멀티미디어 응용의 효율적 프리젠테이션을 위한 이종 객체 처리에 관한 연구

이규남* · 나인호*
*군산대학교

A Study on Management of Heterogeneous Object for Effective Presentation of Multimedia Application

Kyu-nam Lee* · In-ho Ra*
*Kunsan National University
E-mail : knlee@kunsan.ac.kr

요 약

본 논문은 효율적인 프리젠테이션을 위한 이종 객체의 처리 방안에 대하여 설계를 중심으로 기술하였다. 이종 객체의 개별적인 특징을 살리기 위하여 각 미디어별로 세분화된 클래스를 두었다. 특히 멀티미디어 프리젠테이션을 위한 저작 과정에서 세분화된 각 클래스를 사용자 정의가 가능한 각각의 객체와 통합적으로 처리할 수 있는 방법을 제안하였다. 본 논문에서 제안한 데이터 구조와 처리 방법은 향후 또 다른 특화된 클래스 정의를 추가함으로써 손쉽게 데이터 객체의 처리 기능을 확장할 수 있도록 설계되었다.

ABSTRACT

In this paper, we describe a design method to manage heterogeneous objects for effective presentation of multimedia application. In order to utilize the individual feature of a heterogeneous object, we define a specialized class for each media object, and it also gives a method to process each specialized class combined with various figures that can support user-definition during the authoring phase for a multimedia presentation. The proposed data structures and processing methods are designed to easily extend the functions for handling a data objects by adding another specialized class definitions to existing object classes.

키워드
Multimedia, Presentation, Authoring Tool

I. 서 론

과거 모노 미디어(mono media)를 중심으로 이루어졌던 컴퓨터의 환경이 현재는 거대한 양의 데이터와 그러한 데이터들이 혼합되어 새로운 정보와 가치를 창출하는 멀티미디어(multimedia) 시대가 사회 전반에 도래하였다. 특히, 텍스트, 이미지, 오디오, 비디오 등 여러 미디어 고유 특징을 활용하여 다양한 형태의 새로운 멀티미디어 컨텐츠를 생성하고, 이를 로컬(local) 혹은 네트워크를 이용한 기술의 개발에 관심이 집중되고 있다.

멀티미디어 서비스는 다양한 모노 미디어의 시공간적 조합에 대한 프리젠테이션 과정이라 볼 수 있다. 또한 현재의 컴퓨터 환경에서 파일의 형태로 저장되어 있는 디지털 미디어뿐만 아니라 사용자가 원하는 정보를 효율적으로 표현하기 위하여 도형을 비롯한 다양한 객체를 멀티미디어 프리젠테이션에 활용하고 있는 추세이다. 이러한 형태의 복잡 다양한 프리젠테이션을 위한 멀티미디어의 구성은 이종의 데이터를 다루어야 되기 때문에 그러한 객체들의 생성 및 관리에 있어 많은 고려가 필연적이다. 이에 본 논문은 다양한 파일의 형식으로

존재하는 대표적인 디지털 미디어들과 사용자가 정의할 수 있는 선(line)을 비롯한 도형들의 각각의 특징을 살리는 장점을 취하면서 그 관리 및 취급은 용이하게 할 수 있도록 일반화 할 수 있는 데이터의 설계와 그 운용에 대하여 다루었다.

본 논문에서는 먼저 멀티미디어 프리젠테이션에 대해 기술한 뒤 이종의 미디어 객체들을 일반화하여 다루면서도 그 개별적인 특징을 프리젠테이션에 충분히 활용할 수 있는 데이터 구조의 설계와 처리 방법에 대하여 기술하였다.

