Journal of Korea Multimedia Society Vol. 8, No. 4, April 2005(pp. 536-543)

DirectX를 이용한 게임 설계에서의 생성 패턴 적용 기법

김종수†, 김태석††

요 약

국내 게임 분야는 다양한 게임 장르 중에서 사용자들에게 사실감을 더해주는 3D 기반 온라인 게임이 주류를 이루고 있다. 국내의 게임산업은 전문 인력이 부족하고 개발 기업이 영세하며, 게임과 관련된 설계 기술의 보안 때문에 기술 공유가 어려운 실정이다. 이러한 측면에서 볼 때, 인력과 시간이 많이 드는 네트워크 게임제작 시에 기존에 작성된 코드를 재사용이 가능하도록 소프트웨어를 설계하는 기법이 중요하다. 본 논문에서는 DirectX를 기반으로 하는 네트워크 게임의 클라이언트 측 설계에서 사용사례(use case)를 이용하여 요구 사항을 분석하고, 클래스 설계에 GoF(Gang of Four)의 디자인 패턴분류 중 생성패턴에 대한 게임 소프트웨어 설계의 재사용 기법을 제안한다.

The Creational Patterns Application to the Game Design Using the DirectX

Jong-Soo Kim†, Tai-Suk Kim††

ABSTRACT

3D online game, with its striking realistic value, is leading the entire Korean game market which has various game genres. Technology sharing is very hard within the Korean game industry. That is because 1)there are few professionals, 2)most of the companies are small-scaled, and 3)there are security reasons. Therefore, it should be significant if we have software design techniques which make it possible to reuse the existing code when developing a network game so that we could save a lot of efforts. In this paper, the author analyzes the demand through the case in the client's design of the network game based on DirectX and proposes the effective software design methods for reusable code based on the creative patterns application in the GoF in the class design.

Key words: Network Game(네트워크 게임), DirectX, UML, Design Patterns(디자인 패턴)

1. 서 론

첨단 게임은 영화나 오페라를 능가하는 지식기반 종합예술로서 음악, 영상, 시나리오, 디자인 등 다양한 분야의 전문적인 컨텐츠로 구성된다. 전 세계적으로 게임 산업은 MS사의 X-Box, Sony의 PlayStation II 등이 시장 선점을 위해 각축하고 있으며, 새로운 차원의 실시간 3D 게임 제작을 위해 노력하고 있다[1]. 현재 국내의 몇몇 회사들도 기존에는 거의 불가능할 것으로 예상되었던 네트워크 기반 실시간 전략 시뮬레이션 게임을 제작하고 있다[2].

소프트웨어의 라이프 사이클이 짧아지고 있는 추세이므로, 보다 재미있는 게임을 얼마나 빨리 개발할 수 있느냐 하는 것은 중요한 일이다. 그러나 게임 소

※ 교신저자(Corresponding Author) : 김태석, 주소 : 부산시 부산진구 엄광로(가야 산24번지)(614-714), 전화 : 051)890-1707, FAX : 051)890-1724, E-mail : tskim@deu.ac.kr
접수일 : 2004년 9월 8일, 완료일 : 2004년 11월 2일
† 정회원, 동의대학교 영상 미디어센터 PM연구원
(E-mail : seatree@deu.ac.kr)
†† 종신회원, 동의대학교 공과대학 소프트웨어공학과 교수

프트웨어를 설계하고 개발하기 위한 기법은 회사의 중요 자산이므로 아주 철저하게 보안되고 있고, 정보의 공유가 어렵다[3]. 이러한 이유로 게임 개발 기술에 적용되는 설계기법의 효율성에 대한 평가는 거의 없다[4].

본 논문에서는 DirectX를 기반으로 하는 3D 네트워크 게임 제작에서 소프트웨어의 재사용을 높일 수 있는 GoF의 디자인 패턴 영역 중 생성 패턴의 적용에 대해서 연구한다. 소프트웨어 개발에 있어서 객체지향 패러다임을 적용하는 이유는 여러 가지가 있지만, 분산된 개발을 가능하게 하고, 기존에 개발된 소프트에어의 재사용을 쉽게 한다는 장점이 있기 때문이다[5]. 본 논문에서는 게임 개발자들이 네트워크 게임을 제작하는데 있어서 효과적으로 사용할 수 있는 몇 가지 설계 패턴을 제시한다.

2. 관련 연구

게임 소프트웨어를 제작하기 위해서는 성능이 우수한 하드웨어 이외에도 소프트웨어적으로 여러 가지 기술이 필요하다. PC를 기반으로 해서 화려한 그래픽, 애니메이션 그리고 다양한 사운드가 지원되는 게임 소프트웨어를 제작하기 위해서는 기본적으로 MS 사가 제공하는 DirectX API를 이용하여 3D를 렌더링 할 수 있는 기능과 애니메이션 기능, 사운드 기능을 효율적으로 구현하는 것이 필요하다. 이를 위하여 3D 네트워크 게임 구현에 필요한 DirectX와 관련된 기술을 알아본다.

