

객체지향 이론을 적용한 멀티미디어 데이터 처리

김 홍 섭*

Multimedia data processing using object-orient theory

Hong Sop Kim*

요 약

인터넷이 확장되고 멀티미디어의 통합 기술이 발전함에 따라 다양한 방식으로 정보를 표현하고 제공함으로써 컴퓨터 사용자들은 다양한 형태의 데이터를 접하게 된다. 개발자의 관점에서 볼 때 데이터의 처리는 여러 가지 문제를 야기할 수 있다. 사운드, 이미지, 영상 등 다양한 매체와, 같은 매체라도 서로 다른 자료구조로 인한 상호 호환성 문제는 개발자에게 더 많은 작업을 필요로 한다. 최근 대두되고 있는 객체지향 개발방법론은 이런 문제를 효율적이고 효과적으로 해결할 수 있는 기반을 제시한다. 본 고에서는 객체지향 이론의 핵심개념인 상속성과 다형성을 적용하여 효과적으로 멀티미디어 데이터를 처리하는 방법을 제시하고 게임 프로그램 개발에 적용한 그 구현 예를 제안하였다.

Abstract

According as the Internet has expanded, technology of multimedia has developed, information has been expressed and provided in many ways, and users have been faced with various forms of data. However, data process probable has many problems from the developer's point of view.

The problem of compatibility caused by the different data structure in the media and multimedia such as sound, image, video and so forth.

Even if they have the same structure requires the more task to the developers, makes developer work more. The object oriented theory has recently come to the fore as the effectual solution to this problem.

This paper provides how to proceed multimedia data more effectively by using inheritance and polymorphism, which come from the main concept of object oriented theory, and shows the example of then applied to the development of a game program.

I. 서론

인터넷의 발달과 멀티미디어 서비스에 대한 요구확대는 분산환경에서의 멀티미디어 서비스를 증대시키고있다[1].

특히 멀티미디어 데이터를 처리해야하는 개발자의 관점에서 볼 때 텍스트, 사운드, 이미지, 영상 등 다양한 매체와, 같은 매체라도 서로 다른 자료구조로 인한 상호 호환성 문제는 개발자에게 더 많은 작업을 필요로 한다. 객체지향 개발 방법은 이를 극복할 수 있는 뛰어난 기법으로 인식되고 있다[2]. 본 고에서는 객체지향 이론의 핵심 개념인 상속성과 다형성을 이용하여 멀티미디어 데이터를 처리하는 설계과정을 UML표기법으로 제시하고[3, 4], 게임 프로그램 개발에 적용한 사례를 기반으로 멀티미디어 데이터에서 객체지향 이론의 상속성과 다형성 개념 적용의 필요성을 제안하였다.

II. 관련연구

2.1. 상속성과 다형성의 개념

일반적으로 객체 지향적 방법론에서의 객체의 특징들은 주체성(identity), 분류화(classification), 다형성(polymorphism), 상속성(inheritance) 4가지 특성으로 요약될 수 있으며 상호 보완적인 역할을 수행한다[5].

주체성은 데이터가 다른 객체와 구분되는 특성으로 유일하게 참조될 수 있는 것을 의미하며, 분류화는 동일한 자료구조와 행동을 가진 객체가 하나의 클래스(class)로 그룹 지어 지는 것을 말한다. 여기서 클래스란 애플리케이션의 중요한 특성들에 대한 추상화(abstraction)이며 클래스는 캡슐화를 위반할 수 없고, 유일하다[6, 7]. 또한 각 객체는 그 클래스의 인스턴스(instance)가 된다.

다형성은 동일한 연산이 서로 다른 클래스들에서 서로 다르게 동작할 수 있는 것을 의미하며 상속성은 클래스의 메소드와 속성을 상속받아 새로운 클래스를 생성하며, 또한 여러 클래스로부터 상속받음으로써 공통부분의 재사용성을 높일 수 있다[8].