II. 멀티미디어 프리젠테이션

멀티미디어란 여러 가지 모노 미디어가 혼합된 미디어로 정의할 수 있다[1]. 이러한 멀티미디어를 구성하고 있는 모노 미디어는 시간과의 관련성 여부에 따라 이산 미디어(discrete media)와 연속 미디어(continuous media)로 분류할 수 있다[1,2,3]. 이산 미디어 혹은 정적 미디어(static media)는 텍스트와 이미지처럼 미디어 자체에 시간의 개념이 없어서 한번 출력하면 사용자의 조작이 가해지지 않는 한 원래의 상태를 그대로 유지하게 된다. 반면, 연속 미디어 혹은 동적 미디어(dynamic media)는 사운드나 동화상과 같이 미디어 자체에 재생(play-back) 시간이 포함되어 있으며, 재생한 이후에 자체의 시간에 따라 동기(synchronize)되어 출력되는 미디어를 말한다.

텍스트를 비롯한 이미지, 사운드, 동화상은 현재 멀티미디어 프리젠테이션에 사용되는 대표적인 미디어라 할 수 있다. 이러한 미디어는 또 각각 수많은 종류의 서로 다른 데이터 형식(format)으로 구성되기 때문에 그 처리 방법이 매우 상이한 문제가 있다. 그럼에도 불구하고 멀티미디어 프리젠테이션은 이러한 모노 미디어를 시·공간적으로 그 관계를 정의하여 종합적으로 처리하므로써 프리젠테이션을 진행하게 된다. 여러 모노 미디어간의 시간 관계를 정의하는 것은 미디어들의 출력, 지연, 그리고 종료시간의 관계를 정의하는 것이며, 이를 시간적 조합(temporal composition)이라 한다. 미디어들의 공간을 조합하는 것은 모노 미디어의 출력될 위치를 배치하는 것으로써 공간적 조합(spacial composition)이라 한다[1,4,5].

하지만 일반적으로 멀티미디어의 프리젠테이션은 단순한 모노 미디어들의 시공간적 조합으로만 이루어지지 않으며, 사용자의 효율적인 프리젠테이션을 위하여, 시나리오 제작 과정에서 사용자가 선을 비롯한 사용자 정의 가능한 여러 도형을 배치하고 효과(effect)를 추가하여 프리젠테이션 데이터를 작성하는게 일반적이다. 따라서 이렇게 구성된 프리젠테이션 멀티미디어 데이터는 여러 미디어와 각종 도형, 그리고 이것들에 부여된 프리젠테이션 효과의 유기적인 동작을 최대한 유지할 때에 만족할만한 QoS(Quality of Service)를 얻을 수 있다.

따라서, 프리젠테이션에 사용되는 여러 모노 미디어와 사용자 정의 객체, 그리고 이에 부여되는 프리젠테이션 효과를 함께 고려하는 것은 멀티미디어 저작에 있어 매우 중요한 부분이며, 실제 이러한 개별 데이터를 다루는데 있어 그 데이터의 상이함으로 인해 많은 어려움이 있다.

본 논문에서는 다양한 형태의 미디어 및 객체와 효과를 일관된 형식으로 다루면서, 그 각각의 데이터 특징 및 장점을 활용할 수 있도록 하는 이종 객체 처리 방안에 대하여 주요한 데이터 구조를 설계하고 그 이용에 대해 기술하였다.

III. 미디어와 객체의 관리 방법

본 논문에서는 이러한 여러 종류의 미디어를 비롯한 이종 객체들을 함께 처리할 수 있도록 할 수 있는 클래스 계층을 다음과 같이 설계하였다.

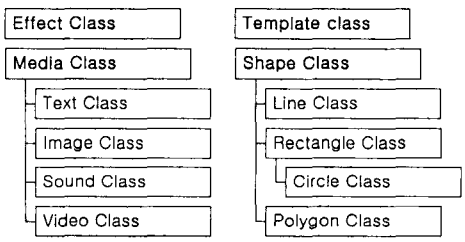

그림 4. 클래스 계층 구조

1. Template Class

클래스 템플릿(class template)이란 서로 다른 데이터형을 처리할 수 있는 일반화된 데이터 구조를 정의하는 경우에 이용하는 C++의 중요 기법으로써 프리젠테이션에 이용될 다양한 형식의 미디어와 사용자 정의 객체의 특징을 유지하면서 일반화할 수 있는 방안을 제공한다.

```
template <class Type>
class CTemplate
{
public:
                :
        CTemplate(const CTemplate<Type> &Object);
        ~CTemplate();
private:
        Type *m_pMedia;

public:
                :
        BOOL OpenMedia(const char *pPath);
        BOOL PlayMedia(void);
        BOOL PauseMedia(BOOL bPause);
        BOOL StopMedia(void);
        BOOL CloseMedia(void);
                :
}
```