2.1 네트워크 게임 엔진 구성

네트워크 게임 엔진 설계와 제작은 소프트웨어의 재사용이라는 측면이 고려되어야 하므로, 엔진간의 인터페이스가 서로 종속적이지 않아야 효율적이다. 그림 1은 일반적인 네트워크 게임에서 엔진의 구성을 보여준다. 3D 게임 엔진은 크게 클라이언트 부분과 서버 부분으로 나눌 수 있다[6]. 인공지능 엔진은 클라이언트와 서버가 모두 필요하고, Input 엔진과 3D Graphic, 3D Sound 엔진은 클라이언트 측, 데이터베이스 관리 엔진은 서버 측에 위치한다.

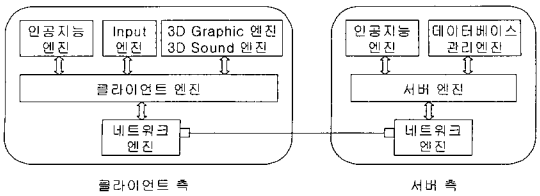

그림 1. 클라이언트/서버 게임의 엔진 구성도

2.2 DirectX

실시간 네트워크 게임 제작에 있어서 3D 그래픽 엔진은 렌더링과 애니메이션 기능을 가지고 있어야 한다[7]. 렌더링 API로 OpenGL이나 DirectX를 사용할 수 있는데, 본 연구에서는 마이크로소프트사가 제공해 주는 DirectX를 기초로 연구되었다. DirectX는 렌더링 엔진으로써 정점(Vertex), 폴리곤(Polygon), 메시(Mesh), 행렬(Matrix), 광원(Lighting), 텍스처(Texture) 등과 같은 것을 쉽게 다룰 수 있는 기능이 있다[8].

Direct3D 렌더링 엔진을 제작하기 위해서는 사원수, 삼각함수, 빌보드(billboard), 셰이더(Shader), 쿼드트리(Quadtree)와 같은 많은 수학적 기초 지식과 최적의 애니메이션 구현하기 위한 기술이 필요하다.

애니메이션을 처리를 위한 기법으로는 키 프레임 애니메이션(key frame animation), 정점 애니메이션(vertex animation), 층적 애니메이션(hierarchical animation), 뼈대 애니메이션(bone based animation, skeletal animation), 얼굴 표정 애니메이션(facial expression)과 같은 것이 있다.

2.3 GoF의 디자인 패턴(Design Patterns)

게임과 같이 복잡한 소프트웨어를 작성하는데 있어서 패턴의 사용은 매우 효율적인데, 이러한 패턴의 사용은 PC용 게임과 같은 소프트웨어 개발자가 다른 사람의 전문성을 쉽게 이용할 수 있게 해준다.

GoF가 제안한 디자인 패턴은 크게 3개의 패턴 영역으로 분류되는데, 각각은 생성패턴, 구조패턴, 행위패턴이다. 생성 패턴은 객체의 생성 방식을 결정하는 포괄적인 방법을 제공하고 구조패턴은 상속 기법을 이용하여 더 큰 구조를 형성하기 위해 클래스와 객체를 어떻게 합성하는가와 관련된 것이고 행위패턴은 객체의 행위를 조직화(organize), 관리(manage), 연합(combine)하는 방법을 제공한다.

3. DirectX를 이용한 게임 분석 및 설계

게임 설계에 있어서 GoF가 제안한 디자인 패턴을

적용하였는데, 게임 소프트웨어의 분석과 설계에도 디자인 패턴이 유용하게 적용될 수 있음을 보이고, 소프트웨어 개발 단계에 따른 분석과 설계 시에 디자인 패턴을 이용함으로써, 객체지향 패러다임이 가지고 있는 소프트웨어 분산 개발과 소프트웨어 재사용이라는 측면이 최적의 게임 소프트웨어를 제작하는데 효용이 있음을 보인다.

3.1 사용사례(use case) 작성

요구 사항은 일반적으로 만들어질 소프트웨어가 갖추어야할 능력과 조건을 말하는데, 사용사례의 작성은 요구사항을 분석하고 기록하는데 널리 사용되는 기법이다[9]. 게임 분석 설계에 필요한 요구사항은 게임 소프트웨어 개발자들 간의 아이디어 회의를 통해서 분석해 내는 경우가 많다.