2.2 멀티미디어 데이터 객체 지향적 개념 적용

인터넷과 멀티미디어 통합기술이 발전함에 따라 네트워크 상에는 다양한 형태의 데이터가 존재하며 기존의 단순한 사운드, 이미지 정보의 이용에서 점차 음성과 이미지가 결합된 멀티미디어 정보를 이용하는 추세로 확장되고 있다. 또한 멀티미디어 구성요소 데이터는 문자코드, 이미지픽셀, 오디오 샘플의 시퀀스와 같은 실제 데이터(raw data)와 실제 데이터를 화면에 출력하거나 처리하기 위해 필요한 화면 해상도, 픽셀당 비트 수, 색 표현 방식 등의 등록 데이터(registration data), 해당 미디어 특징, 구조, 제목 등을 기술한 설명 데이터(description data)로 구성되어 다양한 구조와 포맷으로 존재하며 따라서 멀티미디어의 데이터 처리가 복잡해지고 이의 처리 방법에 있어서도 다양한 방법을 써야 한다[9].

그러나 사용자는 멀티미디어 내부 구조와는 상관없이 간단한 작동을 통해 멀티미디어 데이터를 사용하고 싶어 하며 이를 위해서는 적절한 개발 방법이 필요하다.

그런 점에서 객체 지향적 방법론은 이러한 환경의 변화와 멀티미디어 데이터 처리 요구를 해결할 수 있는 해결책을 제시할 수 있다.

2.3 객체에 대한 상속성과 다형성의 개념

상속성은 동일한 기원(선조)을 가진 다양한 데이터(객체)들에 대해 동일한 문맥(context)으로 데이터를 처리할 수 있는 기능을 제공할 수 있게 한다.

예를 들어 Bitmap 과 GIF file은 이미지 파일이지만, 서로 다른 구조를 가지고 있다. 그러나 이들은 상속성과 합수 오버로딩을 통하여, 그리기라는 기능인 draw()를

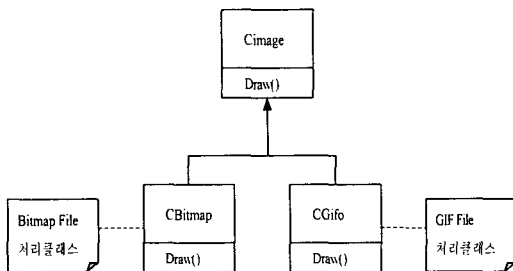


그림 1. 상속성을 이용한 다양한 이미지 처리 클래스

제공할 수 있고, 이는 파일 포맷에 따라 적절하게 반응하지만, 사용자는 동일한 문맥인 그리기라는 개념으로 작업을 시킬 수 있다. 상속성을 이용한 다양한 이미지 처리 클래스 예는 다음과 같다.

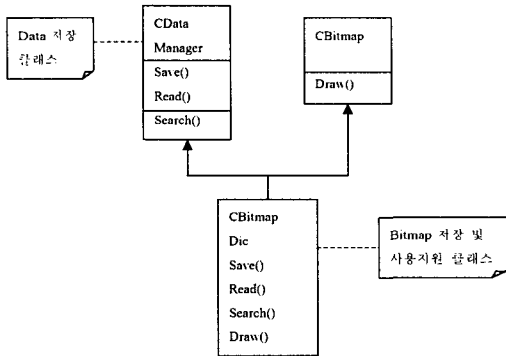


그림 2. 다형성을 이용한 멀티미디어 DATA 사전

한편 다형성은 서로 다른 문맥과 목적을 가진 객체들을 특정한 목적을 위해 융합시키는 개념이다. 멀티미디어 데이터 처리에서의 적용은 여러 가지 장점을 갖는다. 가령, 멀티미디어 편집과 플레이 기능을 가진 객체와 자료 보관 및 검색 기능을 가진 객체를 융합하여 멀티미디어 데이터를 편집, 플레이와 검색 보관 등을 모두 가능하게 하는 다형성을 이용한 멀티미디어DATA 사전을 생성할 수 있다[10].