그림 5. Template Class의 정의

C++의 클래스 템플릿 기법을 사용하여 "Template Class"를 설계하였으며, "Template Class"는 "Media Class"와 "Shape Class"의 인스턴

트를 저장할 수 있는 클래스 포인터를 속성으로 포함하도록 하였다. 그림 2에 "Template Class"의 설계된 정의를 보였다.

"Template Class"는 "Media Class"와 "Shape Class"를 일반화해야 하기 때문에 " Media Class"와 "Shape Class"의 주요 인터페이스를 제공하도록 하였으며, 프리젠테이션 과정에서 이렇게 설계된 "Template Class"를 통하여 실제 인스턴트(instance)의 메소드(method)와 통신하도록 하였다. "Type *m_pMedia" 변수는 사용자 정의 도형(shape)과 저장된 미디어의 실제 인스턴스를 처리할 수 있도록 하였다. 그림 3에 "BOOL PlayMedia(void)" 메소드에 대한 정의를 나타내었다.

```
template <class Type>
BOOL CTemplate<Type>::PlayMedia()
{
        return m_pMedia->PlayMedia();
}
```

그림 6. PlayMedia() 메소드 정의

2. Media Class

"Media Class"는 텍스트와 이미지, 사운드와 비디오 등 저장된 디지털 미디어에 대한 상위 클래스로 설계하였다. 저장된 디지털 미디어는 여러 가지 공통적인 특징을 가지고 있으며, 예를 들면 파일의 이름과 같은 파일 정보들이다. 이러한 파일 속성에 대한 처리를 위하여 "Media Class"는 파일 포인터 등을 가지고 있으며, 파일의 개방과 닫기 연산을 수행할 수 있도록 하였다. "Media Class"의 특징을 상속받는 하위 클래스들은 부모 클래스의 특징을 모두 담고 있기 때문에 자식 클래스에 대하여 부 클래스는 공통의 일반화된 특징을 다룰 수 있도록 설계될 수 있다.

3. 기타 미디어 클래스

"Text Class", "Image Class", "Sound Class", "Video Class"는 이종의 개별 미디어를 처리 가능하도록 하였다. "Text Class"는 텍스트 문서에 대하여 특화된 클래스로, 일반 ASCII로 구성된 서식없는 텍스트를 비롯하여 서식 있는 포맷 등을 다룰 수 있도록 하위 클래스를 세분화하였다. " Image class "도 비트맵 파일을 중심으로 파일의 개방과 출력 크기 조절 등의 연산을 할 수 있도록 하였다. 특히 압축된 형식의 JPEG와 GIF의 데이터 포맷을 처리할 수 있도록 하였다. 동적 미디어인 사운드와 비디오의 데이터는 "Sound Class"와 "Video Class"로 각각 특화하였으며, 이들은 다시 지원 가능한 파일 포맷별로 세분화된 클래스 정의를 가지고 있도록 하였다. 이렇게 여러 가지로 구분될 수 있는 각각의 미디어 포맷은 여러 식별자를 정의하여 구분하도록 하였다.