성공을 거두고 있는 게임 소프트웨어를 분석해 보면, 잘 짜여진 게임 스토리가 바탕이 되어 있는 경우가 많음을 볼 수 있는데, 사용자에게 흥미로운 게임을 제작하기 위해서는 게임 스토리의 작성이 중요하다는 것을 보여준다. 게임은 다양한 장르가 있지만, 본 논문에서 구현된 게임은 MMORPG(Massive Multi-player Online Role Playing Game) 형태의 네트워크게임을 위한 클라이언트 측 구현이다.

표 1. "Legend of Avantas" 게임 스토리

스토리 보드
제목 : L.o.A(Legend Of Avantas) 장르 : MMORPG(Massive Multi-player Online Role Playing Game) 게임 스토리 및 배경 고대 상계에 신계와 마계가 있었다. (...생략) 마을별 특징 및 특성 ① 라피돌 마을 아반타스의 신으로부터 내려온 라세스 종족이 머무는 곳 .....(이하 생략)

표 1은 구현하고자 하는 게임의 스토리에 나타난 요구사항을 보여준다. 표 1에서 작성된 게임스토리를 바탕으로 갖춘 형식의 사용사례를 작성하였는데, 표 2와 같다.

표 2의 머리말 구성요소에서 주요 엑터(Primary Actor)는 게임 서버이다. 네트워크 게임에 있어서 가장 중요한 역할을 하기 때문에 양식의 첫 머리에 두었다. 발주자(stakeholder)와 관심사를 적는 부분은 구현하고자 하는 게임 소프트웨어가 무엇을 해야 하는지를 제시하고 범위를 설정하고, 발주자에 대한 관심사를 상세히 기술함으로써 시스템이 무엇을 해야 하는지 상세하게 알 수 있게 한다.

표 2. 게임 스토리 사용사례를 통한 요구사항 분석

UC1: 게임 시작
주요 엑터(Primary Actor) : 게임 서버(Game Server) 발주자(stakeholder)와 관심사 - 게임 서버(Game Server) : 안정적인 운영을 위해 하드웨어적인 부하 분산 시스템 설계가 필요하다. .....(중간 생략) 전제 조건(Preconditions) : 게임 서버는 가동 중이어야 한다. 성공 보장(Success Guarantee) 로그이한 정보가 저장된다. 주요 성공 시나리오(Main Success Scenario) 1. 게임 서버는 게임 플레이어의 접속을 기다린다. .....(중간 생략) 확장(Extensions) * 임의의 시점에서 시스템 실패 .....(이하 생략)

사용 사례 작성에 있어서는 주요 성공 시나리오(Main Success Scenario)와 이것의 확장(Extensions)에 대해 되도록 상세하게 기록하는 것이 핵심 사항이다[10]. 주요 성공 시나리오(Main Success Scenario)는 발주자의 관심사를 만족시키는 전형적인 성공 경로를 기술하는데, 어떤 조건이나 분기를 포함하지 않도록 작성한다. 확장(Extensions)은 성공 시나리오 외의 모든 시나리오와 성공과 실패시의 분기를 표현하는데, 일반적으로 갖춘 형식에서는 확장 부분이 주요 성공 시나리오 부분보다 더 길게 나타난다.

표 2의 요구사항 분석을 통하여 작성된 사용 사례 다이어그램(Use Case Diagram)은 그림 2와 같이 나타났는데, 다양한 엑터들을 볼 수 있다.

엑터란 시스템 내에서 행동의 주체가 되는 것을 말하는데, 표 2를 바탕으로 분석하면 Game Player, Game Server가 주요 엑터이고, 보조 엑터는 지불인증서비스다. 그리고 숨은 엑터는 Login Server, DB Server, NPC Server, Chatting Server 등이 된다.

다이어그램을 바탕으로 요구 사항을 분석해 보면 먼저 클라이언트 측의 가칭 "Avantas" 시스템 경계에서 나타난 Login과 관련된 처리, 게임 진행과 관련된 처리, 게임 캐릭터를 관리하기 위한 처리, 서버

간의 채팅을 처리하기 위한 프로세서, 게임에 등장은 하지만 움직이지 않고 스토리만을 처리하기 위한 NPC 관리, 사용자와 그에 따르는 보안 관리 등이 필요하다. Game Process는 Chatting Process, NPC관리, Character관리를 포함한다.