객체 지향적 개념인 상속성과 다형성을 적용하면, 다양한 멀티미디어 데이터 처리에 대해서 보다 효과적이고 적절한 기능들을 만들어내고 추가할 수 있으며, 기존의 클래스를 재사용하여 작업을 최소화시킬 수 있게 해준다[11].

Ⅲ. 시스템 설계

3.1 게임 소프트웨어를 통한 개념 적용

게임은 이미지, 동영상, 배경 음악, 효과음 및 텍스트 등 다양한 멀티미디어 데이터 처리가 필요한 대표적인 사례이다.

객체 지향적 상속과 다형성은 이러한 게임 소프트웨어의 멀티미디어 데이터 처리를 위한 핵심 엔진을 구성하는데 있어서 모범적인 솔루션을 제공한다.

3.2 데이터 저장과 검색을 위한 객체 구성

데이터 저장 구조는 데이터를 보존 기능과 효율적인 검색 기능을 동시에 수행해야 하며, 저장될 데이터는 특정한 포맷으로 미리 정할 수 없으므로, 이러한 객체는 저장 방법과 검색 방법만을 가지고 있을 뿐이다. 데이터 저장 및 검색 객체는 환경에 따라 다양한 저장 형태와 검색 전략을 구사할 수 있어야 한다.

상속성을 이용하여 다양한 저장 형태나 검색 전략을 구현할 수 있다. 예를 들어 선조 클래스에서 search()라는 기능으로 순차적 탐색 전략을 사용하고 있으나, 이진 검색 방법이 필요할 경우, 후손 클래스는 선조 클래스를 상속하면서, search()기능만 적절한 이진 기능으로 교체할 수 있다. 이때, 다른 기능들은 모두 재사용이 가능하므로, 선조가 가진 기능을 구현하기 위해 다시 프로그램을 작성할 필요는 없다. 저장이나 관리에 관해서도 동일한 개념이 적용될 수 있다.

또한 다형성 적용의 경우에는 linked list로 구성된 인덱스를 지원하는 객체와 이진 트리로 된 인덱스를 지원하는 객체가 따로 존재할 때, 이들을 동시에 상속하므로써 linked list와 이진 트리를 모두 사용할 수 있는 후손을 만들 수 있다.

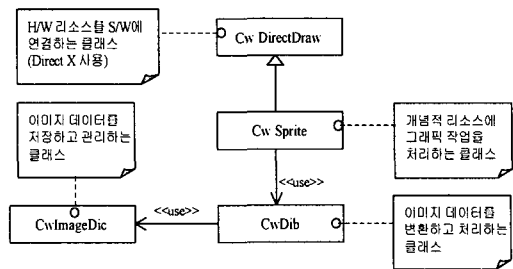


그림 3. 상속성과 다형성 적용의 linked list 구조

3.3. 이미지 처리를 위한 객체 구성

이미지 처리는 정지 화상의 디스플레이에 관련된 기능들과, 동영상 처리를 위한 기능들로 구성된다. 따라서 이 객체는 다양한 이미지 포맷을 읽어들이고 디스플레이하기 위한 형태로 변환하는 기능들을 갖추어야 한다. 동시에 하드웨어 상황(해상도나 픽셀 포맷)에 따라 적합한 변환을 위한 기능들이 필요하다.

이를 위한 구성방법은 다음과 같다. 먼저 하드웨어 처리를 위한 중간 단계 처리 객체를 두고 그 객체를 바탕으로, 특정 이미지 포맷을 전문적으로 처리하는 클래스(일반 상속) 및 클래스들을 조정하는 클래스(다형성, 상속성 이용)를 구성한다.

