었다면, DreamCast, Playstation, Playstation2 등이 2차 대전에 참여하였다. 1차 대전에서는 2D 게임이 대세였고 그 결과 마지막 승자가 슈퍼페미콤이였다면, 2차 대전은 3D 게임이 중심이었고 파이널판타지를 필두로 수많은 킬러타이틀을 제공한 소니의 PS2가 최종 승자였다. 이제는 바야흐로 차세대 비디오(콘솔) 게임기를 중심으로 3차 대전이 시작되었다. 핸드헬드(휴대용) 게임의 경우, 1차 대전의 승리는 일본이었다. 게임보이, 게임보이 어드벤스 등이 어린이를 대상으로 큰 성공을 거두었다. 2차 대전 역시 일본이 주도하였는데 PSP, NDS 게임들이 시장에서 좋은 반응을 얻었다.

그리고, PC게임 시장에서도 3차 대전을 앞두고 있다. 1차 대전은 미국 시장이 대세였으며, 많은 게임들이 XT, AT 기종으로 나왔다. 2차 대전 역시 서구권 시장이 대세였다. 웨스트우드, 오리진, 엑티비전, 아타리, 레릭, 바이오웨어, 아이언로어, 블리자드 등 수많은 게임 제작 및 퍼블리싱 회사들이 경쟁하였다. Dune, Command & Conquer, 울티마, 홈 월드, 히어로즈 오브 마이트 앤 매직, 디아블로, 발더스 게이트, 네버윈터나이츠, 스타크래프트, 리니지, 워록, 스페셜포스, 마기노비, 퀘이크 등 PC

게이머에게 익숙한 수많은 게임들이 등장하였다. 2차 대전에서의 특징이라면 멀티플레이 게임 또는 온라인게임이 등장하여 진화를 거듭했다는 점이다.

그렇다면, 게임산업의 3차 대전은 무엇을 의미할까? 이미 게임 업계는 2D와 3D로 제작된 수많은 장르의 게임이 넘쳐나고 있다. 그러나, 각 시장마다 특징이 있다. 콘솔 게임기는 온라인게임 기능이 약하고, PC는 콘솔 시장과 게이머의 성향이 많이 다르다. 3차 대전은 플랫폼별 특징을 살린 크로스 플랫폼에 기반한 컨버전스가 중심이 될 것으로 보인다.

3차 대전은 이제 막이 올랐다. 아직은 국지전에 불과하여 작은 전투에 불과하지만 머지않아 대규모 전투가 시작될 것이다. 우리나라 게임 업계도 3차 대전에 이미 뛰어들고 있다. 3차 대전의 승자가 되기 위해서는 콘솔, PC, 핸드헬드를 넘나드는 크로스 플랫폼은 필수적이다. 또한, 크로스 플랫폼을 지향하는 툴이 보급되면서 다양한 플랫폼에 도전할 수 있는 길도 열리고 있다. 예를 들어, XNA를 통해 PC게임과 Xbox360 게임을 동시에 개발할 수 있다. 특히, 온라인게임 시장에 편중되어 성장에 한계를 느끼고 있는 국내 게임업계의 입장에서 크로스 플랫폼은 새로운 대안이 될 수 있을 것이다.

제 4 절 발전된 온라인게임 엔진의 기능

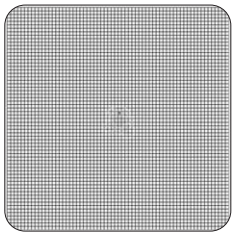
1. 최고의 온라인게임과 엔진

2004년 11월 등장한 Blizzard의 "World of Warcraft"(이후 WOW)는 전세계 온라인게임 시장을 강타하였고, 기존 온라인게임에 대한 시각을 바꾸어 놓는 계기가 되었다.

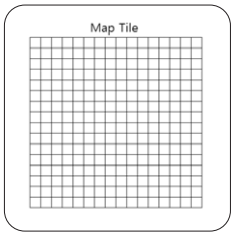
WOW의 등장 시점을 기준으로 국산 온라인게임을 나누어 보면, 그래픽적인 면에서 WOW 이전에는 캐릭터의 정교함과 화려함에 치중하였던 반면, 그 이후에는 배경의 자연스러운 빛과 색의 조화에 중점을 두고 있는 형태로 변해가고 있다. 또한, 게임의 특성면에서는 적당한 크기의 맵에서 액션을 강조하는 형태에서, 거대한 대륙을 자유롭게 이동하면서 다양한 콘텐츠를 즐기는 게임 형태로 변모해 가고 있다. 이런 변화에 따라 다양한 프로그래밍 기법들이 대두되고 있는데 예를 들면, 작은 규모의 맵 안에 있는 건물들과 장식물들이 이동 중 장애물로서만 의미를 갖고 있었던 상황에서, 건물 하나 하나에 의미를 부여하고 다층 건물로서의 기능을 수행하도록 하는 지금까지와는 전혀 다른 프로그래밍 기법들이 적용되고 있다.

WOW가 나온 지 벌써 3년 이상의 시간이 흘렀지만 WOW와 같은 규모의 게임은 아니더라도 WOW의 엔진이 갖고 있는 장점들을 이용한 온라인게임조차 아직 나오지 않은 상황이다.

여기서 WOW와 같이 드넓은 대륙, 그 안에서의 거대한 도시들, 크고 작은 건물들과 장식물 등을 이용하여 실제 살아 숨쉬는 대륙을 표현할 수 있는 기법들과 WOW의 엔진을 능가하기 위해 부가적으로 지원되어야 할 기능들에 대해 살펴보자.



월드의 구성 (Map Tile : 64×64개)



맵 타일 (Map Tile : 16×16개)

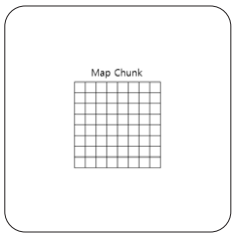
2. 온라인게임 엔진의 주요 기능

1) 지형

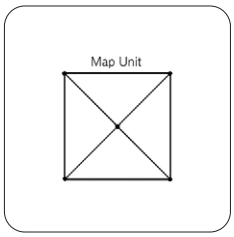
약 가로 32m, 세로 32m 이상의 지형을 표현할 수 있다고 한다면, 백그라운드에서 지속적으로 로딩을 하면서 대륙을 이동할 수 있기에, 이론적으로 무한대에 가까운 대륙을 표현할 수 있다. 먼저, 약 가로 세로 32m의 지형을 표현하는데 있어 필요한 용어들을 정의해 보자.

가. 월드의 구성

하나의 월드는 파일 단위인 Map Tile이 최대 64개×64개 = 4,096개로 구성되며, 각 Map Tile 파일은 설정 단위인 Map Chunk 16개×16개로 구분된다. 또 하나의 Map Chunk는 8개×8개의 Map Unit으로 구성된다.



맵 청크 (Map Chunk : 8×8개)



맵유닛 (Polygon 4개)

나. 맵의 구성

맵 타일(Map Tile) 파일은 일반적으로 높이 값을 갖고 지형을 표현하는 높이 맵(Height Map) 형태의 자료 구조를 갖고 있으며, 지형의 바닥과 그 안에 존재하는 구조물과 장식물들을 표현하기 위한 정보들을 갖고 있다.

높이 지형

〈표 5-2-4-01〉 용어 정의

용어	설명	구성
World	대륙 (32,768km * 32,768km)	Map Tile 64개 x 64개
Map Tile	맵 파일 저장 단위 (512m * 512m)	Map Chunk 16개 x 16개
Map Chunk	지형 정보 저장 (32m * 32m)	Map Unit 8개 x 8개
Map Unit	그려지는 단위 (4m * 4m)	

맵 타일 파일에 포함되어 있는 자세한 정보들에 대해 살펴보자.