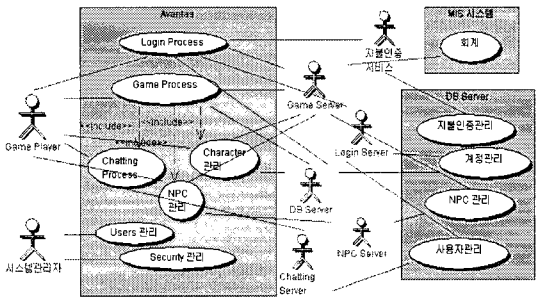

그림 2. 요구사항 분석에 기초한 Use Case 다이어그램

그리고 게임 진행을 처리하기 위한 서버 측에서 지불 인증 서버, Game Server, Login Server, DB Server, NPC Server, Chatting Server가 필요함을 알 수 있다. 또한 서버 측에 따로 구현되어야 하는 시스템으로 DB Server와 MIS 시스템이 있는데, DB Server를 구축하기 위해서는 여러 개의 사용 사례의 분석이 필요하다.

3.2 DirectX를 이용한 애플리케이션 디자인

일반적으로 게임 제작을 위해서는 플레이어가 실제로 게임을 하기 위한 게임 애플리케이션과 게임을 하기 전에 캐릭터가 위치하는 공간인 맵을 디자인하기 위한 맵 에디터로 구성되는데, 본 논문에서는 게임에 필요한 여러 가지 캐릭터를 디자인하기 위한 모델 툴을 별도의 애플리케이션으로 설계하였다.

표 1의 스토리보드와 표 2의 요구사항 분석을 토대로 게임 제작에 필요한 주요 클래스의 기능과 적용되어야 할 애플리케이션은 표 3과 같이 정의되었다. 표에 나타난 클래스 외에 맵 툴과 모델 툴의 경우는 MFC를 이용한 GUI(Graphical User Interface)를 작성할 수 있다. GUI를 작성하기 위해서 프레임의 생성과 구성을 관리하는 클래스, 자료를 관리하고 저장하는 클래스 등이 필요하지만 표에 나타내지 않았다. 우리가 구현하고자 하는 게임 애플리케이션이 갖추어야 할 요구사항은 게임을 위한 기본적인 기능이 제공되어야 하고, 소프트웨어 리팩토링의 단계에서

표 3. 애플리케이션 구현에 필요한 클래스 분석

주요 클래스	주요 기능	적용 부분
GameManager	카메라 관리	gm
InputManager	마우스와 키보드 관리	gm
UIManager	사용자 인터페이스를 제어	gm
Effect	움직임, 사운드 효과 관리	gm
AIManager	인공지능 관리	gm
Terrain	게임 영역을 관리	gm, mp(ru)
QuadTree	지형 검색 클래스	gm, mp(ru)
AseLoad	3D Max의 Ase 파일 처리	md(n)
Direct3D	메시 정보 처리 클래스	gm, mp(ru), md(ru)
ModelManager	전체 모델의 현재 상태 관리	gm, mp(rw)
Camera	캐릭터들의 카메라 관리	gm, mp(rw), md(rw)

(gm:게임, mp:맵, md:모델, n:신규, ru:재사용, rw:재작성)

설계에 사용된 디자인 패턴이 정련과 반복단계에 효율적으로 사용될 수 있어야 한다는 것이다.

표 3에서 나타난 클래스들의 거의가 게임 애플리케이션 구현에서 사용되는 클래스들이고, 게임 애플리케이션을 제작하기 위해 구현된 클래스의 일부가 맵 툴이나 모델 툴을 구현하기 위해 신규로 작성되어야 하는 것을 볼 수 있다.

네트워크 게임에서는 사이버 상의 캐릭터가 위치할 수 있는 다양한 맵(Map)이 존재하는데, 맵 에디터는 지형과 관련된 정보를 기초로 다양한 나무나 건물 같은 유닛(Unit)과 주인공 캐릭터의 위치, 적 캐릭터들의 위치를 지정한다. 맵 에디터에서 사용되는 클래스는 애니메이션과 사운드와 관련된 클래스를 제외하면 많은 클래스들이 맵 에디터에서도 같이 사용되거나 재 작성되어 사용될 수 있음을 볼 수 있다.

모델 툴은 게임에서 사용되는 캐릭터들의 속성을 조작하기 위한 툴이다. 캐릭터들의 속성을 부여하기 위해 3D MAX에서 가공해 주는 파일인 확장자가 KTK라는 파일을 이용하여 파일 자체에서 캐릭터들에 대한 속성을 가지고 있도록 구성하는 것이 효율적이다. 모델 툴에서도 게임 애플리케이션에서 구현되어진 몇 개의 클래스들이 재사용되거나 재 작성될 수 있음을 볼 수 있다.