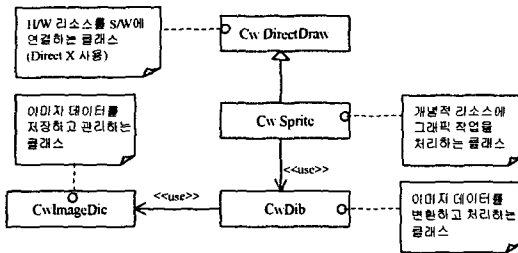


그림 4. 이미지 처리를 위한 클래스 구성

3.4. 음성 및 음악 Data 처리를 위한 객체 구성

특정 사운드 파일 포맷을 음성(음악) 플레이를 위한 변환 기능들을 갖춘 객체이다. 이미지 데이터와 마찬가지로 하드웨어 환경이나 다양한 포맷에 따라 적절한 변환을 수행하는 기능들을 추가할 수 있어야 한다. 구성 방법은 이미지 사전과 같다.

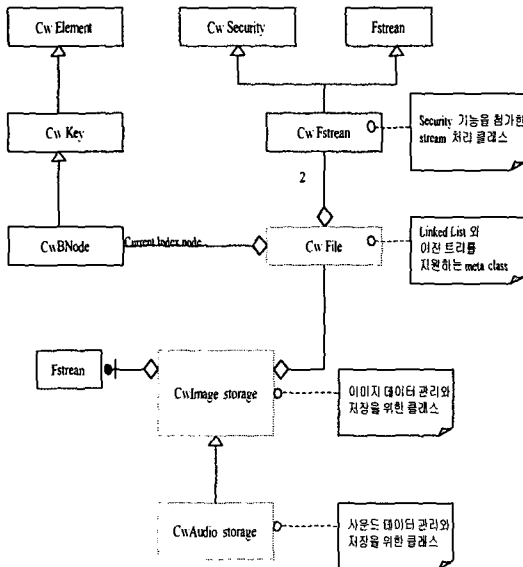


그림 5. Storage Manager Group 관련 주요 객체모델

3.5. 게임용 다목적 멀티미디어 사전 구성

게임에서 사용되는 이미지, 사운드 및 텍스트 데이터를 보관하고, 검색하며, 동시에 용도에 따라 디스플레이 하거나 플레이할 수 있도록 하는 환경을 지원하는 기능을 가진 객체를 구성해야 한다. 저장 및 검색 전략을 구현한 객체와 이미지, 사운드, 텍스트 등을 처리하는 객체들을 동시 상속하여 이미지 사전, 사운드 사전, 텍스트 사전 등을 구현한다.

IV. 클래스 생성

4.1 멀티미디어 사전을 위한 클래스 정의

4.1.1 CWFile 클래스의 개요

다양한 멀티미디어 기본 데이터를 데이터 관리 알고리즘에 융합할 수 있도록 클래스 자체를 템플릿으로 구현하며(메타 파일의 C++ 구현 방법), 검색과 저장 전략은 데이터 형태와 무관하므로, 동일한 전략을 다양한 멀티미디어 데이터에 적용할 수 있도록 클래스를 구성한다. 특정 멀티미디어 데이터 전용 사전을 구현하기 위한 클래스를 만들 때, 선조가 될 수 있도록 다양한 기능의 프로토타입을 멤버 함수로 제공하며, 효율적인 검색 기능을 지원하기 위해 내부 자료를 최적화 하는 기능을 제공한다 [12, 13].

```

template <class CWRecFamily> class CWFile
(
public:
    // 생성자와 소멸자 관련 멤버 함수들
    CWFile(){ initialize(); };
    CWFile( char *i_pchFileName, int
    i_nKeySize, int i_nIndexFlag ) throw( int );
    virtual int create( char *i_pchFileName,
    int i_nKeySize, int i_nIndexFlag=0 );
    virtual ~CWFile();
    // 데이터 검색 및 저장과 관리를 위한 함수들
    virtual int search( const char * i_pchKey );
    virtual int add( CWRecFamily & i_Rec );
    virtual int read();
    virtual int update( CWRecFamily & i_Rec );
    virtual int remove( const char * i_pchKey );
    ...
    // 데이터 노드 사이를 운항하는 기능
    virtual int previous() throw( int );
    virtual int next() throw( int );
    virtual void goTo( long i_nPos );
    
```

```

virtual int goRoot() throw( int );
virtual int goUp() throw( int );
virtual int left() throw( int );
virtual int right() throw( int );
....
};
    
```