〈표 5-2-4-02〉 맵 타일의 구성 요소

목록	내용
텍스처	사용된 텍스처 목록
구조물	사용된 구조물 목록
장식물	사용된 장식물 목록
지형 높이 값	지형의 형태를 구성하는 높이 값
다중 텍스처 적용	다중 텍스처 적용을 위한 적용 값
그림자	구조물, 장식물에 의한 그림자
물의 높이 값	설정된 물의 높이 값

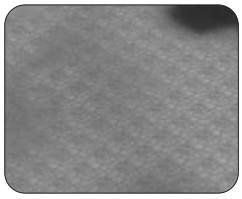
하나의 맵 타일 파일에 포함되어지는 정보는 지형의 형태를 구성하는 높이 값들과 타일 텍스처(Tile Texture)를 다중으로 적용하기 위한 텍스처 적용 값, 그림자 처리를 위한 그림자 값, 각각의 위치에 구조물들과 장식물들을 설치하기 위한 값들로 구성되어 있으며, 강물이나 바다와 같이 특정 위치에 설정된 물들을 위한 값들이 포함되어 있다.

일반적인 높이 맵(Height Map) 형태의 자료구조는 쿼드 트리(Quad Tree) 자료구조로 관리되고 화면으로 랜더링 되는데, 시야에 높은 지형들이 가로막고 있을 경우, 뒤에 존재하는 지형들이 랜더링 되지 않기 위해, 미리 계산을 수행하도록 하여 랜더링 시간을 절약하도록 한다.

다. 다중 텍스처

정교한 지형을 만들기 위해서는 높이 맵(Height Map)의 간격이 조밀하거나 지형 텍스처가 정교하게 설정되어 있느냐에 따라 달라진다. 이중에서 같은 간격의 높이 맵을 사용하면서도 정교한 지형을 제작하는 법에 대해서 살펴보자. 일반적으로 지형은 다중 텍스처 적용 방식을 취하고 있으나 그 적용방법에 따라서 크게 차이가 나게 된다.

다중 텍스처 적용 1

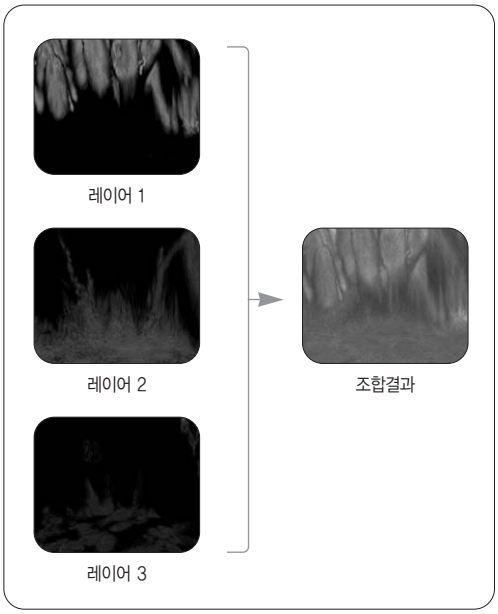


다중 텍스처 적용 2

많은 수의 레이어(Layer)를 사용하여 텍스처를 설정하는 것과 적은 수를 사용하여 설정하는 것은 큰 차이를 보여주는데, 같은 수의 레이어를 사용하여 텍스처를 설정한다면 얼마나 촘촘히 텍스처들을 다양하게 적용하고 있느냐에 따라 텍스처가 반복되어진다는 느낌이 줄어들게 된다. 이런 동일한 방법을 적용한다고 해도 지형이 또렷하게 그려진다는 느낌과 뿌옇게 번진다는 느낌이 드는 경우가 있는데 이는 여러 텍스처가 어떻게 섞이도록 설정하느냐에 따라 크게 좌우된다.

일반적인 타일 텍스처(Tile Texture) 적용 방법으로는 부드러운 브러시(Smooth Brush)를 사용하여 적용시키고 있으나, 이 방법만으로 현실감 있는 지형을 구성할 수 없어 이와 다른 다양한 방법들을 적용해 더욱 더 자연스러운 지형을 완성시킨다.

다양한 타일 텍스처의 적용 방법에 따라 자연스러운 지형의 형태가 나오게 된다.



일반적인 브러시와 특수 브러시 적용에 의한 자연스러운 지형

라. 원거리 지형

먼 곳에 떨어진 지형까지 시야가 확보된다는 것은 원거리 지형을 표현해야 한다는 것을 의미한다. 일반적인 방법으로는 랜더링 시에 상당한 부담을 주기 때문에 부하를 줄이면서도 원거리 시야를 확보하고 있는 것 같이 속여주는 것이 가장 현명한 선택이다.

타일 텍스처를 포함하고 있는 근거리 지형에는 일정한 한계가 있다. 저사양에서는 반경 100m 정도, 고사양에서도 반경 300m를 넘지 않도록 하는데 원거리를 바라보고 있는 느낌을 표현하기 위해서는 반경 500m에서 1km 정도를 표현해야 하는 광활한 평원에서 그 진가가 나오게 된다.

이를 위해, 먼저 준비해야 하는 것은 기본 지형 데이터를 이용하여 맵 청크(Map Chunk) 단위의 저해상도 지형을 생성한다. 그런 다음 안개(Fog)의 끝지점부터 안개색으로 저해상도 원거리 지형을 그린 후, 그 위에 타일 텍스처가 포함된 근거리 지형을 그려주게 되면 자연스럽게 근거리 지형과 원거리 지형이 이어지게 된다.

이 작업의 결과를 묘사해보면 먼 산을 바라보면서 계속 향하다 보면 어느새 인가 타일 텍스처가 포함된 선명한 산이 서서히 눈앞에 드러나게 되는데, 이때 원거리 지형에 성이나 기타 원거리 지형의 윤곽에 변화를 줄 수 있는 구조물들이 존재한다면 그것도 같이 표현하는 것이 더욱 더 실감나는 원거리 지형을 표현하는 방법이다.

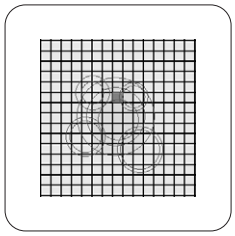
2) 지역 조명

드넓은 대륙을 횡단하다 보면, 설원지역, 화산지역, 사막지형, 초원지역, 밀림지역 등 다양한 환경과 접하게 된다. 이들 각 지역마다 특유의 빛과 색이 존재하게 되는데, 대륙전체를 끊임없이 이동할 수 있는 경우, 그 구분을 정확하

근거리 지형 | 원거리 지형

근거리 + 원거리 지형 | 근거리 + 원거리 + 안개

게 잘라서 표현할 수가 없다. 이를 위해 영향력 범위를 설정해서 자연스럽게 그 영향력을 부가할 수 있도록 각 지역 조명의 분포 범위를 설정한다.



월드에 설정된 조명

〈표 5-2-4-03〉 지역조명의 값들

목록	설명
범위	위치, 반경
하늘	하늘의 높이에 맞는 시간대별 색
물	물의 깊이에 따른 시간대별 색
안개	배경 및 안개의 시간대별 색과 깊이
태양, 달, 별, 구름	하늘의 장식물의 시간대별 색

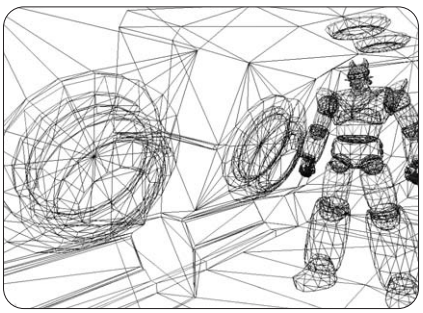
보다 자연스러운 빛과 색의 변화를 유도하기 위해 각 지역의 조명들이 중앙부와 주변부로 나누어져, 집중광원(Spot Light)과 같이 영향력들이 겹쳐지면서 자연스럽게 이어갈 수 있도록 해야 한다. 각 조명들은 해당 위치에서 해당 영역에 영향을 주는 지역조명으로서 하늘, 물, 안개, 태양, 달의 빛과 색을 시간의 흐름에 맞춰 설정할 수 있도록 하여 그 사실성을 극대화한다. 이러한 지역 조명은 단지 그 빛과 색으로서의 의미뿐만 아니라 그 지역에 있는 지형, 구조물, 캐릭터에 영향을 주도록 하여 자연스러운 화면을 연출하게 된다.