일반적으로 모델 툴과 같은 기능은 맵 에디터에 같이 구현되는 경우가 많지만 본 논문에서는 모델 툴을 따로 구현하도록 설계하였다. 모델 툴도 맵 에디터와 같이 MFC가 제공하는 윈도우 프레임으로 작성될 수 있지만, 맵 에디터와 비교하면 기능면에서 다른 점이 있다. DirectX를 다루기 위한 DirectInput

과 Direct3D클래스는 기존의 클래스를 재사용할 수 있지만, 모델 툴은 3D MAX에서 가공한 ASE 파일과 KTK 파일을 조작하는 것이 우선적인 기능이기 때문에 ASE 파일을 조작하는 클래스인 AseLoad 클래스가 신규로 필요하다. KTK 파일을 조작하기 위한 DecodeTKM클래스는 기존의 클래스에 필요로 하는 기능들이 추가되어야 한다.

표 3을 토대로 클래스가 적용될 애플리케이션을 비교 분석해 보면, 네트워크 게임 제작에 있어서 소프트웨어를 구현하기 전에 선행되어야 하는 소프트웨어의 분석과 설계가 보다 효율적인 클래스를 만들 수 있고, 소프트웨어 재사용을 위한 설계에 다양한 생성 패턴이 적용될 수 있음을 볼 수 있다.

4. UML을 이용한 DirectX 게임 설계

본 절에서는 DirextX를 이용한 게임 제작에 있어서, UML 설계도구인 볼랜드사의 Together를 이용하여 효율적인 소프트웨어를 제작할 수 있도록 해주는 객체지향 개념과 디자인 패턴에 대해서 논의한다. 네트워크 게임의 클라이언트 측 구현에서 기존에 정의된 상속관계와 클래스의 합성과 집합(Composite and Aggregation)관계를 토대로 하여, 효율적인 소프트웨어의 제작을 위한 생성 패턴의 적용으로 GoF가 제안한 팩토리(Factory), 싱글톤(Singleton) 패턴의 적용을 검토한다.

4.1 Adapter 패턴 적용(구조패턴)

본 논문의 주제는 DirectX를 이용한 게임 설계에서 효율적으로 사용될 수 있는 생성패턴에 관한 것이지만, 연구와 관련된 설계에서 생성패턴의 구현과 밀접한 관계가 있는 어댑터 패턴을 약간 다루었다.

그림 3은 게임에서 사용될 여러 개의 캐릭터 클래스와 건물들의 클래스 설계에 적용된 어댑터 패턴이다. 사용자가 조정하게 되는 캐릭터는 일반적으로 사용자가 로그인을 하고, 게임을 시작할 때, 사용자가 선택한 특정한 캐릭터를 가지고 게임을 하게 된다. 캐릭터는 일반적으로 전사, 궁수, 법사로 나누어지고, 도둑, 광대와 같은 캐릭터들도 있다. 또한 게임 속에서 다양한 건축물들이 존재하는데, 무기상점, 약물상점, 전직을 위한 신전과 같은 것이 있고, 각각은 직업이나 종족별로 특색 있게 디자인된다.

그림 3의 클래스 다이어그램에서 나타난 게임에 필요한 캐릭터나 건물 클래스를 우선적으로 살펴보

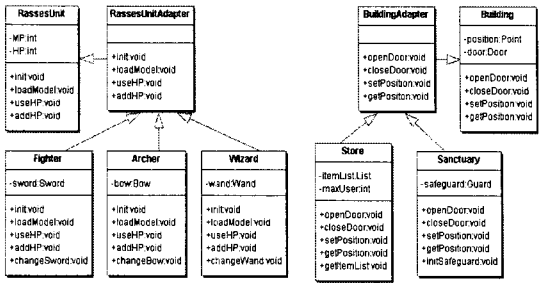

그림 3. Adapter 패턴을 적용한 캐릭터 설계

면, 같은 이름을 가진 메소드가 여러 개의 클래스에 같이 나타나는 것을 볼 수 있다. Fighter, Archer, Wizard 클래스의 useHP(), addHP()와 같은 메소드들이 그 예이다. Fighter, Archer, Wizard의 경우 추상화 개념을 적용하여, RassesUnit이라는 추상기저 클래스를 만들 수 있는데, RassesUnit 추상 기저 클래스는 init(), loadModel(), useHP(), addHP()와 같은 구현되지 않은 추상 메소드를 가지고 있고, 이 추상기저클래스를 상속받은 RassesUnitAdapter를 만들 수 있다.

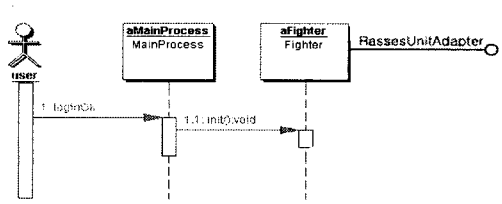

그림 4. RassesUnitAdapter의 시퀀스 다이어그램

각각의 Fighter, Archer, Wizard 클래스는 RassesUnitAdapter를 상속받아서 특성에 따른 구현에 필요한 부분만 재정의(overriding)하면 된다. 이러한 방법을 사용함으로서, 외에 새롭게 정의되는 캐릭터에 대해서도 RassesUnitAdapter를 이용할 수 있고, 새로운 캐릭터가 정의하고 있지 않는 속성과 메소드를 코드의 수정 없이 사용할 수 있다는 장점이 있다.