그림 4의 식별자를 이용하여 "Media Class"에서 실제 인스턴스의 종류가 무엇인지를 구별할 수 있도록 하였으며, 이 구분된 종류는 "Template Class"에서 개별적인 미디어 인스턴스의 특징을 최대한 활용할 수 있는 인터페이스를 제공하도록 하였다.

```
enum MEDIATYPE
{
        UNKNOWN_TYPE            = 0x00,
        MPEG_VIDEO_TYPE         = 0x11,
        AVI_VIDEO_TYPE          = 0x12,
                :
        CD_AUDIO_TYPE           = 0x21,
        WAVE_AUDIO_TYPE         = 0x22,
        MIDI_SEQUENCE_TYPE      = 0x23,
                :
        BMP_GRAPHIC_TYPE        = 0x31,
        JPG_GRAPHIC_TYPE        = 0x32,
        GIF_GRAPHIC_TYPE        = 0x33,
                :
        ASCII_TEXT_TYPE         = 0x41
        RTF_TEXT_TYPE           = 0x42,
        TWD_TEXT_TYPE           = 0x43,
                :
};
```

그림 7. 이종의 미디어 타입 정의

4. Shape Class.

"Shape Class"는 일반 멀티미디어 응용에서처럼 프리젠테이션의 사용자 정의 도형 기능의 클래스이다. 이러한 도형에는 선, 사각형, 원, 다각형 등으로 크게 구분할 수 있으며, 점과 선, 선의 속성과 면의 속성 등 공통의 특징을 찾을 수 있다. 이러한 공통의 부분에 대한 처리를 "Shape Class"에 포함시키고, 자식 클래스에는 도형의 모양에 따른 특징을 처리하도록 설계하였다. 그림 5는 " Shape Class"의 정의를 나타내고 있다.

```
class CShape
{
public:
        ////////// 여러가지 생성자, 그리고 소멸자
        CShape();
                :
        virtual ~CShape();

protected:                      :
        ////////////////////// 선과 관련된 멤버
        CPen m_penLine;
        COLORREF m_clrLineColor, …;
        int m_nLineStyle, m_nThickness,…;
        ////////////////////// 면과 관련된 멤버
        CBrush m_brushFill, …;
        COLORREF m_clrBrushColor, …;
        int m_iFillStyle, m_iTransparency, …;
public:                 :
        EShapeClass WhatKindOfShape(void) const;
        virtual void MoveTo(const CRect &Rect);
        virtual void Resize(const CRect &rect);
        virtual void Draw(CDC *pDC);
                :
}
```

그림 8. Shape Class의 정의

그림5와 같이 도형은 여러 속성으로 나타낼 수

있으며, 이러한 속성은 도형의 모양에 관계없이 동일하다는 특징이 있다. 또한 도형의 모양을 규정하는 것은 점과 점을 연결하는 선분이며, 점의 속성을 다양하게 함으로써 기본 도형으로부터 확장된 여러 가지 모양의 도형을 다룰 수 있다.

5. Effect Class

"Effect Class"는 앞서 기술한 여러 가지 디지털 미디어와 사용자 정의 객체인 "Shape Class"에 시간 특성과 효과를 정의할 수 있는 데이터이다. 사용자는 각각의 개별 미디어와 정의된 도형의 시·공간적인 조합으로 프리젠테이션할 새로운 멀티미디어를 정의하게 되는데 여러 가지 미디어와 객체에 개별적인 특수 효과를 부여할 수 있도록 " Effect Class"를 두었다. "Effect Class"는 사용자의 프리젠테이션 될 미디어를 비롯한 이종 객체의 개별적인 효과를 추가할 수 있도록 하였다.

그림 6은 이러한 용도를 고려한 "Effect Class"의 효과 종류를 나타내었다.

```
enum EFFECT
{
        EFFECT_SHOW        = 0x01,
        EFFECT_HIDE        = 0x02,
        EFFECT_OVERLAY     = 0x03,
        EFFECT_MOVETOLEFT= 0x04,
        EFFECT_MOVETOTOP = 0x05,
                :
        EFFECT_INVERSION   = 0x10,
        EFFECT_DIVIDE      = 0x11,
        EFFECT_BLINK       = 0x12,
        EFFECT_SCROLL      = 0x13,
        EFFECT_CLOSEUP     = 0x14,
                :
};
```