3) 구조물

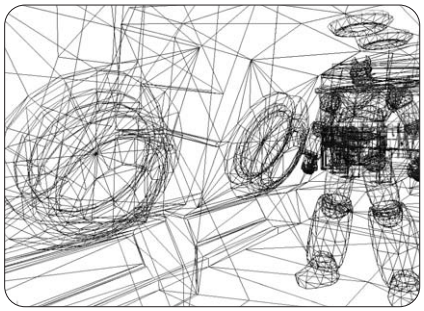
지형의 적절한 위치에 구조물을 위치시켜 원하는 형태의 맵 타일(Map Tile)을 완성시키고자 할 때, 거대한 성이나 수많은 집들이 모여있는 경우, 적절한 처리 없이는 그 부하를 감당할 수 없다.

구조물 밖에서 구조물을 바라볼 경우, 보이지도 않는 구조물의 내부가 복잡하면 복잡할수록 점점 더 많은 부하가 걸리므로 이를 철저하게 걸러낼 필요가 있다. 이런 복잡한 구조물, 가칭 다층 구조물을 처리하기 위해서 일반적으로 포털 엔진(Portal Engine)을 사용하고 있다.

랜더링 화면



포털 On



포털 Off

Shell

포털 Off

그림에서와 같이 포털 엔진의 적용과 미적용에 따라 확연한 차이를 보여주고 있는데, 만약 건물이 아니라 하나의 큰 도시라고 생각한다면 포털 엔진의 미적용 시, 거대한 도시를 표현할 수 있느냐 없느냐를 결정할 정도로 강력한 성능을 나타낸다.

보통의 포털 엔진은 실내 엔진(Indoor Engine)에서 사용되는데, 이를 수정하게 되면 실외 엔진인 지형 엔진과 실내 엔진인 구조물 엔진이 적절하게 혼용되어 처리속도를 극대화시키면서 구조물 내부의 정밀도를 높일 수 있다.

구조물의 실내와 실외를 적절히 처리하기 위해서는 실내와 실외의 중간지대로서 실외에서 구조물을 바라보게 될 경우, 보여지게 되는 부분을 나누어 껍질(Shell) 이라 칭하고, 그 경계에 포털(Portal)을 설치하여 실내, 실외의 모든 부분에서 만족시키도록 하고 있다.

이러한 포털 엔진의 적극적인 활용을 통해, 복잡한 미로나 지하도시 등 지형 엔진이 표현하지 못하는 정교한 지형이나 건축물들을 표현하도록 하여 지형 엔진과의 역할을 적절하게 분담토록 한다.

4) 캐릭터

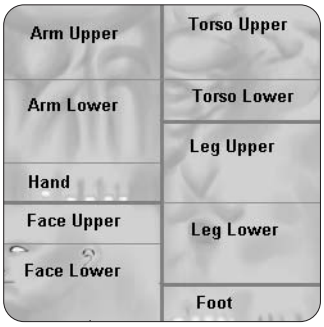
다양한 장착 아이템에 맞춰 캐릭터의 다양한 변화가 가능토록 제작되는데, 이를 구현하기 위한 방법은 크게 두 가지로 나누어진다. 갑옷과 장갑, 신발과 같은 몸의 일부분에 해당하는 부분이 변경될 경우, 첫째, 메쉬(Mesh)와 텍스처(Texture)가 같이 세트로 변경되는 경우, 둘째, 메쉬(Mesh)는 적절하게 변하지만, 공통 텍스처의 일부가 변경되는 경우, 두 가지로 나눠볼 수 있다.

첫 번째, 메쉬와 텍스처를 세트로 제작하는 경우, 다양한 자유도와 미적 극대화를 꾀할 수 있지만 동시에 많은 작업이 병행되어야 하기 때문에 다양성을 부가하는데 있어서 많은 시간과 노력이 들게 된다.

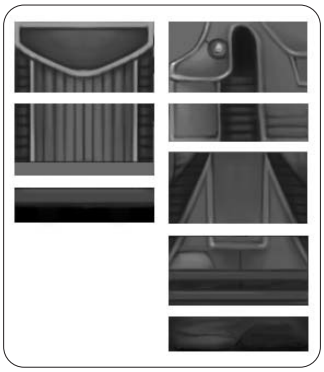
두 번째, 공통 텍스처 조합방식을 이용하는 경우, 약속된 위치의 텍스처를 변경하면 해당 메쉬(Mesh)에 적용되어 다양한 의복류, 갑옷류의 표현이 가능하여 같은 성별의 캐릭터라면 이 작업만으로도 다양한 캐릭터의 표현이 가능하게 되나 비압축 텍스처의 사용으로 압축방식에 비해 약 4배 정도의 메모리 손실이 뒤따르게 된다. 그리하여, 하나의 캐릭터를 표현하는데 있어서 작은 크기의 텍스처(256×256)를 사용하게 되어 정밀도가 떨어지는 단점이 생길 수 있다.

게임의 한 장면을 상상해 보자. 한 마을이나 지역으로 입장할 경우, 같은 종족끼리의 조우 시 모델의 로딩 없이 같은 메쉬(Mesh)를 공유하게 되어 모델 로딩의 부하는 현저하게 줄어들면서 해당 캐릭터의 고유 텍스처 한 장을 로딩하거나 텍스처를 조합하여 한 장의 텍스처를 사용하여 랜더링을 하게 될 경우, 제작하고자 하는 게임의 특성을 완성된 아름다움보다 다양한 조합에 더 큰 의미를 부여한다면 큰 효과를 낼 수 있다.

캐릭터의 기본 텍스처를 바탕으로 놓고 장착된 장비에 따른 각 부분의 아이템 텍스처를 덧



맨몸을 구성하는 기본 텍스처



아이템 텍스처

씌워 하나의 텍스처로 만들어 사용한다. NPC 같은 경우는 실시간 조합이 필요없으므로 적절하게 조합된 텍스처를 미리 만들어 저장한 후 게임에서 바로 사용하도록 하여 수행 속도를 향상시킨다.

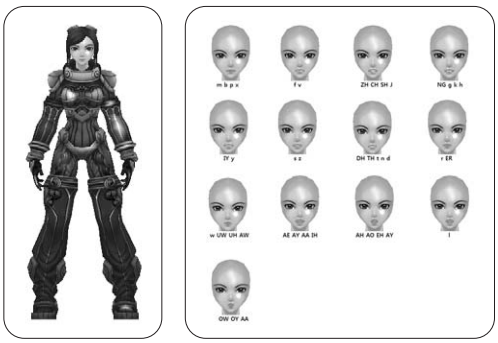
5) 립싱크

점차 많은 게임들이 감정에 호소하는 연출을 요구함에 따라 립싱크 기술이 점차 도입되고 있고, 많은 상용 립싱크, 캐릭터 모션 엔진들이 판매되고 있다. 실시간 립싱크 기능을 지원하는 모션 엔진들과 기존 엔진의 렌더러(Renderer)와 연계를 하다 보면 많은 부분들이 중복되기도 해 성능상 비효율적인 면이 있거나 일정 CPU를 지속적으로 할당해 사용해야 하는 경우가 생기며 음성파일이 존재하는 경우가 발생하여 MMO의 개발 특성 상 빠른 콘텐츠 제작에 있어 장애가 되기도 한다.