그림 4는 RassesUnitAdapter를 사용하기 위한 시퀀스 다이어그램을 보여준다. 어댑터 클래스를 사용하기 위해서 고려해보아야 하는 것은 어댑터클래스의 생성시점이다. Fighter 클래스의 객체가 RassesUnitAdapter를 사용한다는 것을 나타내고 있지만, 생성시점에 대해서는 표시되지 않았다. 이러한 문제는 팩토리(Factory)와 싱글톤(Singleton) 패턴을 사용함으로써 해결될 수 있다.

4.2 팩토리(Factory) 패턴 적용

게임 애플리케이션 설계에서는 앞에서 분석된 어댑터클래스 외에도 추가적으로 필요한 어댑터가 있을 수 있는데, 이러한 어댑터클래스를 언제 어떻게 생성할 것인가 하는 것을 고려해 보아야 한다. 만일 문제영역에서 분석된 특정 클래스가 어댑터를 생성하게 되면, 객체 생성의 책임성 문제가 생길 수 있다.

현재 나타난 설계에서 그림 3의 어댑터 클래스의 생성과 관련하여 생각해 보면, 어떤 특정 클래스가 어댑터 클래스의 생성 책임을 맡아야 하지만, 그렇게 하는 것은 객체의 응집도를 떨어뜨린다. 어댑터 클래스의 생성 책임과 관련한 문제는 설계의 기본 원리인 관심사항의 분리(separation of concerns) 원칙을 적용하는 것이 각각의 클래스가 응집된 목적을 가질 수 있도록 하는 좋은 방법이다.

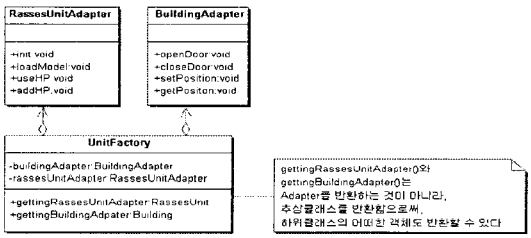

그림 5. Adapter를 생성하기 위한 UnitFactory클래스

그림 5는 RassesUnitAdapter와 BuildingAdapter를 생성하기 위한 팩토리(Factory)를 정의하고 있다. UnitFactory 클래스가 2개의 어댑터클래스 생성 책임을 맡고 있다. UnitFactory 클래스 내에 2개의 메소드가 정의되어 있는데, 각각의 메소드는 RassesUnit과 Building을 반환한다. 각각의 메소드는 Adapter클래스를 반환하는 것이 아니라 추상클래스를 반환함으로써, 다형성에 의해 하위클래스의 어떠한 객체도 반환할 수 있게 된다. 어댑터 클래스의 생성에 팩토리 패턴을 사용함으로써 복잡한 객체 생성 책임을 응집도가 높은 협력 객체로 분리할 수 있고, 잠재적으로 복잡한 생성논리를 숨길 수 있다.

4.3 싱글톤(Singleton) 패턴 적용

클래스의 응집도를 높이기 위한 UnitFactory 클래스의 사용에서 고려해보아야 하는 것은 클래스 자체의 생성과 접근을 책임지는 객체를 선택하는 문제인데, UnitFactory의 경우는 어댑터를 생성하기 위한 팩토리이므로 인스턴스가 여러 개일 필요가 없고 한 개만 생성되면 된다.

이러한 문제를 해결하기 위해 우선적으로 UnitAdapter 인스턴스가 필요한 객체들에 대해 필요한 곳마다 매개변수로 전달하는 방법을 생각할 수 있는데, 이 방법은 시스템의 유지 보수라는 측면에서 볼 때 불편한 점이 많다. 이에 대한 해결책으로 그림 6과 같은 싱글톤(Singleton) 패턴이 사용될 수 있다.

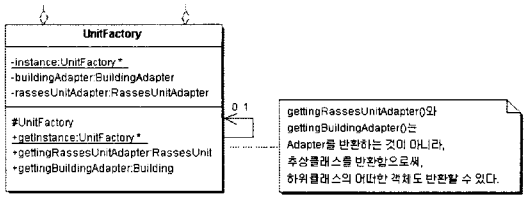

그림 6. UnitFactory에 적용된 Singleton 패턴

그림 6에서 UnitFactory는 private으로 선언된 자신의 주소를 참조하는 인스턴스를 속성으로 가지고 있고, protected로 선언된 생성자가 있으며, private 인스턴스에 접근할 수 있는 정적 메소드인 getInstance()를 가지고 있다.