4.1.2 이미지 저장 사전 메타 클래스의 정의

이미지 저장 사전 구현을 위해 CWFile을 상속하여 데이터 관리와 저장 검색 전략을 모두 활용하며 이미지 데이터 처리를 위한 전문적 기능들을 추가적으로 구현한다. 또한 미래에 추가될 다양한 이미지 포맷들을 처리하는 후손 클래스를 위해 주요 멤버 함수들을 virtual로 선언하였으며 다양한 이미지 데이터를 지원하기 위해 역시 템플릿으로 구성하였다.

```

template <class _cwImageIndex> class cwImageStorage
{
public:
    // 클래스 생성 및 소멸 관련 기능
    cwImageStorage() {initialize();};
    cwImageStorage(char *i_pchFileName, int
    i_nKeySize, int i_IndexFlag );
    ~cwImageStorage();
    virtual int create(char *i_pchFileName, int
    i_nKeySize, int i_IndexFlag);
    // 이미지 데이터 저장, 관리 및 검색을 위한
    overload된 기능들
    virtual int add( _cwImageIndex & i_AI,
    char * i_pchWaveFileName );
    virtual int add( char * i_pchKey, char *
    i_pchWaveFileName );
    int search( char * i_pchKey, char **
    o_ppchMem, DWORD & o_nSize );
    int update( char * i_pchKey,
    _cwImageIndex & i_AI );
    int remove( char *pchKey );
    ...
    // 이미지 레코드 인덱스를 위한 구조
    CWFile<_cwImageIndex> m_KeyStorage;
    ...
};
#endif
    
```

4.1.3 계층구조를 통한 상속성 및 다형성의 강화

객체지향 시스템의 개발은 클래스를 통해서 이루어진다. 즉 문제 영역에 존재하는 실제계의 대상 또는 개념을 클래스로 모델링하고, 이로부터 생성된 객체들 사이에 메시지 교환을 통해서[14], 즉 특정 데이터 구조를 전문적으로 처리하는 클래스들과, 게임 시나리오나 운영 및 하드웨어 환경과의 문제를 중재하는 클래스들을 각각 독립적인 계층으로 분리하여, 서로 연합시킴으로써, 다른 계층의 변화에 독립적이면서도 동시에 안정적인 시스템을 구축할 수 있다.

또한 계층 구조는 클래스들의 상호 연결 규칙 및 전략을 결정하게 하는 것으로 장기적인 소프트웨어 진화에 중요한 역할을 할 수 있다.

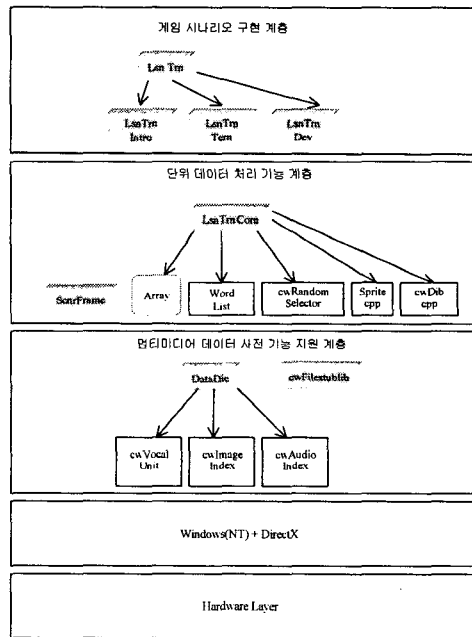


그림 6. 클래스 계층 구조

V. 결론

게임 소프트웨어는 다양한 멀티미디어 데이터를 다양한 시나리오와 결합해야 하는 문제를 항상 안고 있다.