실시간 립싱크 기술이 아닌 미리 계산된 립싱크 기술을 활용할 경우, 약간의 정교함을 포기한다면 상용 라이브러리의 도움도 필요없이 Microsoft Speech SDK 5.1 의 Speech

Recognition Interface들을 이용하여 이를 해결할 수 있다. 음성파일을 SDK로 분석을 한 후, 분석된 텍스트 단음의 시작, 종료 시간을 저장하도록 한다. 이를 바탕으로 캐릭터에 설정된 캐릭터 애니메이션 기능을 적극 활용하여 립싱크가 가능하도록 한다.

미리 계산된 립싱크 기능을 사용하게 되면 실시간으로 음성 파일을 분석하는 부하가 줄어들 뿐만 아니라, 성우를 이용한 모든 음성 파일이 게임콘텐츠 제작과 동시에 필요하지 않게 되어 작업의 용이성이 확보되며 지속적인 작업 진행이 가능해진다.



립싱크 세트 (Annosoft LLC Set 1)

6) 스크립트

스크립트는 게임 내에서 특수한 장면이나 퀘스트(Quest)의 연출 시에 가장 많이 사용되고 있는데 스크립트를 이용해 많은 수정 작업을 반복적으로 쉽게 할 수 있고 업무를 분업화할 수 있도록 한다.

퀘스트의 설명이나 상황 설정 등의 기능은 게임 서버와는 상관없이 클라이언트 자체의 기능만으로 캐릭터를 생성하고 행동시키고, 카메라의 이동 등을 수행하게 되는데 일반적인 스크립트는 미리 등록된 함수를 호출하는 형태로만 사용하게 되어 그 이외의 확장된 기능을 위해 범용 스크립트를 이용한다. 범용 스크립트들 중에서 루아(LUA)는 가장 광범위하게 사용되고 있는 스크립트로서 성능과 속도 면에서 다른 스크립트들과 비교해 뛰어나고 적용이 쉬워 게임 코드와의 연계를 더욱 수월하게 한다.

루아는 관련 사이트(www.lua.org)에서 쉽게 구할 수 있으며 수많은 예제와 C, C++ 코드와 연결할 수 있는 많은 자료들이 공개되어 있다.

스크립트

소환하고자 하는 몬스터 셋의 위치와 이름을 설정하고 이동함수를 등록하여 몬스터 소환 후 몬스터를 강제 이동시키도록 하는 스크립트이다. 캐릭터 립싱크들과 같이 이용하면 다양한 연출이 가능하다.

7) 인터페이스

개발시간의 단축과 유저에 의한 인터페이스 설정 및 수정을 지원하기 위해서는 인터페이스의 개발 초기부터 스크립트를 지원할 수 있도

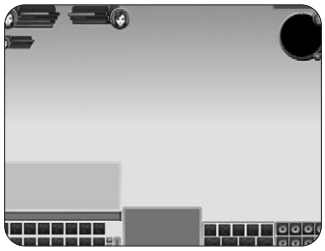
록 구성되어야 한다. 인터페이스 작업의 분업화와 자동화를 위한 인터페이스 툴은 크게 두 가지 기능으로 나누어진다.

첫째, 인터페이스를 구성하게 되는 그림 파일에 UV 값들을 설정하고 이름을 붙인다. 하나 하나의 아이콘에 이름을 붙이는 이유는 스크립트와의 연계를 지원하기 위함이다.

UV 설정

둘째, 게임 내의 화면이나 창을 구성하기 위한 인터페이스 구성 툴로 나누어진다. 각각의 창은 현재 제공되고 있는 기능들을 이용하여 구성하고 위의 UV 설정 툴에서 결정한 이름들을 사용하여 그림을 선택하게 된다.

여기서 설정한 화면은 파일로 저장한 후 추후에 불러들여 수정할 수도 있으며, 게임 내에서 사용하기 위해 소스코드와 헤더파일로 출력하기도 한다. 게임 내에서의 사용은 저장된 인터페이스 파일을 사용하거나 출력된 소스코드를 활용할 수 있으나 둘 다 장단점이 존재한다. 저장된 파일을 사용하게 될 경우, 새로운 기능이나 형식의 변경 시 버전 관리를 해주어야 한다는 단점이 존재하나 설정된 그대로 게임 내에서 쉽게 표현되는 장점이 있다. 하지만, 설정



인터페이스 스크린 구성

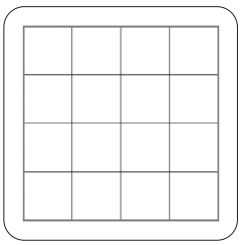
된 그대로의 단순한 기능만으로 게임 내에서 쉽게 구현이 되는 경우가 드물기 때문에, 일일이 설정하기 쉽게 코드 출력을 통한 방법을 사용하는 편이 버전 관리나 차후 유지 보수팀의 접근이 용이한 장점을 갖고 있다.

8) 길 찾기

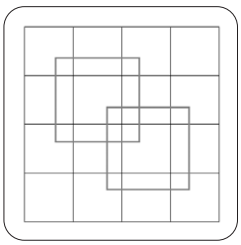
게임 AI의 시작은 길 찾기 엔진에서부터 시작한다고 해도 과언이 아니다. 수많은 인공지능에 대해서 기술하고 있는 참고자료들을 살펴보아도 길 찾기부터 시작하는 경우를 많이 볼 수 있다.

온라인게임의 기반기술 중 하나인 길 찾기를 광활한 대륙에 적용하고자 할 때, 신중히 고려해야 할 사항은 메모리와 효율성이다. 우선, 가로, 세로 약 32km×32km 의 지형을 표현하고자 할 때의 메모리 사용량을 계산해 보자. 하나의 월드에 최대 맵 타일(Map Tile = 512m×512m) 파일이 64개×64개 = 총 4,096개가 필요하게 되므로, 하나의 맵 타일 파일에 길 찾기 정보로서 3MB 정도의 메모리가 필요하다면 4,096개×3MB = 12,288MB = 약 12.3GB, 1MB라 가정한다고 해도 4,096개×1MB = 4,096MB = 약 4GB 라는 엄청난 메모리의 소비를 요하게 된다. 또한, 이렇게 드넓은 지형정

보를 관리하는 방법에 따라 두 가지 경우로 나누어 보면, 약 가로, 세로 32km×32km 지형을 하나의 지형정보로 저장을 하거나 일정한 가로, 세로 개수로 나눈 후 일정지역을 겹치도록 한 후 자료를 관리하는 경우로 나누어 볼 수 있다.



하나의 지형으로 사용하는 경우



일정 개수로 나누어 사용하는 경우

하나의 지형으로 관리하는 경우, 길 찾기가 매번 수행될 때마다 거대한 자료구조를 다루게 되는 단점이 있어 길 찾기 성능이 저하될 수 있다. 하지만, 두 번째의 경우처럼 일정 개수로 나누어 관리를 하더라도 겹쳐지는 만큼의 메모리가 더 낭비가 되므로 길 찾기 성능은 향상될 수 있어도 더 많은 메모리의 소비를 요하게 되는 단점이 있다. 상황이 이렇다면 성능 향상을 위해 일정 개수로 나누어 관리를 하더라도 메모리 사용량을 현저히 감축시키지 않고서는 실제로 적용시키기에 어렵다는 결론에 이르게 된다.

