1R−9

A Framework for Supporting Multimedia in Distributed Object Server

Tong-Haing FIN, Kouichi KAJITANI
Tokyo Research Laboratory, IBM Japan Ltd.

1 Introduction

Recent advances in hardware support for continuous media (digital audio and video) and in high-speed communication technology have led to the new and exciting innovation of mixing text, graphics, audio, and video (multimedia) with a computer. Computer-supported multimedia applications are just emerging and they include multimedia encyclopedia, video editing, simulation, adventure games, and learning aids. However, the development of multimedia applications is presently hampered by two key problems. First, multimedia involves novel concepts to many users and developers; they will start with conventions that they understand - books, films, computer games, audio and video equipment - and then discover new ways to use the multimedia environment. Second, today multimedia support typically requires special hardware, real-time handling of huge digital data, and special support for synchronizing different media types, leading to lack of portability and difficult development.

One way of alleviating these two problems is by providing programmers with a high level programming model for virtualizing multimedia devices and for handling multimedia data. In this paper, we will describe an object-oriented framework for supporting multimedia applications in distributed environment.

2 Multimedia Object Model

The most important requirements of a distributed multimedia application are the integration of multimedia data retrieved from databases distributed across a network, and the handling and synchronization of multimedia data. To fulfill these requirements, we use the concepts of object, composition, and connection as the base for our multimedia model.

Most of as are familiar with Lego^{TM}, where structures of any shape can be composed from a set of basic building blocks and those structures can in turn be combined to form larger structures and so on. The resultant group of structures can be manipulated as a unit. This is the basic idea of *composition*.

In real life, to play/record music or video clips, home users manipulate buttons or *connect* cables of audio/video equipment. This familiar way of making analog signal (media data) flowed from device to device leads to the intuitive use of *connection* as the mechanism to support multimedia data.

In our model, we use an *object-oriented approach* to encapsulate multimedia data and its processing, *object composition* as the mechanism for structuring and organizing multimedia information (objects) in applications, and *object connection* to support multimedia data handling and synchronization.

Media Devices

In our framework, we provide a device-independent access to audio adapter, CD-ROM device, videodiscs and other real hardware devices through the concept of *media device*.

Media devices are the logical representation of the function available from either a real hardware device, software emulation in combination with real hardware, or pure software emulation.

Frequently, there is a one-to-one correspondence between a real hardware device, such as a CD-ROM drive and its associated *CD-ROM media device*. Other hardware may be represented as multiple media devices. An example in this category is a multi-function audio adapter, which can be represented as a waveform audio, MIDI sequencer, and amplifier mixer media devices.

The following shows some examples of media devices:

- Waveform Device
- MIDI Sequencer
- CD-DA/CD-XA Device
- Speaker
- Analog Video Window
- Digital Video Window

Media Object

Media object is an abstraction for processing and storing *media data*. Each media data of a media object is of some *media type*, such as waveform, MIDI, analog video, and so forth.

The media type of the media data in media objects is used as the base for classifying media objects into classes. For example, the media objects having waveform type of audio data is classified as objects of *waveform audio* class. Since the messages understood by media objects are tightly related to the type of media data, a default processing is provided to each class of media objects to simplify programming.

The messages supported by media objects can be grouped into 2 categories: basic messages and media-specific messages. All media objects should understand and support *basic messages*. The *media-specific messages* are messages that are supported by only one specific class of media objects and not by others. For example, a waveform audio media object might support recording whereas a laserdisc media object does not.

Media-Basic Messages

Open	Open a media object
Close	Close a media object
Load	Load media data
QueryStatus	Query status of media object
SetParameter	Set object's attributes

Media-Specific Messages

Associate	Associate a media object to a device object
Start	Start streaming media data
Stop	Stop streaming media data
Pause	Pause streaming media data

Device-Basic Messages

Open	Open a media device
Close	Close a media device
QueryCapability	Query the capability of media device
SetStatus	Set status of media device
QueryStatus	Query status of media device
SetParameter	Set device's attributes

Device-Specific Messages

Associate	Associate a device object to a device object
Play	Start playing
Record	Start recording
Stop	Stop playing/recording
Pause	Pause playing/recording

Media Connector

A *connector* is a software representation of the physical way in which multimedia data moves from one device to another. A connector can have an *external* representation, such as a headphone jack on a CD-ROM player, or it can also have an *internal* representation, such as the flow of digital information into an audio adapter. The direction of flow is uni-directional from one object/device to another object/device.

Each connector is defined as being of a certain type (e.g. MIDI connector, audio in/out, etc.).The type and the number of connectors will be defined by each media object/device. The connectors will be connected together to stream media data from one object/device to another object/device. A connector can only be connected to its *compatible* connector and the type of each connector will define if the connection is possible. As an example, the audio out of a laserdisc player can be connected to the audio in of an amplifier mixer device.

Multimedia requires special support when being processed or displayed: multimedia data has to be available within certain time intervals to be useful for an application, it has to be handled in a uniform fashion to avoid gaps or jitter, and it has to be presented to or obtained from I/O devices at certain rates to fulfill its operational requirements. The concept of media connector will provide the mechanism to handle these requirements for dealing with multimedia and will provide a network-transparent access to multimedia data.

3 Implementation

In our prototype implementation, we have used Multimedia Presentation Manager/2 (MMPM/2^{TM}) and OS/2^{TM} as the platform for the low-level support of audio and video data. On top of this platform, a C++ class library is being developed to support our multimedia object model. Also, an object server is being developed to handle the management of multimedia objects. Fig. 1 shows an example of our multimedia class hierarchy.

4 Conclusion

We have described an object-oriented framework for supporting multimedia applications in distributed environment. This model is based on a few concepts such as media, device, and connector. By using this framework, developing a multimedia application is similar to constructing composite objects by combining and connecting basic objects. At the time of writing, the media synchronization encapsulated by the media connection abstraction is not fully developed and will be further investigated. Another future work will focus on the validation and the refinement of our framework for supporting diverse advanced multimedia applications.

```
Media
   Audio
      Waveform
      MIDI
   Video
      Digital Video
      Analog Video
         LaserDisc
         VideoTape
      Software Motion Video
   Bitmap
Device
   Waveform Device
   MIDI Sequencer
   CD-ROM Device
      CD-DA Device
      CD-XA Device
   Speaker
   Video Device
      Analog Video Device
         LaserDisc Player
         VCR
         Camera
      Digital Video Device
   Video Window
      Analog  Video Window
      Digital Video Window
Filter
   Text-to-Speech
   Speech-to-Text
```

Fig.1 Example of Multimedia Class Hierarchy