때문에 기존의 방법론에 의한 게임 프로그램의 개발은 프로그래머들에게 비교적 많은 작업을 요구한다. 그러나 객체 지향적 방법론을 사용하였을 경우 실험한 예와 같이 코드의 재사용성과 이식성이 높아지고 동시에 유연하게 주어진 상황에 대처할 수 있기 때문에 비교적 최소의 노력으로 안정적이고 효과적인 개발을 할 수 있었다.

멀티미디어 데이터는 지속적으로 발달하면서 계속 새로운 형태의 처리 기술과 포맷들을 요구한다. 따라서 환경에 맞는 적절한 기능을 제공하면서, 동시에 이전에 제공된 기능들 중 현재에도 가치 있는 것들도 지속적으로 제공하여야 한다. 이러한 상황에서 상속성은 보존과 기능의 진화적인 확장을 구현하는 개념적 기반을 제공한다. 이는 기존의 클래스를 상속하여, 부가적인 기능을 적절하게 제공하게 함으로써, 소프트웨어의 안정적인 진화를 가능하게 하는 요소이다.

또한 다형성을 이용하여 멀티미디어 사전의 경우에서 처럼, 특정 목적을 위해 만들어진 객체들을 동시 상속하여, 복잡한 기능들이 조합된 솔루션을 구성할 수 있다.

상속성과 다형성은 프레임워크나 아키텍처 개념과 결합하여 거대 시스템이나 소프트웨어 핵심 엔진의 진화를 구현하는 장치를 제공할 수 있다.

VI. 참고 문헌

[1] J.F.K. BufordEds., Multimedia Systems, ACM Press, 1994..

[2] <http://cs.chungnam.ac.kr/~cmsong/100/methodology/ooad-cyoun/ooa-ood.htm>, 객체지향 분석 및 설계 기법.

[3] Grady Booch, James Rumbaugh, Ivar Jacobson, "The Unified Modeling Language User Guide", Addison Wesley, 1999.

[4] 윤희진의 3인, "UML 기반 콤포넨트 통합 테스트", 정보과학회논문지(B) vol.26, NO.9, pp.1105, 1999.

[5] James Rumbaugh, Michael Blaha, William Premeriani, Fredrick Eddy, William Lorensen,

Object-Oriented Modeling and design, Prentice-Hall, 1991.

[6] 김도형, 정기원, "객체지향 분석과정에서 오류와 일관성 점검 방법", 정보과학회논문지(B) vol. 26, NO.3, pp.380, 1999.

[7] 신행렬, 최은만, "객체 지향 소프트웨어 매트릭스를 근거로 한 품질 평가 도구의 설계 및 구현", 정보과학회 가을 학술발표논문집, vol.25, NO.2, pp.460, 1998.

[8] 김기환외 3인, "객체모델링에서 관련성의 코드 생성에 관한 연구", 정보과학회 가을 학술발표논문집, vol.25, NO.2, pp.597, 1998.

[9] 멀티미디어 개론, 광성근, 도서출판 글로벌, 1999.

[10] 한번 더 생각한 객체지향 모델링과 구현, 김창완, 도서출판 대림, 1998.

[11] 김재생, 송영재, "클래스의 재사용을 위한 정보 분석 및 품질 척도", 정보과학회논문지(B) vol.26, NO.3, pp.393, 1999.

[12] Richard C. Lee, and William M. Tepfenhart, "UML and C++", Prentice Hall, 1997.

[13] bjarue stroustrup, "The C++ programming language 3rd edition", Addison Wesley, 1997.

[14] 채홍석외2인, "객체지향 시스템의 클래스에 대한 응집도", 정보과학회논문지(B) vol.26, NO.9, pp.1095, 1999.

저 자 소 개



김 홍 섭

1984년 3월 동국대학교 학사 (전산원)

1994년 3월 동국대학교 석사 (전산교육)

1997년 7월 동국대학교 박사과정(컴퓨터 공학)

1994년 9월 오산대학 정보관리과 조교수(현)