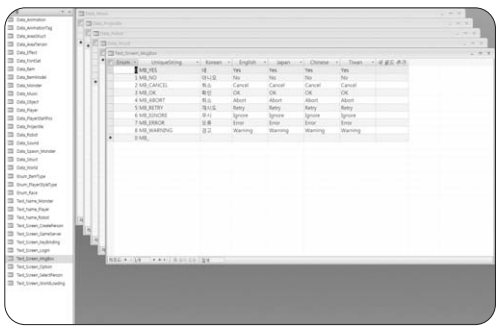
즉, 기존의 네비게이션 메쉬나 기타 자료구조를 활용하더라도 지형의 정보를 얼마나 간소화시키느냐에 따라 실용성이 결정된다. 여기에 추가적으로 지형 정보뿐만 아니라 구조물들의 정보가 같이 포함이 되어야 하기에 길 찾기를 위해 최대한으로 간소화된 자료구조로 만드는 것이 우선이다.

가장 넓은 영역을 포함하고 있는 지형의 정밀도를 Map Unit 단위에서 Map Chunk 단위의 정보로 간소화한 저해상도 지형정보를 이용하여 길 찾기 정보를 생성한다면, 약 가로 32km, 세로 32km 크기의 지형에서도 이전보다 더 적은 부담으로 자료를 관리할 수 있다. 하지만, 저해상도의 정보를 활용한 만큼의 오차를 감수해야만 하는 단점이 있다.

9) 데이터

게임 설정 데이터를 다루는 것은 지속적인 관리와 집중을 요하는 중요한 업무이다. 이러한 데이터를 관리하는 방법에는 전통적인 Text 파일 방법에서 Excel을 이용한 방법으로 발전해 왔다. 하지만, Excel로는 수많은 데이터를 효과적으로 관리하기에 적절하지 않아, Access를 활용한 방법이 사용되고 있는 상황이다.

Excel과 Access는 같은 Microsoft사의 제품으로 데이터의 교환이 손쉬워 쉽게 전환할 수 있으며 Object Data Base를 기반으로 하고 있는 Access의 장점을 살린다면 많은 양의 데이



Access를 사용한 데이터 관리

터의 비교분석이 가능하며 하나의 파일로도 보안설정에 따라 데이터를 관리할 수 있다. 또한, 동시에 Access 파일에 접근하여 작업을 수행하여도 해당 테이블(Table)별로 동기화가 제한되므로 데이터의 안정성이 확보된다는 장점이 있다.

게임 내에서 사용하기 위해서는 각 테이블별로 관리된 자료를 내보내기 기능을 이용하여 텍스트 파일로 저장하도록 한다. Access는 Unicode가 지원되므로 다양한 언어의 지원과 설정이 가능하며, 조건별 선택과 타 테이블의 참조를 통한 선택이 가능하여 설정상의 오류를 최소화할 수 있다.

3. 결론

'World of warcraft' 와 같이 대규모의 리소스가 투입되는 게임을 제작하는데는 많은 시간과 인력이 투입되어야 하므로 제작 시에 어려움이 따른다. 하지만, 실제 게임에 사용되는 엔진 기술은 가장 일반적인 기술들을 극대화하여 적용시켰다는 점에서 그리 어렵지 않게 구현할 수 있는 기술들이다. 일반적으로 완성된 게임 엔진의 구입 후 그 활용기술을 습득하는데 상당한 시간이 소요된다. 따라서, 충분히 자체 기술로서 확보하고 발전시킬 수 있는 방법을 선택하는 것이 다시 한번 온라인게임의 기술 강국임을 확고히 하는 계기가 될 것이다.

게임 그래픽의 발전 방향

- ATI RadeonTM HD 3800 시리즈

■ 소개 Introduction

GPU 테크놀러지 분야에서, Microsoft로부터 그 기술적 수준이 공개된 Microsoft® DirectX ® 10.1은 ATI RadeonTM HD 3800 시리즈에 의해 표현되는 가장 최신의 애플리케이션 프로그래밍 인터페이스(Application Programming Interface, API)이다. 핵심적인 특징은 애플리케이션의 더 나은 호환성을 위해 최신의 셰이더 모델(Shader model)을 포함, 향상된 안티앨리어싱(Anti-aliasing)의 지원, 좀 더 유동적인 데이터 액세스(Flexible data access) 그리고 더 높아진 사양(Specification) 등이다. 이러한 특징들은 실시간 전역 조명효과(Real-time global illumination)와 같은 새로운 기법들을 떠오르게 할 것이며, 상호작용 3D 그래픽(Interactive 3D graphics)의 향후 방향을 정하게 될 것이다.

DirectX10은 API의 등장 이후 가장 중요한 업데이트 중 하나였지만, DirectX 10의 사양이 최종적으로 승인된 후 몇가지 한계가 드러나 DirectX 10.1은 이를 처리하는 진보적인 업데이트로 표현된다. DirectX 10.1의 지원은 2008년 초 Windows Vista™ 운영체제의 서비스팩 릴리즈와 함께 이루어질 예정이다.

많은 기간 동안, ATI는 DirectX 기술 발전의 선두에 있었으며 새로운 그래픽 형태를 정의하고 충족시키기 위해 Microsoft와 함께 적극적으로 노력해 왔다. 2006년도의 ATI와 AMD의 합병은 이러한 노력의 계승일뿐 아니라, GPU, CPU 그리고 시스템 칩셋 간 플랫폼 수준의 시너지를 위한 새로운 가능성을 가능하게 한다.

새로운 GPU인 ATI RadeonTM HD 3800 시리즈는 DirectX 10.1뿐만 아니라 PCI Express 2.0, 통합 비디오 디코더(Unified Video Decoder, UVD), 하드웨어 가속 모자이크화(Hardware accelerated tessellation) 그리고 효율적인 전력 사용을 위한 55nm의 트랜지스터 설계 등을 포함하는 첨단 기술로 설계된 최초의 제품이다. 이 제품은 현재의 게임들뿐만 아니라, 2008년 이후, 등장하는 차세대 게임 타이틀에서도 최선의 경험을 전달하기 위해 시장에 등장하였다.

이 문서에서는 DirectX 10.1의 새로운 특징들을 기술하고, 이를 사용하기 위해 어떻게 적용되는가에 대한 몇 가지 예를 제공할 것이다. 이러한 기술들을 설명하는 것을 돕기 위해, AMD에서는 PingPong이라 불리는 상호작용 게임을 개발해왔다. 이 게임은 ATI RadeonTM HD 3800 시리즈 제품에서 흥미있고 유익한 방법으로 DirectX 10.1의 이점을 강조하기 위해 DirectX 10.1의 특징들이 광범위하게 사용되도록 개발되었다.

■ DirectX의 발전 The Evolution of DirectX

DirectX 10.1은 DirectX 10의 전반적인 구조와 프로그래밍 모델을 유지하는 반면, 수많은 개선점이 제공된다. 버텍스, 지오메트리 그리고 픽셀셰이더 명령어 세트는 Shader Model 4.1로 업데이트 되었다.

DirectX 10.1의 새로운 특징들은 새로운 셰이딩(Shading)과 텍스처링(Texturing)의 호환성, 안티앨리어싱의 향상, 높아진 사양 등 세 가지 일반적인 범주로 구분된다. 다음의 표는 이러한 범주의 각 핵심적인 특징들과 그에 따른 이점을 나타내고 있다.