4.4 클래스 다이어그램을 이용한 게임 설계

효율적인 소프트웨어 제작을 위하여 UML과 디자인 패턴을 사용하는 것이 많은 도움을 준다. 본 논문에서 구현된 애플리케이션은 정련과 반복을 거치면서 그림 7과 같은 다이어그램으로 설계되었다.

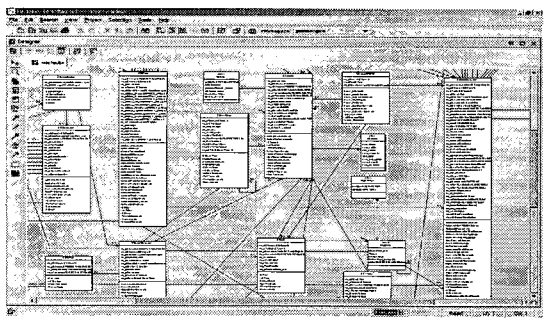

그림 7. 게임 애플리케이션 클래스 설계

게임 애플리케이션의 설계를 위해 가벼운 다이어그램을 시작으로 해서 여러 번 반복단계를 거치도록 하여 전체적인 구조를 보강하는데 중점을 두었다.

5. 생성 패턴을 이용한 DirectX 게임 시스템의 구현

게임과 툴을 구현하기 위해 사용된 자원들은 시스템 버스 스피드 1.6GHz의 인텔 팬티엄 4 CPU, 주메모리(Main Memory)는 256Mega 바이트, 그래픽 카드는 RIVA TNT2로 메모리는 64Mega 바이트를 사용하였다.

5.1 게임 애플리케이션의 실행

그림 8에서 보는 게임은 생성 패턴과 DirectX 9 API를 이용하여 제작하였고, 네트워크 게임의 클라이언트 측 기본 기능 구현에 중점을 둔 게임 애플리케이션으로 게임을 위한 기본적인 기능을 제공한다.

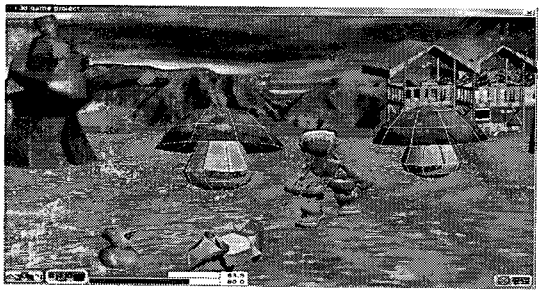

그림 8. DirectX 3D를 이용한 게임의 예

DirectX가 지원하는 다양한 스크린 모드가 있지만 구현에 사용된 스크린 모드(mode)는 800×600 모드를 사용하였다. 화려한 게임을 제작하기 위해 게임에서 나타낼 수 있는 색상의 깊이 선택도 중요한데, 너무 깊은 색상을 사용하면 자연스러운 색상을 나타낼 수 있지만, 메모리를 많이 사용해야 한다는 단점이 있다. 그래서 본 논문에 구현된 게임 애플리케이션은 16비트 컬러(color)를 사용하였다. 애니메이션 처리는 CPU의 속도, 메인 메모리의 크기, 선택된 스크린 모드와 컬러, 그래픽 카드의 성능을 모두 고려해야 하는데, 구현된 애플리케이션에서 DirectX 3D 렌더링 API는 초당 대략 50~90 프레임을 처리하였다.

5.2 맵 툴과 모델 툴의 실행

맵툴의 일반적인 기능은 게임 내에서 사용자가 캐릭터가 움직일 수 있는 폭과 길이를 설정하고, 캐릭터와 건물과 같은 객체의 배치를 결정한다. 그림 9는 구현된 맵 툴을 보여준다.

맵 에디터에서는 정적으로 구성된 위치 정보만 중요하므로 DirectX가 제공하는 스크린 모드를 사용하지 않아도 된다. 맵 에디터의 프레임 구성에 C++가 제공하는 MFC의 클래스를 사용하였고, 필요한 부분에 DirectX API를 사용하여 구현하였다.