그림 9. 지정 가능한 효과의 종류

이렇게 정의된 식별자는 "Effect Class"의 정의에서 개별적인 이종의 객체에 대한 여러 가지 효과를 구분하는데 사용된다. "Effect class"의 대표적인 속성으로는 효과의 종류와 시작 및 종료 시간 등이고, 중요한 연산으로는 효과의 실행과 관련한 연산 등이며, 그림 7과 같다.

```
class CEffect
{
protected:
        EFFECT m_eEffectType;
        TIME m_tStartTM, m_tEndTM
        UINT m_uRepeat, …;
                :
public:
        BOOL Play(void) const;
        BOOL Pause(void) const;
        BOOL Stop(void) const;
                :
};
```

그림 10. Effect Class의 정의

"Effect Class"는 프리젠테이션을 위한 미디어의 시·공간 속성을 정의하는 과정에서 개별 미디어와 연결되도록 하여 사용자의 프리젠테이션 진행 명령에 의해 "Effect Class"의 데이터를 기반으로 프리젠테이션 스케줄링 및 프리젠테이션 진행을 할 수 있도록 하였다.

이러한 이종 데이터를 일반화하는 기법은 프리젠테이션 과정에서 각 데이터에 독립적인 프리젠테이션이 가능하게 하며, 세부적인 각 미디어 및 객체의 특징은 특화된 각 클래스에서 처리하도록 함에 따라 효율적인 프리젠테이션을 위한 멀티미디어 컨텐츠를 개발할 수 있도록 하였다.

IV. 결 론

본 논문은 효율적인 프리젠테이션을 위한 멀티미디어 저작을 위해 이종 객체의 처리 방안에 대하여 설계를 중심으로 기술하였다. 이종 객체의 개별적인 특징을 유지하기 위하여 각 미디어별로 세분화된 클래스를 설계하였으며, 그러한 클래스와 사용자가 정의한 여러 도형을 지원할 수 있는 객체의 통합된 처리가 가능하도록 하였다. 여러 종류 객체와 이종 미디어의 특성을 살리면서 프리젠테이션의 데이터 형식에 독립된 처리가 가능하도록 하기 위하여 C++의 클래스 템플릿을 응용한 "Template Class"를 두어 세부적으로 특화된 객체를 일반화 할 수 있도록 고려하였다. 특히, 멀티미디어 프리젠테이션에 사용되는 개별 미디어의 여러 가지 특수 효과를 나타내기 위하여 "Effect Class"를 두어 멀티미디어의 시?공간 조합 과정에서 개별 미디어의 효과에 따른 시간 등의 속성을 정의하도록 하여 각 미디어의 다양한 프리젠테이션 특성을 표현할 수 있도록 하였다.

본 논문에서 설계한 데이터 구조와 처리 방안은 향후 또 다른 종류의 미디어 객체의 특화된 클래스 정의를 추가함으로써 일반화 "Template Class"를 활용하여 손쉽게 처리 데이터를 확장할 수 있도록 고려하였다.

향후 본 논문에서 설계된 데이터와 방법을 기반으로 실제 저작에 응용할 수 있는 구현을 계획하고 있으며, 이종의 객체를 효과적으로 프리젠테이션 할 수 있는 프리젠테이션 방안에 대하여 함께 연구할 예정이다.

참고 문헌

[1] T.D.C. Little, A.Ghafoor, "Distributed Multimedia Objects Composition and Communication", IEEE Network Magazine, pp.72-84, Nov.1990

[2] B. Prabhakaran, S. V. Raghovan, "Synchronization Models for Multimedia Presentation with User Participation", Multimedia Systems Springer-Verla|, pp. 53-62, 1994.

[3] Y.Ishibashi, S.Tasaka, Y.Tachibana, "Adaptive Causality and Media Synchronization Control for Networked Multimedia Applications", 2001 IEEE International Conference on Communications, Vol 3of10, pp. 952-958, 2001

[4] Y.Kato, K.Hakozaki, "Application QoS Management for Distributed Computing Systems", 2001 IEEE International Converence on Communications, Vol 4of10, pp.1201 -1205, 2001

[5] S. Ricarel, W. Tonathen, M. David, "Actuality-of- Service Specification for Multimedia Presentation System", Springer -Verlag, pp. 251-263, 1995