셰이더와 텍스처의 개선

특징	기능	이점
큐브 맵 정렬 Cube Map Arrays	단일 렌더링 패스에서의 다중 큐브 맵의 읽기/쓰기 가능	프로그래머가 각각의 픽셀을 위한 개별적인 샘플 기억 장소 제어 가능
MRT마다 블렌드모드 분리 Separate Blend Mode per-MRT	개별 블렌드 모드에서 픽셀셰이더의 다중 버퍼 출력 가능	복합 3D 장면에서 향상된 성능을 위한 효율적인 지연 렌더링(Deferred rendering)
증가된 버텍스셰이더 입출력 Increased Vertex Shader I/O Gather4	셰이더당 128bit로 처리 가능 한 값이 16에서 32로 배로 증가	복합 셰이더를 위한 향상된 성능
Gather4	단일 이선형의 필터된 텍스처 참조 대신 필터 되지 않은 텍스쳐값의 2×2 블럭 참조 가능	스트림 컴퓨팅 애플리케이션을 위한 향상된 성능
LOD 명령어 LOD Instruction	필터된 텍스처 참조를 위해 세부적인 레벨로 되돌리는 새로운 셰이더 명령어	- 최적화된 성능과 품질을 위한 커스텀 - 텍스처 필터링 기법

향상된 안티앨리어싱

멀티샘플 버퍼의 읽기와 쓰기 Multi-sample buffer reads and writes	셰이더에 의해 직접적으로 접근되는 멀티샘플에서 개별적인 색상 및 깊이 샘플 허용	- 최적화된 성능과 더불어 고품질 안티앨리어싱을 위한 커스텀 에지 검출 필터 - 향상된 적응형 안티앨리어싱(adaptive AA) 성능 - HDR 렌더링과 동시 적용되는 안티앨리어싱 품질의 향상된 호환성 - 지연 렌더링 기법이 적용된 안티앨리어싱의 향상된 호환성과 성능
픽셀 범위 마스크 Pixel Coverage Masks	픽셀 셰이더에서 프로그래밍이 가능한 안티앨리어싱 허용	
픽셀 범위 마스크 Pixel Coverage Masks	프로그래머가 각각의 픽셀을 위한 개별적인 샘플 기억 장소 제어 가능	- 일시적인 안티앨리어싱 (Temporal AA) - 다중 GPU 안티앨리어싱 기법을 위한향상된 이미지 품질

높아진 사양

부동소수점 32비트 필터링 FP32 filtering	128비트 유동점 텍스처를 필터링	- 하드웨어 호환성 확보에 의한 높은 정밀도의 데이터 구성 사용을 촉진
정수연산 16비트 블렌딩 Int16 blending	64비트 정수 픽셀의 블렌딩	
최소 4x MSAA 지원 요구 Minimum 4x MSAA support	픽셀당 최소 4샘플의 멀티샘플 안티앨리어싱이 32-bit와 64-bit 픽셀 포맷에 모두 지원	- 모든 DirectX 10.1 GPU들 상에서 동등하게 안티앨리어싱 적용가능 - 개선되는 견고함에 의한 안티앨리어싱의 지원 촉진
표준화된 AA 샘플 패턴 Standardized AA sample pattern	하드웨어가 지원하는 2x, 4x, 8x 그리고 16x AA 모드를 위한 사전 정의된 샘플 기억주소	
부동소수점 연산을 위한 높아진 정밀도 Increased precision for FLOP	모든 부동소수점 연산(가/감/곱/제함)과 블렌딩 연산이 요구되는 0.5 ULP 정밀도	- 라운딩 시 발생 오류 제거 - 많은 종류의 연산들을 위한 IEEE 표준요구 만족

5

■ 차세대 이미지 품질 Next Generation Image Quality

광원과 그림자

현재 실시간 렌더링과 비실시간 렌더링간 이질성은 물리적으로 사실적인 광원에 기초한 전역 조명효과를 비실시간 렌더링으로 사용하기 때문에 발생한다. 임의점에서 광원 반사의 색상과 총량을 결정하는 프로세스는 렌더링이 이루어진 장면에서 깊이의 감각을 주는 것과 3D 공간 내 객체의 위치와 이동을 관찰자가 판단하게끔 하는 데에 큰 영향을 끼친다.

이러한 예로 간접적 또는 반사되는 광원이 관찰자에게 한 장면 내에서 시각적으로 객체를 정렬하도록 하는 것을 들 수 있다. 한 객체가 밝게 채색된 다른 객체로 다가갈 때, 반사된 빛은 컬러 블렌딩을 일으키고, 이는 두 객체의 접근에 대해 중요한 시각적 역할을 담당한다. 예를 들어, 백색 공이 적색 공에 접근할 때 벽을 향하고 있는 공의 부분은 벽에 반사된 빛에 의해 불그스름한 색을 띄기 시작할 것이다. 인간의 지각 체계는 이러한 정보를 처리하여 직관적으로 벽 가까이 공이 있음을 이해한다. 이러한, 사실은 게임 플레이에서 중요한데 특히, PingPong 내에서 이러한 지각과정은 플레이어가 3D 공간 내 공의 위치를 자연스럽게 파악할 수 있기 때문이다. 이는 플레이어에게 더 풍부한 경험을 주고, 전략적인 게임플레이에 대한 판단이 수월하도록 만들어준다.

현재 라이트/셰도우 맵핑(Light/Shadow mapping), 라디오시티(Radiosity) 그리고, 광선 추적(Ray tracing)과 같이 3D 렌더링에 의한 수많은 조명 기법이 있다. 광선 추적 기법은 CPU와 유사한 아키텍처에서 더 나은 프로세싱을 보이는 경향이 있는 반면, 라이트/셰도우 맵핑 기법과 특정의 라디오시티 기법은 GPU와 유사한 아키텍처에서 더 나은 프로세싱을 보이는 경향이 있다.

ATI RADEON PingPong 게임의 이 장면은 전역 조명효과로서, 플레이어가 제어할 수 있는 형태의 물리적으로 시뮬레이션된 수천 개의 객체에서 사실적인 광원과 그림자가 나타나는 것을 보여준다. 우측 이미지는 전역 조명효과가 활성화되지 않은 장면이 어떻게 나타나는가를 보여준다. 이러한 매우 동적인 형태를 그리기 위해 광선 추적(Ray tracing) 기법을 사용하는 것은 오늘날 소비자의 하드웨어로는 실시간으로 렌더링하여 만들어내는 것이 실제로 불가능하기 때문이다.

라이트/셰도우 맵핑은 각 광원에서 하나 또는 그 이상의 텍스처로의 관점으로부터 이미지의 표현을 렌더링하여 이루어진다. 각 픽셀에서, 각각에 의해 기여되는 빛의 색상과 총량을 결정하기 위해서 또한, 각각의 광원이 그 픽셀에서 보이는 정도(만약 보이지 않으면 어두워지는 것으로 고려)를 이들 텍스처로부터 참조하게 된다. 이후, 표면 셰이더 프로그램은 재료 물성을 고려하고, 최종 픽셀 색상을 결정하기 위해 모든 기여 정보를 평가하고 조합한다.

이러한 조명 기법은 표준의 흩어져 있는 빛의 캡처를 위한 직접적인 방법이지만, 임의로 처리된 빛의 반사가 묘사되기 위해서는 일반적으로 독립된 큐브 맵이 요구된다. 그것은 상대적으로 빠르고 동적인 장면들을 잘 처리하지만, 반사와 그림자 품질은 라이트/셰도우 맵의 제한된 해상도로 인해 저하될 수 있다. 또한, 많은 수의 포인트 또는 영역의 광원은 잘 처리하지 못하며 간접 조명은 캡처하지 못한다.