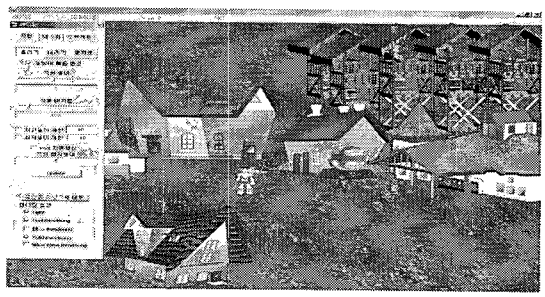

그림 9. 맵 에디터 구현 예

모델 툴의 주요 기능은 3D Max로부터 생성된 .ASE 파일을 파싱해서 파일에 포함된 애니메이션 정보를 알아내고, 캐릭터의 능력치를 조정해서 .KTK 파일 형태로 저장하는 역할을 한다. 모델 툴 구현에서는 게임 애플리케이션에서 구현되어진 몇 개의 클래스들이 효율적으로 재사용되거나 재 작성되었다. 그리고 게임 애플리케이션과 같이 DirectX가 제공하는 스크린 모드를 사용하지 않아도 되므로 C++가 제공하는 MFC의 클래스를 사용하여 윈도우 형태의 프레임으로 구현할 수 있었다.

6. 결 론

본 논문에서 DirectX API를 기반으로 하는 네트워크 게임 클라이언트 측 구현을 통해서 사용사례를 이용한 요구사항 분석과 GoF의 디자인 패턴을 이용하여 게임과 관련한 새로운 애플리케이션을 개발하거나 기존에 개발된 게임 소프트웨어를 추가적으로 개발하기 쉬운 설계 기법을 제안하였다. 또한 게임 애플리케이션과 그와 관련된 여러 애플리케이션 제작에 GoF의 디자인 패턴 영역 중 생성 패턴이 다양하게 적용될 수 있다는 것을 알 수 있었다.

또한, 게임 애플리케이션 설계에서, 다양한 사용사례를 이용한 요구 사항 분석이 필요한 클래스 추출에 유용하게 쓰일 수 있었다. 그리고 분석된 표와 클래

스 다이어그램이 반복과 정련의 단계를 거치면서 생성 패턴이 적용되었고, 생성 패턴을 적용함으로써 게임 개발에 필요한 추가적인 애플리케이션의 개발이 손쉽다는 것을 알 수 있었다. 어댑터 클래스의 객체 생성과 관련해서 클래스의 응집도를 높이기 위해 팩토리 패턴과 싱글톤 패턴이 적용될 수 있음을 볼 수 있었다.

참 고 문 헌

[1] Frederic Cordier, Hyewon Seo, and Nadia Magnenat-Thalmann, MiraLab, University of Geneva, Switzerland. "Made-to-Measure Technologies for an Online Clothing Store", IEEE Computer society, pp44, January/February 2003.

[2] 최성, "게임 산업과 기술 전망", 정보처리학회지 특집 게임 기술, 제9권 3호, pp. 11-23, 2002.

[3] 김종수, 이종민, 김태석, "생성 패턴을 사용한 네트워크 기반 게임 API 설계", 한국멀티미디어학회 추계학술발표대회논문집, 제6권 2호(하), pp.669-674, 2003.

[4] 김종수, "웹을 기반으로 한 JAVA 네트워크 게임 시스템의 설계와 구현", 석사 논문, 부산외국어대학교. pp. 1-11, 2003.

[5] Erich Gamma, Richard Helm Ralph Johnson, Hohm Vissides 공저, "GoF의 디자인 패턴", Pearson Education Korea. pp. 1-11, 2002.

[6] 남재욱, "온라인 게임 서버 프로그래밍", 한빛미디어, pp. 18-30, 2004.

[7] 안승중, "Direct 3D", 도서출판대림, pp. 61-63, 2000.

[8] 김용준, "3D 게임 프로그래밍", 한빛미디어, pp. 24-26, 2003.

[9] GRAIG LARMAN 원저 "UML과 패턴의 적용" 홍릉과학출판사, pp. 397-402, 2003.

[10] Ricardo Sanz, Janusz Zalewski, "Pattern-Based Control Systems Engineering" Florida Gulf Coast University, IEEE Control Systems Magazine, pp46, June 2003.

김 종 수

1992년 2월 부경대학교 냉동공학과(학사)
1998년 12월 On Line Technology 대표
2003년 2월 부산외국어대학교 컴퓨터공학과 석사
2003년 3월 ~ 현재 동의대학교 소프트웨어공학과 박사과정
2003년 4월 ~ 현재 동의대학교 영상 미디어센터 PM연구원
관심분야 : 네트워크 게임 제작과 설계, Web 프로그래밍

김 태 석

1981년 경북대학교 전자공학과 졸업(공학사)
1989년 일본 KEIO대학 이공학부 계산기과학전공(공학 석사)
1993년 일본 KEIO대학 이공학부 계산기과학전공(공학 박사)
1993년 일본 국제전신전화연구소(KDD) 기술고문
1993년 일본 KEIO대학 이공학부 객원연구원
1994년 ~ 현재 동의대학교 소프트웨어공학과 교수
관심분야 : 정보시스템, 기계번역, 인터넷비즈니스