광선 추적은 스크린 상에서 모든 가시 픽셀의

위치를 향한 관점에서부터 조사(照射)된 광선에 의해 이루어진다. 각각의 광선이 객체와 교차하는 첫번째 포인트를 결정하고 나서, 모든 방향에서 그러한 각각의 포인트로 광선이 조사된다. 빛의 반사가 끝나는 포인트의 색상과 총량을 결정하기 위해 각 포인트에서 셰이더가 수행된다. 이런 프로세스는 간접 조명을 얻기 위해 복합적인 반사의 요구가 반복될 수 있다. 또한, 이 방법은 빛의 반사, 굴절 그리고 그림자의 처리에 우수하다. 하지만, 광선 추적은 상식적으로 적당한 해상도에서 실시간 수준의 프레임 레이트에 도달하기 위해 초당 수십억 개의 광선을 요구한다. 또한, 장면에서 모든 객체의 위치를 저장하기 위해 복합적인 데이터 구조를 필요로 하는데, 이는 동적인 장면일 경우, 광선추적 기법은 각 프레임이 업데이트되기 위해 이러한 데이터 구조를 요구하게 되어 처리과정이 비효율적이게 된다.

최종적으로, 광선 추적은 셰이딩 처리에는 적합하지 않으며, 이는 복합 표면 셰이더가 많은 후속 광선들의 처리를 위해 각 픽셀에서 반복적으로 수행되어야 하기 때문이다.

오늘날 하드웨어에서 광선 추적은 터무니없이 비경제적이므로, 현재 대부분의 게임들은 '일정 환경 조건(Constant ambient term)' 에 대해서는 간접 조명으로 처리한다(즉, 바꿔 말하면 전체 장면에 기여하는 개별적이지 않은 소스로서의 균일한 조명효과). 이는 최상의 방법에 매우 거친 접근이며, 색상 번짐 또는 그림자 처리를 하지 않는다.

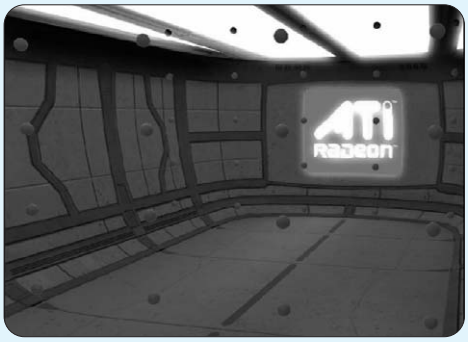
라디오시티는 앞서 소개한 방법과는 또 다른 몇몇 게임에서 사용된 라이팅 및 셰도잉 기법으로 라이트/셰도우 맵핑과 광선 추적의 몇 가지 장점을 조합한 방법이다. 하나의 장면에 대하여 조명의 데이터 수치를 미리 연산하고 저장하기 위해 광선 추적과 유사한 방법으로 처리한 후, 빠르게 간접 조명으로 할당하기 위해 렌더 타임이 되면 앞서 저장했던 수치를 사용한다. 하지만, 광범위한 사전 연산은 광원의 이동이나 변화가 이루어지는 동적인 장면에 대하여 라디오시티 기법으로는 잘 처리하지 못함을 의미하기도 한다.

전역 조명효과

전역 조명효과는 간접 조명에 라이트/셰도우 맵핑의 이점과 실제로 무한한 동적 광원, 사실적인 반사 그리고, 유연한 그림자의 지원을 조합한 렌더링 기법이다. 개발자들은 DirectX 10.1을 통해 복잡하고 상호작용하는 장면들까지도 물리적으로 모델링된 수천 개의 객체에 대한 실시간적인 전역 조명효과가 가능하며, 이를 효율적으로 수행하기 위해서 표시된 큐브 맵 정렬(Indexed cube map array)과 지오메트리 셰이더 (Geometry shaders)를 사용할 수 있다.

이러한 기법은 그와 같은 장면을 큐브의 3D 정렬 형태로 나누어 처리한다. 각 큐브 면에서, 단순한 장면은 외관이 보이는 큐브의 중심에서 누군가의 관점으로부터 그려지게 된다(빛 탐침, Light prove). 이러한 이미지의 해상도와 디테일은 일반적으로 최종 이미지의 해상도와 디테일만큼 높을 필요는 없지만, 요구되는 성능과 정밀도의 수준에 따라서 쉽게 측정가능하다. 각 큐브의 여섯 면이 완전히 그려지면, 곧 큐브 맵 텍스처에 저장된다. 다음 단계는 구형 하모닉스(Spherical harmonics)를 사용하여 각 큐브 맵을 압축된 구형 표현으로 변환하는 것이다. 이런 새로운 표현을 이용하여 소수의 단순 수학 연산으로 임의의 방향과 모든 방향으로부터 큐브 내부의 어떤 한 점 위 라이트 폴링(Light falling)의 색상과 총량을 신속하게 결정할 수 있게 된다. 라이트 폴링들 간 포인트들에서, 조명의 데이터 수치는 근접한 큐브 맵들로부터 취한 수치사이

에서 변조(Interpolation)될 수 있다.



전역 조명효과를 캡처하기 위해 사용된 라이트 폴링의 일러스트레이션

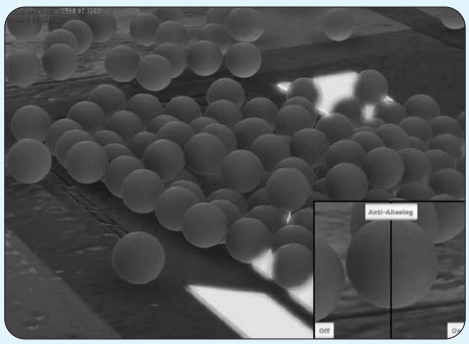
전역 조명효과의 유효성을 증가시키는 방법에는 동적 환경 폐쇄 기법(Dynamic ambient occlusion technique)이 있다. 이는 전역 조명효과 스스로에 의해 제공되는 것을 넘어서는 국부적인 가시성의 변화를 캡처할 수 있는 중요한 요소이다. PingPong 게임에서, 동적 환경 폐쇄 기법은 공들이 서로 접근하거나 벽에 접근하는 장면에서 공에 부드러운 테두리의 그림자를 부여한다. 이러한 접근 그림자는 장면에서 객체의 위치를 분석하는 데 중요한 시각적 역할을 한다. 또한, 큐브 맵 텍스처는 그 장면에서 빛나거나 광택이 있는 객체 상의 고품질 반사를 만드는데 사용된다. PingPong 게임에서, 면밀히 살펴보면 수천 개의 각 공들이 그 표면에서 환경 반사가 이루어지는 것을 볼 수 있다. 조명을 계산하는 이 방법은 대단히 확장 가능하다. 큐브 면 이미지에서의 세부묘사 수준뿐만 아니라, 사용된 큐브의 크기와 수의 변화에 의해서 품질수준 또한, 제어될 수 있다. DirectX 10.1 큐브 맵 정렬에서, 이 기법이 단일 GPU 또는 다중 GPU 상에서 특히 효율적이게 함으로써, 많은 수의 큐브 맵들은 평행한 상태에서 동시에 렌더링 및 샘플링 될 수 있다.

만약, 매우 복합적인 장면들이 요구된다면, 다수의 빛 반사는 최종 이미지를 그리기 전에 이러한 프로세스를 반복하고 또한, 그 결과를 축적함으로써 쉽게 시뮬레이션 될 수 있다. 게다가, PingPong 게임의 경우에서 렌더링 되는 다수의 공들이 단지 물리 시뮬레이션의 속도에 의해서만 제한되는 것처럼, 조명효과의 품질은 최종 장면에서 다수의 객체 또는 폴리곤으로부터 독립적으로 처리될 수 있다.

요약하면, DirectX 10.1은 간접조명, 색 번짐, 부드러운 그림자, 반사와 굴절이 포함되는 실시간 전역 조명효과를 사용함으로써 조명효과의 품질에 있어 획기적인 진전이 가능하다.

■ 안티앨리어싱 향상 Anti aliasing Improvements

앨리어싱으로 발생하는 다듬어지지 않은, 희미한 테두리는 렌더링된 이미지에서 매우 어수선한 모양새를 나타낼 수 있다. 이런 결과물들은 부적당한 렌더링 또는 디스플레이 해상도의 결과물이다. 이들은 픽셀당 다수의 샘플을 사용하고 그들 모두를 블렌딩으로 처리하는 필터링 기법으로 감소 또는 제거될 수 있다. 이 기술은 디테일의 흐림없이 우수한 성능을 유지하면서, 테



ATI RADEON PingPong 게임 이미지에서, 에지 검출 AA 필터로 검출되어 적색으로 강조된 테두리 픽셀은 다른 픽셀들보다 더 많은 샘플이 할당된다. 우측 하단에 삽입된 그림은 공의 테두리에서 울퉁불퉁한 테두리와 희미한 결과물이 안티앨리어싱으로 어떻게 제거되는가를 보여준다. 그 효과는 이미지가 움직일 때 좀 더 드러난다.

두리의 다듬기가 최대한 이루어지도록 샘플의 포인트와 중량을 선택하기 위함이다.

현재, 가장 일반적으로 사용되는 안티앨리어싱 기법은 멀티샘플 안티 앨리어싱(MSAA)이지만, 이는 단지 폴리곤의 테두리에서만 처리되며, 텍스처 앨리어싱이나 셰이더 앨리어싱에는 적용되지 않는다. ATI RADEON HD 2000 시리즈에서는 커스텀 필터 안티앨리어싱(CFAA)이 도입되었으며 이는 정교하게 프로그램 가능한 필터를 가능하게 하였다. 최신의 Catalyst 드라이버는 DirectX 9나 그 이전 버전을 사용하는 거의 모든 게임에 적용 가능한 4종류의 필터 타입을 지원한다. 이 중 가장 발전된 형태는 에지 검출 필터(Edge detect filter)로서, 텍스처와 셰이더에서의 모든 테두리를 검출하고 안티앨리어싱한다.

커스텀 안티앨리어싱

DirectX 10.1은 커스텀 안티앨리어싱이 픽셀 셰이더로 수행되도록 한다. 커스텀 필터는 예컨대 HDR 조명 효과와 지연된 셰이딩 기술처럼, 표준의 MSAA가 적용될 수 있는 확실한 경우에서도 향상된 품질을 제공한다. DirectX 10.1과 호환되는 모든 하드웨어들은 최소 4x MSAA를 지원해야 한다. 그 사양은 오로지 개개의 GPU를 정의하기 위하여 샘플을 남겨둔 DirectX의 이전버전과 달리 다소의 사전 정의된 AA 샘플 패턴들을 포함한다.

DirectX 10.1의 새로운 특징 중 하나는 모든 AA 버퍼가 셰이더에 의해 직접적으로 액세스되도록 허용하는 것이다. 이전에는, 단지 멀티샘플의 색상 버퍼에서만 액세스가 가능하였고, 각각의 샘플이 독자적으로 깊이 버퍼로부터 정보를 액세스하는 것은 불가능하였다. 이는 개발자들이 ATI RADEON HD GPU들이 CFAA를 통하여 수행하는 것처럼 셰이더와 제공된 하드웨어 간 콤비네이션을 이용하여 좀 더 진보된 커스텀 AA 기법을 사용할 수 있도록 한다.

또한, ATI RADEON HD는 부분적으로 투명한 텍스처(잎모양과 체인고리 울타리와 같은)에 안티앨리어싱을 제공하는 적응형 안티앨리어싱에 대한 지원을 도입하였다. DirectX 10.1은 픽셀셰이더가 실행되는 특정한 샘플 위치에 대한 제어가 가능한 샘플 적용범위의 마스킹 (Masking)을 도입하여 이러한 가능성을 확장한다. 이는 개발자들이 적응형 안티앨리어싱 기법을 좀 더 많은 앨리어싱 결과물에 적용하는 것을 확대시키는 것이 가능해진다. 그리고, 커스텀 샘플 패턴들은 모든 호환되는 하드웨어에 의해 지원되어야 하는 기본 모양새를 보완하기 위해 특성화된다. 이러한 많은 가능성들은 이미 이전 세대의 ATI RADEON GPU에서 제공되어졌지만, DirectX 10.1은 그러한 가능성들을 맨 먼저 개발자들에게 직접적으로 접할 수 있게끔 하는 것이다.

요약하면, DirectX 10.1은 최종적으로 개발자들에게 그들이 상호작용의 실시간 게임으로부터 모든 형태의 앨리어싱 결과물들을 제거하기 위해 필요한 도구를 주며, 이는 주로 이미지 품질의 향상을 가져올 것이다

새로운 텍스처 포맷의 요구

최근의 DirectX 버전에서 보다 높은 정밀 텍스처와 출력 포맷을 사용하면서 개발자들을 가로막은 하나의 장애물은 각각의 GPU가 어떤 연산을 수행할 수 있는 능력의 제한이었다. DirectX 10.1은 32bit 부동소수점 포맷의 텍스처 필터링, 16bit 정수 포맷의 블렌딩 연산을 지원하기 위한 호환성 높은 GPU를 요구하여 이러한 문제를 개선한다.

새로운 멀티샘플 안티앨리어싱의 요구

멀티샘플 안티앨리어싱은 이미지 품질 향상을 위해 잘 정립된 기법이다. 그렇지만 수년 동안 많은 개선법들은 기본적인 기법에서부터 발전하였다. 이러한 개선 사항들은 다른 종류의 GPU에서 의미심장하게 다른 결과를 내놓을 수 있었기 때문에 개발자들은 종종 그들의 게임에서 개선 사항을 바로 적용시키길 꺼려하는 경향을 보인다. DirectX 10.1은 사전 정의된 샘플 패턴 세트뿐만 아니라, 최소의 안티앨리어싱 품질 요건을 요구한다. 이는 개개의 GPU상에서 새로운 기법이 여전히 적용되는 동안에 변함없는 고품질의 안티앨리어싱이 모든 GPU에서 지원되는 점을 확실히 한다.

더 높은 정밀도의 요구

모든 데이터 포맷은 그들이 지원할 수 있는 양이 제한된 정밀도를 가지며, 가용할 수 있는 bit의 수에 의존한다. 그러나, 이들 포맷에 이루어진 연산은 반드시 모든 유용한 정밀도의 충분한 이점을 가지는 출력을 내놓지 않는다. 몇 가지 경우를 보면, 출력의 최소 중요한 비트(bit)에서 라운딩 오류를 야기할 수 있는 근사치가 사용된다. 이런 습관은 몇몇의 경우, 예측할 수 없는 거동이 야기될 수 있다(예를 들어, 많은 횟수의 연산이 반복되는 동안 오류가 발생할 때). DirectX 10.1은 32bit 부동소수점에서 기본적인 수학연산이 충분한 정밀도의 이점을 활용할 수 있도록 권한을 위임함으로써 이러한 이슈에 대처한다. 이런 식으로 확실하게 동일한 결과는 호환가능한 모든 GPU에서 마지막 비트로 내려 보낸다.

■ 결론 Conclusion

DirectX 10.1은 DirectX 10의 한계에 대처한 프로그래밍 인터페이스로 향상된 개선점을 제공하며, 2008년 이후 한 단계 높아질 3D 그래픽의 품질을 적용할 수 있는 새로운 그래픽 기법을 열어준다. 그 장점들은 CG 필름에서 사용된 광선 추적 기법과 대등하게 실시간으로 라이팅과 셰도우 품질을 전달하는 전역 조명효과, 어수선하게 희미한 결과물을 분명하게 하기 위한 개선된 안티앨리어싱 기법, 그리고 개선된 호환성을 위해 더 높아진 사양 등을 포함한다. ATI RAEDON HD 3800 시리즈는 이러한 특징과 장점을 PC로 가져다줄 세계 최초의 GPU가 될 것이다.